ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Кваліфікаційна наукова праця
на правах рукопису

ЄВСЄЄВ ВЛАДИСЛАВ В'ЯЧЕСЛАВОВИЧ

УДК 62.932:007.52

ДИСЕРТАЦІЯ

«МЕТОДИ ТА МОДЕЛІ КІБЕР-ФІЗИЧНОГО КЕРУВАННЯ ПРОЦЕСАМИ В ОРГАНІЗАЦІЙНО-ТЕХНІЧНИХ ВИРОБНИЧИХ ОБ'ЄКТАХ»

05.13.07 – автоматизація процесів керування

технічні науки

Подається на здобуття наукового ступеня доктора наук

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело _____ В. В. Євсєєв
Підпис

Науковий консультант: Невлюдов Ігор Шакирович, доктор технічних наук, професор

Цей примірник дисертаційної роботи
ідентичний за змістом з іншими,
поданими до спеціалізованої вченої ради Д 64.052.08

Вчений секретар спецради Д 64.052.08 _____ Плісс І. П.
Підпис, печатка

Харків 2021

АНОТАЦІЯ

Євсєєв В. В. Методи і моделі кібер-фізичного керування процесами в організаційно-технічних виробничих об'єктах. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора технічних наук за спеціальністю 05.13.07 – автоматизація процесів керування. Харківський національний університет радіоелектроніки, Міністерство освіти і науки України, Харків, 2021.

Раніше, у вітчизняній науці управління, розглядали чотири види виробничих ресурсів: трудові, матеріальні, фінансові і технічні, які складали фізичну складову технологічної системи. Вважалося, що цих чотирьох видів цілком достатньо для функціонування виробничих процесів. Інформація (кібернетична складова), хоча і враховувалася в системі управління, однак, до категорії ресурсів її не відносили. Найчастіше вона вважалася видом діяльності і тому зараховувалася до категорії функцій керування.

При визначенні поняття «виробнича система» головною вважалася її матеріальна (фізична) складова, яка визначалася через її матеріальну основу (різноманітні процеси виробництва, спрямовані на виготовлення об’єкта виробництва), а кібернетична використовувалася з метою забезпечення функціонування системи як єдиного, цілісного утворення для органічного взаємозв’язку компонентів в різноманітних процесах та у взаємодії з середовищем.

Нині, в умовах широкого застосування цифрових, мережевих і інтелектуальних технологій, постійного розвитку інтегрованих виробничих інновацій, відбуваються основні та глибокі зміни в філософії розвитку сучасної промисловості, на базі концепції Industry 4.0. Такі зміни ставлять перед виробниками завдання автоматизації процесів керування виробничими процесами в режимі реального часу, що передбачає створення єдиного

інформаційного простору підприємства, який зв'язує воєдино технологічний і бізнес рівні управління підприємством, вирішуючи при цьому безліч найважливіших для промислового підприємства завдань.

Кількість інформації, яку необхідно прийняти і оперативно обробити для формування ефективних керуючих впливів, у сучасних системах керування складними виробничими об’єктами настільки виросла, що набагато перевищує можливості існуючих автоматизованих систем управління, які нині переважно мають класичну 5-шарову архітектуру. Але у зв’язку з розвитком мережевих технологій, архітектура управління видозмінюється, рівні управління об’єднуються, може відбуватися безпосередня передача інформації датчиків у хмарні сервіси, а служби планування виробництва опрацьовують необхідні дані в режимі реального часу.

У міру розгортання досліджень в галузі нових технологій організації виробництва стає очевидним, що мова йде не про якусь єдину, що претендує на загальнонаукове значення концепцію, а про новий напрямок дослідницької діяльності, формування нового підходу до об’єкту дослідження, якими і є кібер-фізичні виробничі системи (CPPS).

Науково-технічні розробки в цій галузі ведуться, в рамках державних програм розвитку цифрового виробництва, в США, Німеччині, Японії, Франції, Австралії, Китаї та ін. Дані програми основними вимогами до керування процесами складних організаційно-технічних об'єктів визначають необхідність забезпечення: єдиного інформаційного простору керування, реалізації технології «Digital Twins», самоадаптації, самодіагностики і самообслуговування.

Але, незважаючи на стрімкий розвиток даних досліджень, розробка і впровадження CPPS все ж таки залишається індивідуальними завданнями для кожного підприємства, і, як результат, не існує єдиного підходу до процесів керування організаційно-технічними виробничими об'єктами.

Таким чином, наявним є протиріччя між індивідуальним характером

виробництва та необхідністю єдиного підходу до процесів керування організаційно-технічними виробничими об'єктами, що може бути розв’язане за рахунок розробки комплексу методів, моделей і технологій організації процесів керування організаційно-технічними виробничими об'єктами на базі кібер-фізичних виробничих систем.

Для розв’язання указаного протиріччя, актуальним є вирішення наукової проблеми розробки ефективної стратегії автоматизації процесів керування складними організаційно-технічними виробничими об'єктами, шляхом реалізації комплексу моделей, методів процесів керування і технології на базі кібер-фізичних систем.

Метою роботи є підвищення ефективності виробничого процесу шляхом розробки методів, моделей і нової технології, алгоритмічного і програмного забезпечення для реалізації управління організаційно-технічними об'єктами на базі кібер-фізичних виробничих систем.

Для досягнення мети роботи необхідно вирішити такі завдання:

– провести аналіз сучасного стану існуючих проблем, концепцій, архітектури, методологій, моделей і методів процесів керування організаційно-технічними об'єктами на базі кібер-фізичних виробничих систем;

– розробити архітектурно-логічну модель представлення процесів керування складними організаційно-технічних виробничих об'єктах на базі CPPS, з урахуванням вимог висунутих технологіями «Digital Twins»;

– розробити методи прийняття рішень на кожному етапі архітектури і технології процесів управління розробкою CPPS;

– запропонувати технологію розробки кібер-фізичних виробничих систем;

– розробити метод синтезу алгоритмів функціонування CPPS і формалізації структурних системних моделей;

– розробити модель формалізації кібернетичної складової на базі параметрів і подій GUI (графічних інтерфейсів розробника), які реалізують

HMI (людино-машинний інтерфейс) системи керування організаційно-технічними об'єктами;

– розробити синтаксичну та семантичну моделі мови визначення і опису параметрів, необхідних і достатніх для автоматизації процесів розробки CPPS;

– розробити програмне забезпечення для реалізації розроблених моделей і методів та провести експериментальні дослідження ефективності отриманих теоретичних результатів.

Об'єкт дослідження – процес кібер-фізичного керування складними організаційно-технічними виробничими об'єктами.

Предмет дослідження – закономірності, методи, моделі та технології кібер-фізичного керування організаційно-технічними виробничими об'єктами.

Під час проведення дисертаційних досліджень використовувалися: теорія мультисистем і моносистем та методи формалізованого представлення систем – для розробки архітектурно-логічної моделі декомпозиції та взаємопов'язаних методів кібер-фізичного керування процесами в складних організаційно-технічних об'єктах; методи теоретико-множинного представлення, методи структуризації та теорії графів – для визначення технології розробки кібер-фізичних виробничих систем; теорія апарату регулярних схем і алгоритмічних алгебр, інфологічна логіка предикатів та методи синтезу – для розробки метода представлення структурних системних моделей кібер-фізичного керування та методу синтезу блоків функціонування на кожному рівні архітектурно-логічної моделі; теорія системного аналізу, теорія множин, методологія візуального об'єктно-орієнтованого програмування (абстрагування, поліморфізм, інкапсуляція), методологія графічного представлення Константайна та методи організації і побудови графічного інтерфейсу користувача – для розробки моделі життєвого циклу керування організаційно-технічним об'єктом, моделі формалізації та структурного представлення кібернетичної складової організаційно-

технічного об'єкта; синтаксична і семантична моделі мов моделювання – для розробки декларативної мови визначення і маніпулювання даними предметної області, близької до деякої підмножини природної мови; теорія розширеної форми Бекуса-Наура, теорії баз знань – для реалізації експериментальних досліджень.

До нових, отриманих особисто автором, належать наступні результати:

– вперше запропоновано архітектурно-логічну модель декомпозиції кібер-фізичного керування процесами в складних організаційно-технічних об'єктах, яка на відміну від існуючих еталонних архітектурних моделей дає можливість представити керування процесом у вигляді єдиного інформаційного простору, який об'єднує в собі фізичні, кібернетичні і стратегічні складові;

– вперше запропоновано взаємопов'язані методи керування процесами у організаційно-технічних об'єктах, як логічно узгоджені послідовності прийняття рішень, які формуються на фізичному рівні за допомогою апаратних засобів, що дозволило реалізувати технологію «Digital Twins»;

– вперше запропоновано технологію розробки кібер-фізичних виробничих систем, яка дозволяє представити їх структуру як логічно пов'язану послідовність алгоритмів функціонування кожного рівня, що дає можливість реалізувати гнучкість процесу керування організаційно-технічним об'єктом;

– удосконалено метод представлення структурних системних моделей кібер-фізичного керування, що дозволило формалізувати алгоритми функціонування в системні моделі, який, на відмінну від існуючих, дає можливість побудови структурних і подієвих моделей функціонування організаційно-технічного об'єкта;

– удосконалено метод синтезу алгоритмів функціонування кібер-фізичного керування, що дозволяє об'єднати алгоритмів функціонування кожного рівня керування організаційно-технічним об'єктом в єдину систему функціонування, який, на відміну від існуючих методів, дозволяє

мінімізувати кількість операторів і спростити структуру системи функціонування організаційно-технічного об'єкту;

– удосконалено модель життєвого циклу керування організаційно-технічним об'єктом, що дає можливість автоматизувати процес керування кібернетичною складовою організаційно-технічного об'єкта, яка, на відмінну від існуючих, дозволила визначити послідовність виконання і взаємозв'язки процесів, дій і завдань, з урахуванням структури синтезованої системи функціонування організаційно-технічним об'єктом;

– вперше розроблено модель формалізації кібернетичної складової організаційно-технічного об'єкта, яка дозволяє представити людино-машинний інтерфейс HMI у вигляді взаємопов'язаних багаторівневих графічних елементів, з урахуванням параметрів і подій, що дозволило автоматизувати реалізацію заданих функцій, відповідно до вимог, які висуваються до організаційно-технічного виробничого об'єкта;

– вперше запропоновано структурне представлення кібернетичної складової організаційно-технічного об'єкта у вигляді математичного опису зв'язків між основними елементами HMI, що дозволило реалізувати представлення адитивного кібер-дизайну, за рахунок автоматизації процесу реалізації подій GUI елементів у вигляді фрагментів програмного коду;

– отримала подальший розвиток методологія сигнально-кодової конструкції, на базі якої запропонований метод графічного представлення конструкції кібернетичної складової організаційно-технічного об'єкта, який, на відміну від існуючих (методології Джексона, Гейн-Сарсон), дозволяє відображати взаємодію основних елементів HMI для редукції розробки структури;

– вперше розроблено синтаксичну і семантичну моделі декларативної мови визначення і маніпулювання даними предметної області, близької до підмножини природної мови. Запропонована мова, яка на відміну від існуючих, не вимагає від розробника знання об'єктно-орієнтованих мов високого рівня програмування, на базі якої розробляється кібернетична

складова, що істотно спрощує процес керування організаційно-технічним виробничим об'єктом.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційна робота виконана на кафедрі комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки Харківського національного університету радіоелектроніки в рамках науково-дослідних робіт: «Теоретичні основи створення перспективних компонентів мікроелектромеханічних систем та технологій їх виробництва» (ДР 0108U002216); «Теоретичні основи мікроелектромеханічних систем, проектування та технології їх виробництва для гнучких інтегрованих систем» (ДР 0110U002594); «Створення експериментальних зразків компонентів мікросистемної техніки для виробництв з інтелектуальними властивостями і їх впровадження» (ДР 0113U0003582); «Створення мікроопто-електромеханічних засобів для інтелектуальних технологічних систем промислового обладнання та робототехніки» (ДР 0115U002433), «Безскладальні гнучко-жорсткі конструкції зі змінною конфігурацією для мікросистемної техніки та інтелектуальних роботів» (ДР 0117U002529), які виконувалися у відповідності з наказами Міністерства освіти і науки України за результатами конкурсного відбору проектів наукових досліджень. В рамках зазначених тем здобувачем, як виконавцем, було розроблено моделі, методи, технології і програмне забезпечення, які дозволяють підвищити ефективність виробничого процесу шляхом покращення продуктивності і ритмічності виробництва.

Практичне значення отриманих результатів дисертації. Практичне значення отриманих теоретичних результатів підтверджено актами впровадження, що доводять коректність теоретичних положень дисертаційної роботи, високу якість розроблених моделей та методів. Результати дисертаційної роботи реалізовані у вигляді програмного забезпечення «Система розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом» та

впроваджено: у виробничий процес АТ «Мотор Січ» (акт від 15.05.2019р.); ТОВ «Науково виробниче підприємство «УКРІНТЕХ»» (акт від 23.10.2019р.); в освітній процес Кременчуцького національного університету імені М. Остроградського (акт від 04.11.2020р.), Харківського національного університету імені В. Н. Каразіна (акт від 29.10.2020р.); Національного університету «Запорізька політехніка» (акт від 23.01.2020р.).

Впровадження у виробництво запропонованих методів та моделей автоматизації керування процесами на базі кібер-фізичних виробничих систем у вигляді програмного засобу, які дозволили підвищити продуктивність на 5% та ритмічність 3% (ВАТ «Мотор Січ»), продуктивність на 1,2% та ритмічність 1,8% на місяць (ТОВ «НВП «УКРІНТЕХ»»), що підтверджено відповідними актами впровадження.

Матеріали дисертації опубліковані в 54 наукових роботах, серед них 22 наукові статті, що індексуються міжнародними наукометричними базами даних (Scopus, Cross Ref, EBSCO, Index Copernicus та іншими), з них: 18 у виданнях з технічних наук, включених до «Переліку наукових фахових видань України, в яких можуть публікуватися результати дисертаційних робіт на здобуття наукового ступеня доктора і кандидата наук», 1 – в журналі категорії А (Scopus та віднесена до третього квартиля (Q3) відповідно до класифікації Journal Citation Reports), 1 – у зарубіжному науковому періодичному виданні, що індексується у наукометричній базі даних Scopus, 2 – у наукових періодичних виданнях інших держав (з них 1 стаття у періодичному науковому виданні, що входять до Європейського Союзу), 5 – написані без співавторів; 9 свідоцтв про реєстрацію авторського права України на твір (з них 1 без співавторів); 23 тези доповідей у матеріалах міжнародних, наукових, науково-технічних та науково-практичних конференціях.

Ключові слова: організаційно-технічний об'єкт, функціонування, синтез, кібер-фізичні системи, керування процесами, методи, моделі, адитивний кібер-дизайн, гнучкість процесу керування, синтаксична модель, HMI, GUI.

Список публікацій здобувача

1. **Yevsieiev V.** Visual components formal description development for the automated design of software products and modules for computer-integrated production technological preparation systems. *Вчені записки таврійського національного університету імені В.І. Вернадського Серія: Технічні науки.* 2018. Том 29 (68) № 1 Частина 1. С.143–147. DOI: 10.31474/2075-4272-2018-1-31-24-31.

2. **Евсеев В. В.** Применение программных метрик кода на раннем этапе жизненного цикла программного обеспечения. *Восточно-европейский журнал передовых технологий*. 2011. Вып. № 1/2 (49). С.19–21.

3. **Yevsieiev V.** Conceptual scheme and basic concepts graphic representation of software and modules visual elements description in CIS TPP design automation problem solution. *Наукові нотатки. Міжвузівський збірник (за галузями знань «Технічні науки»).* 2018. Випуск 61. С. 40–47.

4. **Yevsieiev V.** Visual objects interaction mathematical presentation to solve the problem of software design automation for computer information systems of technological production preparation. *Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація»* 2018. № 1(31). С. 24–31.

5. **Yevsieiev V.** Program code automated system development at early stage of software life cycle. *Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація»*. 2017. №1(30). С.69–77. DOI: 10.31474/2075-4272-2018-1-31-24-31.

6. Невлюдов І.Ш., **Євсєєв В.В.**, Демська А.І. Розробка моделі життєвого циклу розробки програмних продукту та програмних модулів для КІС ТПВ. *Технология приборостроения.* 2017. №1. С.12–16.

7. Nevlyudov I., **Yevsieiev V.**, Miliutina S., Kolesnyk K. High – Level Programming Language Decomposition Parametric Model. *Machine Dynamics Research, Warsaw University of Technology*. 2015. Vol. 39. No 1. P.81–91.

8. Nevlyudov I., **Yevsieiev V.**, Miliutina S. Program Project Development Life Cycle Model. *Комп'ютерні системи проектування теорія і практика*. 2014. № 808. С. 26–30.

9. Невлюдов И.Ш., Андрусевич А.А., **Евсеев В.В.**, Милютина С.С., Замирец Я.О. Формализация объектно-орентированных языков програмирования. *Технология приборостроения*. 2014. №3. С.11–17.

10. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С., Бортникова В.О. Анализ моделей расчета трудоемкости программного продукта при разработке КИС ТПП. *Восточно-европейский журнал передовых технологий*. 2012. Вып.6. №2(60). С.21–24.

11. Невлюдов И.Ш., Андрусевич А.А., **Евсеев В.В.** Анализ жизненного цикла разработки программного обеспечения для корпоративных информационных систем. *Восточно-европейский журнал передовых технологий*. 2010. Вып.6/8(48). С.25–27.

12. **Євсєєв В.В.**, Андрусевич А.О., Власенков Д.П. Аналіз концепції Industry 4.0 в технології IIOT. *Технология приборостроения*. 2020, №1. С.64–68.

13. Плотникова З.В., **Евсеев В.В.** Метод нисходящего анализа с прогнозируемым выбором альтернатив для констекстно – зависимых граматик. *Восточно-европейский журнал передовых технологий*. 2009. Вып. 4/11(40). С.11–13.

14. Nevliudov I., **Yevsieiev V.**, Maksymova S., Filippenko I. Development of an architectural-logical model to automate the management of the process of creating complex cyber-physical industrial systems. *Eastern-European Journal of Enterprise Technologies*. Vol 4. No 3(106). С.44–52. DOI: 10.15587/1729-4061.2020.210761.

15. **Євсєєв В.В.**, Максимова С.С. Технологія процесу керування розробкою кібер-фізичних виробничих систем. *Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки».* 2020. Том 31(70). № 6, С.57–63.

16. Nevliudov I., **Yevsieiev V.**, Omarov M., Bronnikov A., Liashenko V. Method of Algorithms for Cyber-Physical Production Systems Functioning Synthesis. *International Journal of Emerging Trends in Engineering Research (IJETER)*, 2020. Vol. 8, No.10, P.7465–7473. DOI:10.30534 /ijeter/2020/1278102020.

17. Nevliudov I., **Yevsieiev V.**, J. H. Baker, Ahmad M. A., Lyashenko V. Development of a cyber design modeling declarative language for cyber physical production systems. *Journal of Mathematical and Computational Science*. 2021. No.1. PP.520–542. DOI:10.28919/jmcs/5152.

18. Nevliudov,I., **Yevsieiev, V.**, Demska,N., Novoselov, S. Development of a software module for operational dispatch control of production based on cyber-physical control systems. *Innovative Technologies and Scientific Solutions for Industries*. 2020. No.4(14), P.155–168. DOI:10.30837/ITSSI.2020.14.155.

19. Невлюдов І.Ш., **Євсєєв В.В.**, Бортнікова В.О. Розробка програмного модуля для автоматизованого проектування технологічного процесу виготовлення мікроелектромеханічних акселерометрів. *Системи управління, навігації та зв'язку. Збірник наукових праць*. (2015). Випуск 3(35). С 107–112.

20. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С., Бортникова В.О. Разработка графа параметрической зависимости для КИС ТПП на базе языков высокого уровня программирования. *Вестник Нац. техн. ун-та "ХПИ":сб. науч. тр. Темат. вып.: Новые решения в современных технологиях.* 2012. № 66 (972). С. 67–73.

21. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Модели жизненного цикла программного обеспечения при разработке корпоративных информационных систем технологической подготовки производства. *Вестник Нац. техн. ун-та "ХПИ" : сб. науч. тр. Темат. вып.:Новые решения в современных технологиях. – Харьков : НТУ "ХПИ"*. 2011. № 2. С. 94–101.

22. **Yevsieiev V.**, Bronnikov A. Development of databases interconnection "essences" information model for cyber-physical production systems additive cyber design creation automation. *Збірник наукових праць національного*

університету кораблебудування ім. адмірала Макарова. 2020. №3(481). P. 56–62. DOI: 10.15589/znp2020.3(481).7.

23. Мілютіна С.С., Невлюдов І.Ш., **Євсєєв В.В.** Модуль для голосового управління роботом РМ-01. *Свідоцтво про реєстрацію авторського права на твір №57666 від 17.12.2014.*

24. **Євсєєв В.В.**, Невлюдов І.Ш., Мілютіна С.С. Автоматизована система нормування «НОРМА». *Свідоцтво про реєстрацію авторського права на твір №57667 від 17.12.2014.*

25. Гурін Л.А., Невлюдов І.Ш., **Євсєєв В.В.** Програма для програмування та віддаленого управління мобільним роботом «Programming robots». *Свідоцтво про реєстрацію авторського права на твір №59439 від 24.04.2015.*

26. Горячевська Д.В., Невлюдов І.Ш., **Євсєєв В.В.**, Горячевська І.В. Комп'ютерна програма «Програма для визначення синхронного контролю температурних режимів плат на виробництві «QUAcontrol»». *Свідоцтво про реєстрацію авторського права на твір №59980 від 4.06.2015.*

27. Бортнікова В.О., Невлюдов І.Ш., **Євсєєв В.В.** Комп'ютерна програма «Автоматизована система проектування технологічного процесу виготовлення акселерометрів «АсСАМ» («АсСАМ»). *Свідоцтво про реєстрацію авторського права на твір № 65348 від 16.05.2016.*

28. Голіков М.О., Невлюдов І.Ш., **Євсєєв В.В.**, Функендорф А.О. «Модуль автоматизованого проектування конструкції роботів «Max-Robotics»». *Свідоцтво про реєстрацію авторського права на твір № 74642 від 13.11.2017.*

29. Голіков М.О., Невлюдов І.Ш., **Євсєєв В.В.**, Функендорф А.О. «Модуль автоматизованого проектування технологічних схем складання роботів «Max-CAM»». *Свідоцтво про реєстрацію авторського права на твір № 74619 від 13.11.2017.*

30. **Євсєєв В.В.** «Автоматизована система проектування програмного забезпечення для корпоративно-інформаційних систем технологічної підготовки виробництва «CAD-Programming Code»». *Свідоцтво про реєстрацію авторського права на твір № 74576 від 09.11.2017*.

31. Невлюдов І.Ш., **Євсєєв В.В.**, Бортнікова В.О., Чала О.О. «Автоматизація комп'ютероного зору та обробки відеопотоку для мобільних роботів». *Свідоцтво про реєстрацію авторського права на твір № 80306 від 16.07.18.*

32. Невлюдов І., **Євсєєв В.**, Демська А. Розробка синтаксичної та семантичної моделі мови визначення і опису даних предметної області. *Manufacturing & Mechatronic Systems 2018: Proceedings of IIst International Conference (M&MS 2018)*. (Kharkiv, 25–26 October 2018) P. 48–53.

33. **Yevsieiev V.**, Bronnikov A. Analysis of architectural models for representing the integration of cyber-physical production systems hierarchical levels. *Manufacturing & Mechatronic Systems 2020: Proceedings of IVth International Conference (M&MS 2020)*. (Kharkiv, 22–23 October 2020). P.17–19.

34. **Yevsieiev V.**, Miliutina S., Kollesnyk K. Software development Life Cycle Model . *Warsaw University of techology, Instiyte of Design Fundamenals XXIII Polish-Ukrainin conference CAD in MACHINERY DESIGN (CADMD 2015)*. (Polish, Bochnia, 9–10 October 2015). P.19–20.

35. Nevlyudov I., **Yevsieiev V.**, Miliutina S. Structured Language SQL Parametric Model Development. *CAD in Machinery Design. Implementation and Educational Issues. Proceedings of Ukrainian-Polish Conference (CADMD'2016)*. (Lviv, 21–22 October 2016). P. 49–50.

36. Nevlyudov I., **Yevsieiev V.**, Miliutina S., Kollesnyk K. Object semantic model for life cycle model "Jamp". *CAD in Machinery Design. Implementation and Educational Issues. 25 Proceedings of Polish-Ukrainian Conference (CADMD'2017).* (Polish, Bielsko Biala, 20–21 October 2017). P. 31–32.

37. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Анализ применимости математических моделей СОСОМО при разработке современных корпоративно - информационных систем технологической подготовки производства. *XI Международная научная конференция «Физические процессы и поля технических и биологических объектов»*. (Кременчук, 4–6 листопада 2011). Материалы конференции. КНУ имени Михаила Остроградского. С. 147–148.

38. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Актуальность создания систем автоматизированного проектирования технического задания на разработку программных продуктов для сложных корпоративных информационных систем технологической подготовки производства. *XII Международная научная конференция «Физические процессы и поля технических и биологических объектов»*. (Кременчук, 2–4 листопада 2012). Материалы конференции. КНУ имени Михаила Остроградского. С. 152–153.

39. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С. Параметрическая модель области видимости памяти для языков объектно-ориентированного программирования. *XIV Міжнародна науково-технічна конференція "Фізичні процеси та поля технічних і біологічних об'єктів"*. (Кременчук, 6–8 листопада 2015). Материалы конференции. КНУ имени Михаила Остроградского. С. 120.

40. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С. Разработка графов принадлежности элементов языков программирования высокого уровня. *Материалы 4-й Международной научно-технической конференции «Информационные системы и технологи ИСТ-2015»*. (Харьков, 21–27 сентября 2015). С. 88–89.

41. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Разработка модели жизненного цикла проектирования корпоративных информационных систем технологической подготовки. *Международная научно-практическая конференция «Математическое моделирование процессов в экономике и управлении инновационными проектами (ММП-2013)»*. (Алушта, 9–15

сентября 2013). ХНУРЭ. С. 139–140.

42. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Математическая модель расчета трудоемкости и стоимости программного продукта. *Proceedings XXIII International Conference "New Leading technologies in Machine Building"*. (Rybachie, 2–8 september 2013). P.24.

43. **Евсеев В.В.**, Бортникова В.О. Параметрическая модель декомпозиции структурированного языка SQL. *Сучасні проблеми радіотехніки та телекомунікацій «РТ–2013»: матеріали 9-ої міжнар. молодіжної на-ук.-техн. конф.* (Севастополь, 22–26 квітня 2013). СевНТУ. С.328.

44. **Евсеев В.В.**, Бортникова В.О. Анализ программных метрик при проектировании информационно-компьютерных систем технологии производства. *15 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке».* (Харьков, 18–20 апреля 2011). Сб. материалов форума. ХНУРЭ. С.152–153.

45. **Евсеев В.В.**, Бортникова В.О. Анализ языков высокого уровня программирования применяемых для разработки корпоративно-информационых систем технологической подготовки производства. *16 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке»*. (Харьков, 17–19 апреля 2012). Сб. материалов форума. Том № 2. ХНУРЭ. С.142–143.

46. **Евсеев В.В.**, Бортникова В.О. Параметрическая модель декомпозиции структурированного языка SQL для решения задач расчета трудоемкости. *17 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке»*. (Харьков, 22–24 апреля 2013). Сб. материалов форума. Том № 2. ХНУРЭ. С. 119–120.

47. **Yevsieiev V.**, Jijavadze O. Analysis architectural model of Industry 4.0 (RAMI 4.0). *XXIV Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті».* (Харків, 7–9 квітня 2020). Зб. матеріалів форуму. Т. 2. С.93–94.

48. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Информационная модель автоматизированной системы проектирования корпоративно-информационных систем технологической подготовки производства на ранней стадии разработки технического задания. *Перша Всеукраїнська науково-практична конференція «Актуальні проблеми створення електронних засобів промислових автоматизованих систем»* (м. Сєвєродонецьк, 25–26 жовтня 2011). С. 18–21.

49. **Yevsieiev V.**, Bronnikov A. Analysis of the cyber-physical production systems implementation impact to achieve the goals of lean production. *The IIth International scientific and practical conference «Development of scientific and practical approaches in the era of globalization»* (USA, Boston, 28–30 September. 2020). P.221–226. DOI:10.46299/ISG.2020.II.II.

50. **Yevsieiev V.**, Bronnikov A. Analysis of the CMMI model application for solving the tasks of CPPS control processes automation development. *The IV th International scientific and practical conference «Actual Trends of Modern Scientific Research»* (Germany, Munich, 11–13 October 2020). P.128–132.

51. **Yevsieiev V.**, Bronnikov A. Information systems development methodologies application analysis for cyber-physical production systems development. *III International scientific-practical conference "Theory, science and practice"* (Japan, Tokyo, 5–8 October 2020). P. 398–401. DOI: 10.46299/ISG.2020.II.III.

52. **Yevsieiev V.**, Bronnikov A. Analysis of the multi-agent systems application to solve the problem of cyberphysical production systems development. *The IV th International scientific and practical conference «Integration of scientific bases into practice»*. (Sweden, Stockholm, 12–16 October 2020). P.459 – 462. DOI:10.46299/ISG.2020.IV.

53. **Yevsieiev V.**, Bronnikov A. Structural model of a cyber-physical production system based on multi-agent systems analysis. *Матеріали VII Міжнародної науково-технічної Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування*

організаційно-технічними та технологічними комплексами», (Київ, 26 листопада 2020). С.312–313.

54. **Yevsieiev V.**, Bronnikov A. Comlexity development analysis of cyber-physical production systems for smart manufacturing. *The X th International scientific and practical conference «Trends in the development of modern scientific thought»* (Canada, Vancouver, 23–26 Nov. 2020). P.699–703. DOI:10.46299/ISG.2020.II.X.

ABSTRACT

Yevsieiev V. V. Methods and models of cyber-physical process control in organizational and technical production facilities. – Qualifying scientific work on the rights of the manuscript.

Thesis for the degree of Doctor of technical sciences in specialty 05.13.07 – Automation of control processes. – Kharkiv National University of Radio Electronics, Ministry of Education and Science of Ukraine, Kharkiv, 2021.

Earlier, in the domestic science of control, four types of production resources were considered: labor, material, financial and technical, which constituted the physical component of the technological system. It was believed that these four types are quite enough for the production processes functioning. Information (cybernetic component), although was taken into account in the control system, however, was assigned to the category of resources. More often than not, it was considered a type of activity and therefore was counted in the category of control functions.

When the definition "production system" is considered in main its material (physical) component, which is determined by its material basis (different manufacturing processes aimed at manufacturing facility production), and cybernetic used to ensure the system functioning as a single, integral formation for organic interrelation components in various processes and in interaction with the environment.

Now, in the digital widespread use context, network and intelligent technologies, the constant integrated industrial innovations development, fundamental and profound changes are taking place in the philosophy of the modern industry development, based on the Industry 4.0 concept. Such changes pose a challenge for manufacturers to automate production control processes in real time, which provides for the creation of an enterprise single information space, links together the technological and business enterprise control levels, while solving a lot of the most important tasks for an industrial enterprise.

The amount of information that must be received and promptly processed to form effective control actions has grown so much in modern control systems for complex production facilities that it far exceeds the capabilities of existing automated control systems, now they mainly have a classic 5-layer architecture. But in connection with the network technologies development, the control architecture is being modified, the control levels are combined, there can be a direct transfer of sensor information to cloud services, and production planning services work through the necessary data in real time.

As research unfolds in the field of new technologies for organizing production, it becomes obvious that this is not a single concept to be a general scientific significance, but a new research activity direction, the new approach formation to the object of research, which are cyber-physical production systems (CPPS).

Scientific and technical developments in this area are carried out within the state programs framework for the digital production development, in the USA, Germany, Japan, France, Australia, China, etc. These programs determine the main requirements for complex organizational and technical objects processes control to ensure: a unified information space control, implementation of Digital Twins technology, self-adaptation, self-diagnostics and self-service.

But, despite the rapid development of these studies, CPPS development and implementation still remains individual tasks for each enterprise, and, as a result, there is no single approach to organizational and technical production facilities control processes.

Thus, there is a contradiction between the individual production nature and the need for a unified approach to the organizational and technical production facilities control processes, can be solved by developing a set of methods, models and technologies for organizing organizational and technical production facilities control processes based on cyber-physical production systems.

To solve this contradiction, it is urgent to solve the scientific problem of developing an effective strategy for automating complex organizational and

technical production facilities control processes by implementing a set of models, control processes methods and technology based on cyber-physical systems.

The aim of the work is to increase the efficiency of the production process by developing methods, models and new technology, algorithmic and software for the organizational and technical objects control implementation based on cyber-physical production systems.

To achieve the goal of the work, it is necessary to solve the following tasks:

– to analyze the current state of existing problems, concepts, architecture, methodologies, control processes for organizational and technical objects based on cyber-physical production systems models and methods;

– to develop an architectural and logical model for representing the complex organizational and technical production facilities control processes based on CPPS, taking into account the requirements of the Digital Twins technologies;

– to develop methods for making decisions at each stage of the CPPS development control processes architecture and technology;

– to propose a technology for the cyber-physical production systems development;

– to develop a method for synthesizing algorithms for the CPPS functioning and formalizing structural system models;

– to develop a formalization model of the cybernetic component based on GUI (graphical interfaces of the developer) parameters and events, which implement the control system HMI (human-machine interface) for organizational and technical objects;

– to develop syntactic and semantic language models for the necessary parameters definition and description and sufficient to automate the CPPS development processes;

– to develop software for the developed models and methods implementation and conduct experimental studies of the theoretical results obtained effectiveness.

The object of research is the process of cyber-physical control of complex

organizational and technical production facilities.

The subject of research is the laws, methods, models and technologies of cyber-physical organizational and technical production facilities control.

During the dissertation research, the following were used: the multisystems and monosystems theory and systems formalized representation methods – to develop an architectural and logical decomposition model and cyber-physical processes control interrelated methods in complex organizational and technical objects; set-theoretic representation methods, structuring and graph theory methods – to determine the technology for the cyber-physical production systems development; regular schemes and algorithmic algebras apparatus theory, predicates infological logic and synthesis methods – to develop a method for representing structural cyber-physical control system models and a method for synthesizing functioning blocks at each architectural-logical model level; system analysis theory, set theory, visual object-oriented programming methodology (abstraction, polymorphism, encapsulation), Constantine graphical representation methodology and methods of organizing and constructing a graphical user interface – to develop a life cycle model for control an organizational and technical object, a formalization model and a structural representation the cybernetic component of the organizational and technical object; modeling languages syntactic and semantic models – for the declarative language development to define and manipulate the data of the subject area, close to a certain natural language subset; the extended Backus-Naur form theory, the knowledge bases theory – for the experimental research implementation.

New results obtained personally by the author include the following results:

– for the first time, an architectural-logical model of cyber-physical processes control decomposition in complex organizational and technical objects was proposed, which, in contrast to the existing reference architectural models, makes it possible to represent process control in the form of a single information space that combines physical, cybernetic and strategic components;

– for the first time, interconnected control processes methods in

organizational and technical objects were proposed, as logically coordinated decision-making sequences that are formed at the physical level with the hardware help, which made it possible to implement the “Digital Twins” technology;

– for the first time, a technology for cyber-physical production systems development was proposed, which makes it possible to represent their structure as a logically connected algorithms sequence for each level functioning, which makes it possible to realize the process control an organizational and technical object flexibility;

– the method of structural system cyber-physical control models representing has been improved, which made it possible to formalize functioning algorithms in system models, which, in contrast to the existing ones, makes it possible to construct structural and event models of the organizational and technical object functioning;

– the method of synthesis of algorithms for the functioning of cyber-physical control has been improved, it allows to combine the algorithms for the functioning of each level of management of an organizational and technical object into a single functioning system, which, unlike existing methods, allows minimizing the number of operators and simplifying the structure of the functioning system of an organizational and technical object;

– the life cycle model of the organizational and technical object control has been improved, which makes it possible to automate the control process of the organizational and technical object cybernetic component, which, in contrast to the existing ones, made it possible to determine the execution sequence and processes interrelation, actions and tasks, taking into account the structure of the organizational and technical object functioning synthesized system;

– for the first time, a model of the organizational and technical object cybernetic component formalization was developed, it allows to present the human-machine interface HMI in the interconnected multi-level graphic elements form, taking into account parameters and events, which made it possible to automate the specified functions implementation in accordance with the

organizational and technical production facility requirements;

– for the first time, a structural the organizational and technical object cybernetic component representation in the form of the links between the HMI main elements mathematical description was proposed, which made it possible to implement the additive cyber design representations by automating the process of GUI elements implementing events in the program code fragments form;

– the methodology of the signal-code structure was further developed, on the basis of which a method was proposed for the graphical representation of the organizational and technical object cybernetic component structure, which, in contrast to the existing ones (Jackson, Gain-Sarson methodology), allows displaying the interaction of the main HMI elements to reduce the structure development;

– for the first time, a declarative language syntactic and semantic models for defining and manipulating a subject area data, close to a subset of a natural language, have been developed. The proposed language, unlike the existing ones, does not require the developer to know high-level object-oriented languages, on the basis of which the cybernetic component is developed, which greatly simplifies the process of managing the organizational and technical production facility.

Communication of work with scientific programs, plans, topics. The dissertation work was carried out at the Department of Computer-Integrated Technologies, Automation and Mechatronics of the Kharkiv National University of Radio Electronics within the research framework "Theoretical Foundations of the Creation of Perspective Components of Microelectromechanical Systems and Technologies for Their Production" (No. 0108U002216), "Theoretical foundations of microelectromechanical systems, design and production technologies for flexible integrated systems" (No. 0110U002594), "Creation of experimental samples of microsystem equipment components for production with intellectual properties and their implementation" (No. 0113U0003582), "Creation of micro-optoelectromechanical means for intelligent technological systems of industrial equipment and robotics "(No. 0115U002433), "Non-assembly flexible-rigid

structures with variable configuration for microsystem technology and intelligent robots" (No. 0117U002529), which were carried out in accordance with the orders of the Ministry of Education and Science of Ukraine based on the research projects competitive selection results. Within the framework of these topics, the applicant, as a performer, has developed models, methods, technologies and software that make it possible to increase the efficiency of the production process by improving the productivity and rhythm of production.

The practical significance of the results of the dissertation. The practical significance of the obtained theoretical results is confirmed by the implementation acts and the dissertation work theoretical provisions correctness, the developed models and methods high quality. The dissertation work results were implemented in the form of the software "System for the development of a cybernetic component for managing an organizational and technical production facility" and introduced: into the production process of Motor Sich JSC (act dated 15/05/2019) UKRINTECH Scientific and Production Enterprise LLC (act of 10/23/2019) into the educational process of the M. Ostrogradsky Kremenchug National University (act of 11/04/2020), V.N. Karazin Kharkiv National University (act of 10/29/2020) of the Zaporozhye Polytechnic National University (act dated 01/23/2020).

Manufacturing application into production of the proposed process control automation based on cyber-physical production systems methods and models in the form of a software tool, which made it possible to increase productivity by 5% and rhythm by 3% (Motor Sich JSC), productivity by 1.2% and rhythm 1.8% per month (LLC "NPP" UKRINTECH""), which is confirmed by the relevant implementation acts.

The dissertation materials were published in 54 scientific papers, among them 22 scientific articles are indexed by international scientometric databases (Scopus, Cross Ref, EBSCO, Index Copernicus and others), of which: 18 in publications on technical sciences included in the "List of scientific professional publications Of Ukraine, in which the results of dissertations for the degree of Doctor and Candidate of Sciences can be published ", 1 – in a journal of category

A (Scopus and assigned to the third quartile (Q3) in accordance with the classification of Journal Citation Reports), the first foreign scientific periodical indexed in the scientometric database Scopus, the second scientific periodicals of other states (of which 1 is in the scientific periodicals of the European Union), 5 are written without co-authors; 9 certificates of Ukraine copyright registration for a work (of which 1 without co-authors) 23 – in materials of international, scientific, scientific-technical and scientific-practical conferences.

Keywords: organizational and technical object, functioning, synthesis, cyber-physical systems, process control, methods, models, additive cyber-design, flexibility of the control process, syntax model, HMI, GUI.

List of publications that include the main scientific results:

1. **Yevsieiev V.** Visual components formal description development for the automated design of software products and modules for computer-integrated production technological preparation systems. *Вчені записки таврійського національного університету імені В.І. Вернадського Серія: Технічні науки.* 2018. Том 29 (68) № 1 Частина 1. С.143–147. DOI: 10.31474/2075-4272-2018-1-31-24-31.

2. **Евсеев В. В.** Применение программных метрик кода на раннем этапе жизненного цикла программного обеспечения. *Восточно-европейский журнал передовых технологий*. 2011. Вып. № 1/2 (49). С.19–21.

3. **Yevsieiev V.** Conceptual scheme and basic concepts graphic representation of software and modules visual elements description in CIS TPP design automation problem solution. *Наукові нотатки. Міжвузівський збірник (за галузями знань «Технічні науки»).* 2018. Випуск 61. С. 40–47.

4. **Yevsieiev V.** Visual objects interaction mathematical presentation to solve the problem of software design automation for computer information systems of technological production preparation. *Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та*

автоматизація» 2018. № 1(31). С. 24–31.

5. **Yevsieiev V.** Program code automated system development at early stage of software life cycle. *Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація»*. 2017. №1(30). С.69–77. DOI: 10.31474/2075-4272-2018-1-31-24-31.

6. Невлюдов І.Ш., **Євсєєв В.В.**, Демська А.І. Розробка моделі життєвого циклу розробки програмних продукту та програмних модулів для КІС ТПВ. *Технология приборостроения.* 2017. №1. С.12–16.

7. Nevlyudov I., **Yevsieiev V.**, Miliutina S., Kolesnyk K. High – Level Programming Language Decomposition Parametric Model. *Machine Dynamics Research, Warsaw University of Technology*. 2015. Vol. 39. No 1. P.81–91.

8. Nevlyudov I., **Yevsieiev V.**, Miliutina S. Program Project Development Life Cycle Model. *Комп'ютерні системи проектування теорія і практика*. 2014. № 808. С. 26–30.

9. Невлюдов И.Ш., Андрусевич А.А., **Евсеев В.В.**, Милютина С.С., Замирец Я.О. Формализация объектно-орентированных языков програмирования. *Технология приборостроения*. 2014. №3. С.11–17.

10. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С., Бортникова В.О. Анализ моделей расчета трудоемкости программного продукта при разработке КИС ТПП. *Восточно-европейский журнал передовых технологий*. 2012. Вып.6. №2(60). С.21–24.

11. Невлюдов И.Ш., Андрусевич А.А., **Евсеев В.В.** Анализ жизненного цикла разработки программного обеспечения для корпоративных информационных систем. *Восточно-европейский журнал передовых технологий*. 2010. Вып.6/8(48). С.25–27.

12. **Євсєєв В.В.**, Андрусевич А.О., Власенков Д.П. Аналіз концепції Industry 4.0 в технології ІІОТ. *Технология приборостроения*. 2020, №1. С.64–68.

13. Плотникова З.В., **Евсеев В.В.** Метод нисходящего анализа с прогнозируемым выбором альтернатив для констекстно – зависимых

граматик. *Восточно-европейский журнал передовых технологий*. 2009. Вып. 4/11(40). С.11–13.

14. Nevliudov I., **Yevsieiev V.**, Maksymova S., Filippenko I. Development of an architectural-logical model to automate the management of the process of creating complex cyber-physical industrial systems. *Eastern-European Journal of Enterprise Technologies*. Vol 4. No 3(106). С.44–52. DOI: 10.15587/1729-4061.2020.210761.

15. **Євсєєв В.В.**, Максимова С.С. Технологія процесу керування розробкою кібер-фізичних виробничих систем. *Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки».* 2020. Том 31(70). № 6, С.57–63.

16. Nevliudov I., **Yevsieiev V.**, Omarov M., Bronnikov A., Liashenko V. Method of Algorithms for Cyber-Physical Production Systems Functioning Synthesis. *International Journal of Emerging Trends in Engineering Research (IJETER)*, 2020. Vol. 8, No.10, P. 7465–7473. DOI:10.30534/ ijeter/2020/1278102020.

17. Nevliudov I., **Yevsieiev V.**, J. H. Baker, Ahmad M. A., Lyashenko V. Development of a cyber design modeling declarative language for cyber physical production systems. *Journal of Mathematical and Computational Science*. 2021. No.1. PP.520–542. DOI:10.28919/jmcs/5152.

18. Nevliudov,I., **Yevsieiev, V.**, Demska,N., Novoselov, S. Development of a software module for operational dispatch control of production based on cyber-physical control systems. *Innovative Technologies and Scientific Solutions for Industries*. 2020. No.4(14), P.155–168. DOI:10.30837/ITSSI.2020.14.155.

19. Невлюдов І.Ш., **Євсєєв В.В.**, Бортнікова В.О. Розробка програмного модуля для автоматизованого проектування технологічного процесу виготовлення мікроелектромеханічних акселерометрів. *Системи управління, навігації та зв'язку. Збірник наукових прац*. (2015). Випуск 3(35). С 107–112.

20. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С., Бортникова В.О. Разработка графа параметрической зависимости для КИС ТПП на базе

языков высокого уровня программирования. *Вестник Нац. техн. ун-та "ХПИ":сб. науч. тр. Темат. вып.: Новые решения в современных технологиях.* 2012. № 66 (972). С. 67–73.

21. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Модели жизненного цикла программного обеспечения при разработке корпоративных информационных систем технологической подготовки производства. *Вестник Нац. техн. ун-та "ХПИ" : сб. науч. тр. Темат. вып.:Новые решения в современных технологиях. – Харьков : НТУ "ХПИ"*. 2011. № 2. С. 94–101.

22. **Yevsieiev V.**, Bronnikov A. Development of databases interconnection “essences” information model for cyber-physical production systems additive cyber design creation automation. *Збірник наукових праць національного університету кораблебудування ім. адмірала Макарова*. 2020. №3(481). P. 56–62. DOI: 10.15589/znp2020.3(481).7.

23. Мілютіна С.С., Невлюдов І.Ш., **Євсєєв В.В.** Модуль для голосового управління роботом РМ-01. *Свідоцтво про реєстрацію авторського права на твір №57666 від 17.12.2014.*

24. **Євсєєв В.В.**, Невлюдов І.Ш., Мілютіна С.С. Автоматизована система нормування «НОРМА». *Свідоцтво про реєстрацію авторського права на твір №57667 від 17.12.2014.*

25. Гурін Л.А., Невлюдов І.Ш., **Євсєєв В.В.** Програма для програмування та віддаленого управління мобільним роботом «Programming robots». *Свідоцтво про реєстрацію авторського права на твір №59439 від 24.04.2015.*

26. Горячевська Д.В., Невлюдов І.Ш., **Євсєєв В.В.**, Горячевська І.В. Комп’ютерна програма «Програма для визначення синхронного контролю температурних режимів плат на виробництві «QUAcontrol»». *Свідоцтво про реєстрацію авторського права на твір №59980 від 4.06.2015.*

27. Бортнікова В.О., Невлюдов І.Ш., **Євсєєв В.В.** Комп’ютерна програма «Автоматизована система проектування технологічного процесу виготовлення акселерометрів «АсСАМ» («АсСАМ»). *Свідоцтво про*

реєстрацію авторського права на твір № 65348 від 16.05.2016.

28. Голіков М.О., Невлюдов І.Ш., **Євсєєв В.В.**, Функендорф А.О. «Модуль автоматизованого проектування конструкції роботів «Max-Robotics»». *Свідоцтво про реєстрацію авторського права на твір № 74642 від 13.11.2017.*

29. Голіков М.О., Невлюдов І.Ш., **Євсєєв В.В.**, Функендорф А.О. «Модуль автоматизованого проектування технологічних схем складання роботів «Max-CAM»». *Свідоцтво про реєстрацію авторського права на твір № 74619 від 13.11.2017.*

30. **Євсєєв В.В.** «Автоматизована система проектування програмного забезпечення для корпоративно-інформаційних систем технологічної підготовки виробництва «CAD-Programming Code»». *Свідоцтво про реєстрацію авторського права на твір № 74576 від 09.11.2017*.

31. Невлюдов І.Ш., **Євсєєв В.В.**, Бортнікова В.О., Чала О.О. «Автоматизація комп'ютероного зору та обробки відеопотоку для мобільних роботів». *Свідоцтво про реєстрацію авторського права на твір № 80306 від 16.07.18.*

32. Невлюдов І., **Євсєєв В.**, Демська А. Розробка синтаксичної та семантичної моделі мови визначення і опису даних предметної області. *Manufacturing & Mechatronic Systems 2018: Proceedings of IIst International Conference (M&MS 2018)*. (Kharkiv, 25–26 October 2018) P. 48–53.

33. **Yevsieiev V.**, Bronnikov A. Analysis of architectural models for representing the integration of cyber-physical production systems hierarchical levels. *Manufacturing & Mechatronic Systems 2020: Proceedings of IVth International Conference (M&MS 2020)*. (Kharkiv, 22–23 October 2020). P.17–19.

34. **Yevsieiev V.**, Miliutina S., Kollesnyk K. Software development Life Cycle Model . *Warsaw University of techology, Instiyte of Design Fundamenals XXIII Polish-Ukrainin conference CAD in MACHINERY DESIGN (CADMD 2015)*. (Polish, Bochnia, 9–10 October 2015). P.19–20.

35. Nevlyudov I., **Yevsieiev V.**, Miliutina S. Structured Language SQL Parametric Model Development. *CAD in Machinery Design. Implementation and Educational Issues. Proceedings of Ukrainian-Polish Conference (CADMD'2016)*. (Lviv, 21–22 October 2016). P. 49–50.

36. Nevlyudov I., **Yevsieiev V.**, Miliutina S., Kollesnyk K. Object semantic model for life cycle model "Jamp". *CAD in Machinery Design. Implementation and Educational Issues. 25 Proceedings of Polish-Ukrainian Conference (CADMD'2017).* (Polish, Bielsko Biala, 20–21 October 2017). P. 31–32.

37. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Анализ применимости математических моделей COCOMO при разработке современных корпоративно-информационных систем технологической подготовки производства. *XI Международная научная конференция «Физические процессы и поля технических и биологических объектов»*. (Кременчук, 4–6 листопада 2011). Материалы конференции. КНУ имени Михаила Остроградского. С. 147–148.

38. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Актуальность создания систем автоматизированного проектирования технического задания на разработку программных продуктов для сложных корпоративных информационных систем технологической подготовки производства. *XII Международная научная конференция «Физические процессы и поля технических и биологических объектов»*. (Кременчук, 2–4 листопада 2012). Материалы конференции. КНУ имени Михаила Остроградского. С. 152–153.

39. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С. Параметрическая модель области видимости памяти для языков объектно-ориентированного программирования. *XIV Міжнародна науково-технічна конференція "Фізичні процеси та поля технічних і біологічних об'єктів"*. (Кременчук, 6–8 листопада 2015). Материалы конференции. КНУ имени Михаила Остроградского. С. 120.

40. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С. Разработка графов принадлежности элементов языков программирования высокого уровня.

Материалы 4-й Международной научно-технической конференции «Информационные системы и технологи ИСТ-2015». (Харьков, 21–27 сентября 2015). С. 88–89.

41. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Разработка модели жизненного цикла проектирования корпоративных информационных систем технологической подготовки. *Международная научно-практическая конференция «Математическое моделирование процессов в экономике и управлении инновационными проектами (ММП-2013)»*. (Алушта, 9–15 сентября 2013). ХНУРЭ. С. 139–140.

42. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Математическая модель расчета трудоемкости и стоимости программного продукта. *Proceedings XXIII International Conference "New Leading technologies in Machine Building"*. (Rybachie, 2–8 september 2013). P.24.

43. **Евсеев В.В.**, Бортникова В.О. Параметрическая модель декомпозиции структурированного языка SQL. *Сучасні проблеми радіотехніки та телекомунікацій «РТ–2013»: матеріали 9-ої міжнар. молодіжної на-ук.-техн. конф.* (Севастополь, 22–26 квітня 2013). СевНТУ. С.328.

44. **Евсеев В.В.**, Бортникова В.О. Анализ программных метрик при проектировании информационно-компьютерных систем технологии производства. *15 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке».* (Харьков, 18–20 апреля 2011). Сб. материалов форума. ХНУРЭ. С.152–153.

45. **Евсеев В.В.**, Бортникова В.О. Анализ языков высокого уровня программирования применяемых для разработки корпоративно-информационых систем технологической подготовки производства. *16 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке»*. (Харьков, 17–19 апреля 2012). Сб. материалов форума. Том № 2. ХНУРЭ. С.142–143.

46. **Евсеев В.В.**, Бортникова В.О. Параметрическая модель декомпозиции структурированного языка SQL для решения задач расчета трудоемкости. *17 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке»*. (Харьков, 22–24 апреля 2013). Сб. материалов форума. Том № 2. ХНУРЭ. С. 119–120.

47. **Yevsieiev V.**, Jijavadze O. Analysis architectural model of Industry 4.0 (RAMI 4.0). *XXIV Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті».* (Харків, 7–9 квітня 2020). Зб. матеріалів форуму. Т. 2. С.93–94.

48. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Информационная модель автоматизированной системы проектирования корпоративно-информационных систем технологической подготовки производства на ранней стадии разработки технического задания. *Перша Всеукраїнська науково-практична конференція «Актуальні проблеми створення електронних засобів промислових автоматизованих систем»* (м. Сєвєродонецьк, 25–26 жовтня 2011). С. 18–21.

49. **Yevsieiev V.**, Bronnikov A. Analysis of the cyber-physical production systems implementation impact to achieve the goals of lean production. *The IIth International scientific and practical conference «Development of scientific and practical approaches in the era of globalization»* (USA, Boston, 28–30 September. 2020). P.221–226. DOI:10.46299/ISG.2020.II.II.

50. **Yevsieiev V.**, Bronnikov A. Analysis of the CMMI model application for solving the tasks of CPPS control processes automation development. *The IV th International scientific and practical conference «Actual Trends of Modern Scientific Research»* (Germany, Munich, 11–13 October 2020). P.128–132.

51. **Yevsieiev V.**, Bronnikov A. Information systems development methodologies application analysis for cyber-physical production systems development. *III International scientific-practical conference "Theory, science and practice"* (Japan, Tokyo, 5–8 October 2020). P. 398–401. DOI: 10.46299/ISG.2020.II.III.

52. **Yevsieiev V.**, Bronnikov A. Analysis of the multi-agent systems application to solve the problem of cyberphysical production systems development. *The IV th International scientific and practical conference «Integration of scientific bases into practice»*. (Sweden, Stockholm, 12–16 October 2020). P.459 – 462. DOI:10.46299/ISG.2020.IV.

53. **Yevsieiev V.**, Bronnikov A. Structural model of a cyber-physical production system based on multi-agent systems analysis. *Матеріали VII Міжнародної науково-технічної Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами»*, (Київ, 26 листопада 2020). С.312–313.

54. **Yevsieiev V.**, Bronnikov A. Comlexity development analysis of cyber-physical production systems for smart manufacturing. *The X th International scientific and practical conference «Trends in the development of modern scientific thought»* (Canada, Vancouver, 23–26 Nov. 2020). P.699–703. DOI:10.46299/ISG.2020.II.X.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

АРМ – автоматизоване робоче місце;

БД – база даних;

ЖЦ – життєвий цикл;

ІС – інформаційна система;

ІТ – інформаційні технології;

ММ – мовна модель;

ОС – операційна система;

ОТ – операційні технології;

ПМ – програмний модуль;

ПЗ – програмне забезпечення;

ПП – програмний продукт;

ТП – технологічний процес;

ТПВ – технологічна підготовка виробництва;

ТЗ – технічне завдання;

СКБД – системи керування базами даних;

AI – штучний інтелект (англ. Artificial intelligence);

APS – система синхронного планування виробництва (англ. Advanced Planning and Scheduling);

BD – структуровані і неструктуровані дані величезних обсягів і різноманітності, а також методи їх обробки, які аналізують інформацію розподілено (англ. Big Data);

c-MES – спеціалізоване прикладне програмне забезпечення, призначене для вирішення завдань синхронізації, координації, аналізу та оптимізації випуску продукції в рамках будь-якого виробництва (англ. Collaborative Manufacturing Execution System);

CASE – набір інструментів і методів програмної інженерії для проектування програмного забезпечення (англ. Computer-Aided Software

Engineering);

CC – технології розподіленої обробки цифрових даних, за допомогою яких комп'ютерні ресурси надаються інтернет-користувачеві як онлайн-сервіс (англ. Cloud Computing);

CPPS – кібер-фізичні виробничі системи (англ. Cyber-Physical Production Systems);

CMMS – комп'ютеризована система керування технічним обслуговуванням (англ. Computerized Maintenance Management System);

CMMI – модель зрілості можливостей створення програмного забезпечення (англ. Capability Maturity Model Integration);

DCA– збір і зберігання даних (англ. Data Collection/Acquisition);

DCS – розподілена система керування (англ. Distributed Control System);

DFD – методологія графічного структурного аналізу (англ. Data Flow Diagrams);

DIKW – інформаційна ієрархія, де кожен рівень додає певні властивості до попереднього рівня (англ. Data, Information, Knowledge, Wisdom);

DPU – диспетчеризація виробництва (англ. Dispatching Production Units);

DT– цифровий близнюк (англ. Digital Twin);

GUI – система засобів для взаємодії користувача з комп'ютером, заснована на представленні всіх доступних користувачеві системних об'єктів і функцій у вигляді графічних компонентів екрану (англ. Graphical User Interface);

ERP – планування ресурсів підприємства (англ. Enterprise Resource Planning);

HMI – інженерні рішення, що забезпечують взаємодію людини-оператора з керованими ним машинами (англ. Human-Machine Interface);

Industry 4.0 – прогнозована подія, масове впровадження кібер-фізичних систем у виробництво і т.д;

IoT – це мережа пов'язаних через інтернет об'єктів, здатних збирати дані і обмінюватися даними, які надходять з вбудованих сервісів (англ. Internet of Things);

i-ERP– організаційна стратегія інтеграції виробництва і операцій, керування трудовими ресурсами, фінансового менеджменту і управління активами, орієнтована на безперервне балансування і оптимізацію ресурсів підприємства за допомогою спеціалізованого інтегрованого пакета прикладного програмного забезпечення, що забезпечує загальну модель даних і процесів для всіх сфер діяльності (англ. Enterprise Resource Planning);

JAD – методологія управління проєктами, яка передбачає тісну взаємодію замовників і виконавців, з метою взаємопорозуміння в питаннях, що стосуються, що розробляється (англ. Joint Application Development);

LAM – етап життєвого циклу архітектури (англ. Lifecycle Architecture Milestone);

LOM – етап життєвого циклу мети (англ. Lifecycle Objective Milestone);

LP – ощадливе виробництво (англ. Lean Production);

LUM – керування людськими ресурсами (англ. Labor/User Management);

M2M – загальна назва технологій, які дозволяють машинам обмінюватися інформацією, або ж передавати її в односторонньому порядку (англ. Machine-to-Machine);

MEMS – пристрої, що поєднують в собі мікроелектронні і мікромеханічні компоненти (англ. Micro-Electro-Mechanical Systems);

MSI – інтегрована людино-машинна система забезпечення інформацією (англ. Module internal shared input);

PA– аналіз ефективності (англ. Performance Analysis);

PIMS – система керування виробничою інформацією (англ. Production Information Management System);

PLC – спеціальний різновид електронної обчислювальної машини (англ. Programmable Logic Controller);

PM – керування процесами виробництва (англ. Process Management);

PoE – технологія, що дозволяє передавати до віддаленого пристрою електричну енергію разом з даними через мережу Ethernet (англ. Power over Enternet);

PTG – відстеження і генеологія продукції (англ. Product Tracking & Genealogy);

QM – управління якістю (англ. Quality Management);

RAS — контроль стану і розподіл ресурсів (англ. Resource Allocation and Status);

RED – методологія інкреметного прототипування при проєктуванні ІС (англ. Rapid Application Development);

RPA (Robotic process automation) – автоматизовані робототехнічні процеси для емуляції дій людини (англ. Robotic process automation);

RUP – методологія заснована на інтеративній моделі проєктування ІС (англ. Rational Unified Process);

SADT – методологія структурного аналізу і проєктування ІС (англ. Structured Analysis and Design Technique);

SCADA – програмний пакет, призначений для розробки або забезпечення роботи в реальному часі систем збору, обробки, відображення та архівування інформації про об'єкт моніторингу або керування (англ. Supervisory Control And Data Acquisition);

UML – уніфікована мова моделювання (англ. Unified Modeling Language);

WMS – система управління складом (англ. Warehouse Management System).

ВСТУП

Актуальність теми. Глобальна конкуренція в галузі виробництва високотехнологічних виробів характеризується скороченням життєвих циклів (ЖЦ), ускладненням технічної та технологічної підготовки виробництва (ТПВ) та підвищенням вимог до їх контролю і моніторингу в режимі реального часу. Вимоги досягнення високої якості таких виробів, потребують постійного удосконалення технологічних процесів (ТП), а також зміни структури їх керування, що є центральними чинниками успіху для виробничих компаній.

З розвитком IoT і Industry 4.0 все більш широке застосування отримує впровадження кібер-фізичних виробничих систем (CPPS – англ. Cyber-Physical Production Systems), в рамках концепцій Smart Manufacturung. В свою чергу цифровізація виробничих процесів і процесів керування вимагає обробки і маніпуляції великими обсягами різнорідних промислових даних в масштабі реального часу та протягом всього життєвого циклу виробу. Щоб використовувати промислові дані, для отримання конкурентних переваг, необхідно забезпечити гнучке, навчальне, бережливе виробництво (LP – англ. Lean Production), яке повинно бути орієнтовано на людину.

Автоматизація процесів управління розробкою нових і модернізації існуючих CPPS, в рамках концепції Smart Manufacturung, пов'язані зі складною науково-технічною задачею розробки комплексу математичних моделей і методів, які забезпечують синтез фізичних і кібернетичних властивостей ТПВ високотехнологічних виробів, а також необхідністю пошуку нових підходів до створення систем автоматизації процесів керування їх розробкою, які дозволять підвищити ефективність виробництва, за рахунок інтеграції промислових стандартів Industry 4.0 (RAMI 4.0) і підходів LP.

Істотний внесок у розвиток теоретичних і методологічних основ

розробки кібер-фізичних виробничих систем внесли: Lee J., Bagheri B., Kao H. – розробили 5C архітектуру CPPS; Jiang Jehn-Ruey – вдосконалили архітектуру 5C і на базі неї запропонували архітектуру CPPS 8C; Rasman M., Pipan M., Šimic M., Herakovič N. – запропонували модель LASFA побудови Smart Manufacturung на базі архітектурної моделі RAMI 4.0; Radanliev P., Roure D. De, Nicolescu R., Huth M. – розробили імперичну архітектурну модель інтеграції CPPS-IoE, на базі архітектури CPPS 5C; Zhang H., Zhang G., Yan Q. – запропонували розглядати CPPS, як PMDT модель, в якій основна увага фокусується на етапі виробництва в інтелектуальному цеху і є об'єднанням 5 моделей: модель визначення продукту (PDM), геометрична модель і модель форми (GSM), модель виробничих атрибутів (MAM), модель поведінки і правил (BRM) і модель об'єднання даних (DFM); Wagner T., Herrmann C., Thiede S. – розробили матрицю впливу Industry 4.0 (CPPS) на системні принципи Lean Production Systems і їх ефективне досягнення для виробництва високотехнологічних виробів; Elhoone H., Zhang T., Anwar M., Desai S. – провели дослідження і запропонували основу для розробки кібер-адитивного дизайну для CPPS, з використанням якого можна динамічно розподіляти цифрові конструкції для спрощення розробки HMI для CPPS.

Науково-технічні розробки в цій галузі ведуться в: США, Німеччині, Японії, Франції, Австралії, Китаї, в рамках державних програм розвитку цифрового виробництва.

Істотний внесок у розвиток теоретичних і методологічних основ розробки кібер-фізичних виробничих систем внесли: Lee J., Bagheri B., Kao H., J. Jehn-Ruey, Rasman M., Pipan M., Šimic M., Roure D. De, Nicolescu R., Huth M.; Zhang H., Wagner T., Herrmann C., Thiede S.; Elhoone H., Zhang T., Anwar M., Edward R. та вітчизняні вчені: Жолткевич Г.М., Осадчий С.І., Хаустова В. Є., Лисенко В.П., Хаханов В. І., Крамарєв Г. В., Варшавський О. Є., Васечко Д. Ю., Глазьєв С. Ю., Жученко А.А., Сухарєв О. С., Якубовський М. М, Ладанюк А. П., Невлюдов І. Ш. та інш.

Незважаючи на численні публікації в області автоматизації процесів

керування розробкою CPPS, на даний момент часу існує протиріччя між підвищенням ефективності розробки CPPS, з урахуванням концепцій LP, і обмеженістю існуючих методологій, математичних моделей і методів їх реалізації. Це обумовлює актуальність науково-техічної задачі розробки ефективної стратегії автоматизації управління складними організаційно-технічними виробничими об'єктами, шляхом реалізації комплексу моделей, методів процесів управління і технології на базі кібер-фізичних систем для «безлюдного виробництва».

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційна робота виконана на кафедрі комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки Харківського національного університету радіоелектроніки в рамках науково-дослідних робіт: «Теоретичні основи створення перспективних компонентів мікроелектромеханічних систем та технологій їх виробництва» (ДР 0108U002216); «Теоретичні основи мікроелектромеханічних систем, проектування та технології їх виробництва для гнучких інтегрованих систем» (ДР 0110U002594); «Створення експериментальних зразків компонентів мікросистемної техніки для виробництв з інтелектуальними властивостями і їх впровадження» (ДР 0113U0003582); «Створення мікроопто-електромеханічних засобів для інтелектуальних технологічних систем промислового обладнання та робототехніки» (ДР 0115U002433), «Безскладальні гнучко-жорсткі конструкції зі змінною конфігурацією для мікросистемної техніки та інтелектуальних роботів» (ДР 0117U002529), які виконувалися у відповідності з наказами Міністерства освіти і науки України за результатами конкурсного відбору проектів наукових досліджень. В рамках зазначених тем здобувачем, як виконавцем, було розроблено моделі, методи, технології і програмне забезпечення, які дозволяють підвищити ефективність виробничого процесу шляхом покращення продуктивності і ритмічності виробництва.

Мета роботи і завдання досліджень. Метою роботи є підвищення ефективності виробничого процесу шляхом розробки методів, моделей і нової технології, алгоритмічного і програмного забезпечення для реалізації управління організаційно-технічними об'єктами на базі кібер-фізичних виробничих систем.

Для досягнення мети роботи необхідно вирішити такі завдання:

– провести аналіз сучасного стану існуючих проблем, концепцій, архітектури, методологій, моделей і методів процесів керування організаційно-технічними об'єктами на базі кібер-фізичних виробничих систем;

– розробити архітектурно-логічну модель представлення процесів керування складними організаційно-технічних виробничих об'єктах на базі CPPS, з урахуванням вимог висунутих технологіями «Digital Twins»;

– розробити методи прийняття рішень на кожному етапі архітектури і технології процесів управління розробкою CPPS;

– запропонувати технологію розробки кібер-фізичних виробничих систем;

– розробити метод синтезу алгоритмів функціонування CPPS і формалізації структурних системних моделей;

– розробити модель формалізації кібернетичної складової на базі параметрів і подій GUI (графічних інтерфейсів розробника), які реалізують HMI (людино-машинний інтерфейс) системи керування організаційно-технічними об'єктами;

– розробити синтаксичну та семантичну моделі мови визначення і опису параметрів, необхідних і достатніх для автоматизації процесів розробки CPPS;

– розробити програмне забезпечення для реалізації розроблених моделей і методів та провести експериментальні дослідження ефективності отриманих теоретичних результатів.

Об'єкт досліджень – процес кібер-фізичного керування складними організаційно-технічними виробничими об'єктами.

Предмет дослідження – закономірності, методи, моделі та технології кібер-фізичного керування організаційно-технічними виробничими об'єктами.

Методи дослідження. Під час проведення дисертаційних досліджень використовувалися: теорія мультисистем і моносистем та методи формалізованого представлення систем – для розробки архітектурно-логічної моделі декомпозиції та взаємопов'язаних методів кібер-фізичного керування процесами в складних організаційно-технічних об'єктах; методи теоретико-множинного представлення, методи структуризації та теорії графів – для визначення технології розробки кібер-фізичних виробничих систем; теорія апарату регулярних схем і алгоритмічних алгебр, інфологічна логіка предикатів та методи синтезу – для розробки метода представлення структурних системних моделей кібер-фізичного керування та методу синтезу блоків функціонування на кожному рівні архітектурно-логічної моделі; теорія системного аналізу, теорія множин, методологія візуального об'єктно-орієнтованого програмування (абстрагування, поліморфізм, інкапсуляція), методологія графічного представлення Константайна та методи організації і побудови графічного інтерфейсу користувача – для розробки моделі життєвого циклу керування організаційно-технічним об'єктом, моделі формалізації та структурного представлення кібернетичної складової організаційно-технічного об'єкта; синтаксична і семантична моделі мов моделювання – для розробки декларативної мови визначення і маніпулювання даними предметної області, близької до деякої підмножини природної мови; теорія розширеної форми Бекуса-Наура, теорії баз знань – для реалізації експериментальних досліджень.

Наукова новизна отриманих результатів. Теоретичні та експериментальні дослідження, наведені в дисертаційній роботі, дозволили вирішити важливу наукову задачу підвищення ефективності керування

організаційно-технічними виробничими об'єктами, шляхом розробки нової технології, комплексу методів, моделей, алгоритмічного та програмного забезпечення на базі технології кібер-фізичного керування процесами в організаційно-технічних виробничих об'єктах.

До нових, отриманим особисто автором, належать наступні результати:

– вперше запропоновано архітектурно-логічну модель декомпозиції кібер-фізичного керування процесами в складних організаційно-технічних об'єктах, яка на відміну від існуючих еталонних архітектурних моделей дає можливість представити керування процесом у вигляді єдиного інформаційного простору, який об'єднує в собі фізичні, кібернетичні і стратегічні складові;

– вперше запропоновано взаємопов'язані методи керування процесами у організаційно-технічних об'єктах, як логічно узгоджені послідовності прийняття рішень, які формуються на фізичному рівні за допомогою апаратних засобів, що дозволило реалізувати технологію «Digital Twins»;

– вперше запропоновано технологію розробки кібер-фізичних виробничих систем, яка дозволяє представити їх структуру як логічно пов'язану послідовність алгоритмів функціонування кожного рівня, що дає можливість реалізувати гнучкість процесу керування організаційно-технічним об'єктом;

– удосконалено метод представлення структурних системних моделей кібер-фізичного керування, що дозволило формалізувати алгоритми функціонування в системні моделі, який, на відмінну від існуючих, дає можливість побудови структурних і подієвих моделей функціонування організаційно-технічного об'єкта;

– удосконалено метод синтезу алгоритмів функціонування кібер-фізичного керування, що дозволяє об'єднати алгоритмів функціонування кожного рівня керування організаційно-технічним об'єктом в єдину систему функціонування, який, на відміну від існуючих методів, дозволяє мінімізувати кількість операторів і спростити структуру системи

функціонування організаційно-технічного об'єкту;

– удосконалено модель життєвого циклу керування організаційно-технічним об'єктом, що дає можливість автоматизувати процес керування кібернетичною складовою організаційно-технічного об'єкта, яка, на відмінну від існуючих, дозволила визначити послідовність виконання і взаємозв'язки процесів, дій і завдань, з урахуванням структури синтезованої системи функціонування організаційно-технічним об'єктом;

– вперше розроблено модель формалізації кібернетичної складової організаційно-технічного об'єкта, яка дозволяє представити людино-машинний інтерфейс HMI у вигляді взаємопов'язаних багаторівневих графічних елементів, з урахуванням параметрів і подій, що дозволило автоматизувати реалізацію заданих функцій, відповідно до вимог, які висуваються до організаційно-технічного виробничого об'єкта;

– вперше запропоновано структурне представлення кібернетичної складової організаційно-технічного об'єкта у вигляді математичного опису зв'язків між основними елементами HMI, що дозволило реалізувати представлення адитивного кібер-дизайну, за рахунок автоматизації процесу реалізації подій GUI елементів у вигляді фрагментів програмного коду;

– отримала подальший розвиток методологія сигнально-кодової конструкції, на базі якої запропонований метод графічного представлення конструкції кібернетичної складової організаційно-технічного об'єкта, який, на відміну від існуючих (методології Джексона, Гейн-Сарсон), дозволяє відображати взаємодію основних елементів HMI для редукції розробки структури;

– вперше розроблено синтаксичну і семантичну моделі декларативної мови визначення і маніпулювання даними предметної області, близької до підмножини природної мови. Запропонована мова, яка на відміну від існуючих, не вимагає від розробника знання об'єктно-орієнтованих мов високого рівня програмування, на базі якої розробляється кібернетична складова, що істотно спрощує процес керування організаційно-технічним

виробничим об'єктом.

Практичне значення отриманих результатів дисертації. Практичне значення отриманих теоретичних результатів підтверджено актами впровадження та доводять коректність теоретичних положень дисертаційної роботи, високу якість розроблених моделей та методів. Результати дисертаційної роботи реалізовані у вигляді програмного забезпечення «Система розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом» та впроваджено: у виробничий процес АТ «Мотор Січ» (акт від 15.05.2019р.); ТОВ «Науково виробниче підприємство «УКРІНТЕХ»» (акт від 23.10.2019р.); в освітній процес Кременчуцького національного університету імені М. Остроградського (акт від 04.11.2020р.), Харківського національного університету імені В. Н. Каразіна (акт від 29.10.2020р.); Національного університету «Запорізька політехніка» (акт від 23.01.2020р.).

Впровадження розроблених в дисертаційній роботі методів і моделей в модернізацію існуючих і розробку перспективних систем керування процесами в організаційно-технічних виробничих об'єктах дозволяє скоротити час запуску виробництва та підвищити його продуктивність та ритмічність.

Викладені рішення з розробки систем автоматизації процесів керування розробкою кібер-фізичних систем також захищені авторськими свідоцтвами: №57666 від 17.12.2014р, №57667 від 17.12.2014р., №59439 від 24.04.2015 р., №59980 від 4.06.2015 р., № 65348 від 16.05.2016 р., № 74642 від 13.11.2017 р., № 74619 від 13.11.2017 р., № 74576 від 09.11.2017 р., № 80306 від 16.07.18 р.

Особистий внесок здобувача. Основні наукові результати, наведені у дисертаційній роботі, сформульовані і отримані здобувачем особисто. Наукові праці [1]–[5], [30] опубліковані без співавторів. Окремі етапи дослідження були виконані у співпраці. У публікаціях, написаних у співавторстві, внесок здобувача полягає у такому: [6] – розроблено модель та

метод синтезу візуальних компонентів HMI кібернетичної складової CPPS; [7] – запропоновано теоретичні основи побудови параметричної моделі декомпозиції мов високого рівня програмування для розробки адитивного кібер-дизайну; [8] – розроблено модель життєвого циклу «Jump»; [9] – проведена формалізація об'єктно-орієнтованих мов програмування на базі теорій формальних мов; [10, 24] – проведено дослідження моделей розрахунку вартості розробки ПП для корпоративних інформаційних систем технологічної підготовки виробництва; [11] – проведена оцінка доцільності застосування стандартів ISO/IES для вирішення завдань розробки HMI для CPPS; [12] – проведено дослідження сумісності стандартів Industry 4.0 для розробки кібер-фізичних виробничих систем; [13] – розроблено метод низхідного синтаксичного аналізу з прогнозованим вибором альтернатив для контекстно-залежних граматик при розробці граматичних і автоматних моделей мовних процесорів; [14] – розроблена архітектурно-логічна модель процесу керування розробкою CPPS та методи декомпозиції; [15] – розроблено технологію процесу керування розробкою CPPS та дерева прийняття рішень на кожному етапі; [16] – розроблено системні моделі та метод синтезу елементів CPPS; [17] – розроблено синтаксичну діаграму мови моделювання адитивного кібер-дизайну; [18] – проведено оцінку адекватності моделі та методу розробки адитивного кібер-дизайну; [19] – удосконалена модель візуальної інформаційної структури HMI; [20] – запропоновано інформаційну модель організаційно-технічних об'єктів; [21] – досліджено моделі: Waterfall, V-shaped, Protype, Rapid, Incremental та Spiral з точки зору використання при розробці кібернетичної складової CPPS; [22] – розроблена інформаційна модель для автоматизації створення адитивного кібер-дизайну кібер-фізичних виробничих систем; [23] – запропоновано моделі автоматизації управління ТП на виробництві; [25] – запропоновано етапи керування складними організаційно-технічними виробничими об'єктами на виробництві; [26] – визначено показники допусків для автоматизовані системи неруйнівного контролю; [27] – удосконалено модель візуальної інформаційної структури проектування ТП; [28] – розраховано

цільову функцію вибору модулів; [29] – запропоновано використання методів технологічних схем; [31] – розроблено метод керування.

Апробація матеріалів дисертації. Результати дисертаційної роботи апробовані на 23 міжнародних, наукових, науково-технічних, науково-практичних конференціях: 2-й, 4-й міжнародних конференціях «Manufacturing & Mechatronic Systems (M&MS)» (м. Харків, 2018, 2020р.) [32]–[33]; 23-й, 24-й, 25-й міжнародних конференціях «CAD in Machinery Design. Implementation and Educational Issues (CADMD)» (м. Львів, 2015р., Bochnia, 2016р., Bielsko Biala, 2017р.) [32]–[36]; 11-й, 12-й, 14-й міжнародних науково-технічних конференціях «Фізичні процеси та поля технічних і біологічних об'єктів» (м. Кременчук, 2011р., 2012р., 2015р.) [37]–[39]; 4-й Міжнародній науково-технічній конференції «Информационные системы и технологи (ИСТ)» (м. Харків, 2015р.) [40]; Міжнародній науково-практичній конференції «Математическое моделирование процессов в экономике и управлении инновационными проектами (ММП)» (м. Харків, 2013р.) [41]; 23-й міжнародній конференції «Новые технологии в машиностроении» (м. Рибачє, 2013) [42]; 9-й міжнародній молодіжній науково-технічній конференції «Сучасні проблеми радіотехніки та телекомунікацій (РТ)» (м. Севастополь, 2013р.) [43]; 15-му, 16-му, 17-му, 24-му Міжнародному молодіжному форумі «Радіоелектроніка та молодь у XXI столітті» (м. Харків, 2011, 2012, 2013, 2020) [44]–[47]; 1-й Всеукраїнській науково-практичній конференції «Актуальні проблеми створення електронних засобів промислових автоматизованих систем» (м. Сєвєродонецьк, 2011р.) [48]; IIth International scientific and practical conference «Development of scientific and practical approaches in the era of globalization» (Boston, USA, 2020р.) [49]; IVth International scientific and practical conference «Actual Trends of Modern Scientific Research» (Munich, Germany, 2020р.) [50]; III International scientific-practical conference «Theory, science and practice» (Tokyo, Japan, 2020р.) [51]; IVth International scientific and practical conference «Integration of scientific bases into practice» (Stockholm, Sweden, 2020р.) [52]; 7-й Міжнародній науково-технічній Internet-

конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами» (м. Київ, 2020р.) [53]; Xth International scientific and practical conference «Trends in the development of modern scientific thought» (Vancouver, Canada, 2020р.) [54].

Публікації. Результати дисертації опубліковані в у 54 наукових роботах з яких: 22 наукові статті, що індексуються міжнародними наукометричними базами даних (Scopus, Cross Ref, EBSCO, Scientific Indexed Service, Index Copernicus, Academic Resource Index, Google Scholar, Open Academic Journals Index, General Impact Factor, Scholar Steer, Ulrich‘s Periodicals Directory, Directory Indexing of International Research Journals, Bielefeld Academic Search Engine, WorldCat, DOAJ, ResearchBib, Polska Bibliografia Naukowa), з них: 18 у виданнях з технічних наук, включених в «Перелік наукових фахових видань України, в яких можуть публікуватися результати дисертаційних робіт на здобуття наукового ступеня доктора і кандидата наук», 1 – в журналі категорії А (Scopus та віднесена до третього квартиля (Q3) відповідно до класифікації SCImago Journal and Country Rank або Journal Citation Reports), 1 – у зарубіжному науковому періодичному виданні що індексуються у наукометричній базі даних Scopus, 2 – у наукових періодичних виданнях інших держав (з них 1 стаття у періодичному науковому виданні, що входять до Європейського Союзу), 5 – написані без співавторів; 9 свідоцтв про реєстрацію авторського права на твір України; 23 тези доповідей у матеріалах міжнародних, наукових, науково-технічних, науково-практичних конференцій.

Структура роботи. Дисертація складається зі вступу, шести розділів, висновків, списку використаних джерел і чотирьох додатків. Повний обсяг роботи складає 405 сторінок, що містять 13 таблиць, 105 рисунків (з них 4 таблиці та 7 рисунків на 13 окремих сторінках), 238 найменувань списку використаних джерел на 34 сторінках і 4 додатків на 26 сторінках.

1 ДОСЛІДЖЕННЯ СУЧАСНИХ МОДЕЛЕЙ ТА МЕТОДІВ КЕРУВАННЯ СКЛАДНИМИ ОРГАНІЗАЦІЙНО-ТЕХНІЧНИМИ ВИРОБНИЧИМИ ОБ'ЄКТАМИ

1.1 Концепція Industry 4.0 в технології IIoT

Тенденції розвитку сучасного світу ведуть до збільшення обсягів інформації, підвищення вимог до її точності і своєчасного подання для аналізу і ухвалення рішень в режимах реального часу, вимагають перегляду підходів до використання високих технологій та їх ролі в різних сферах діяльності людини, що, у свою чергу, потребує змінити підходи до промислових технологій.

Провідні країни в галузі інноваційних технологій в промисловості запропонували нову концепцію стратегії цифрової революції – Industry 4.0 [1-9]. У Німеччині концепція Industry 4.0 підтримується міністерствами Federal Ministry of Education and Research Germany (BMBF) [10] і Federal Ministry for Economic Affairs and Energy Germany (BMWi) [11].

В роботе [12] обгрунтовується актуальність впровадження концепції Industry 4.0 в пріоритетні сфери промисловості України.

Дана концепція, за даними IoT Analytics (https://iot-analytics.com/), підтримана трьома провідними міжнародними галузевими організаціями: BitKom (інформаційні технології) [13-15], VDMA (машинобудування) [16-18] і ZVEI (електротехніка) [19] та провідними вендорами сфер застосування технологій Industry 4.0 [20].

Основними представниками на ринку Industry 4.0 є ABB (Швейцарія), Mitsubishi, Yaskawa, FANUC (Японія), KUKA, Siemens (Німеччина), General Electric, IBM, Cisco, Microsoft, Stratasys, Google, Intel, HP, Ansys, AIBrain, SAP, Amazon Web Services і General Vision (США) [21].

Аналізуючи вендорів, можна умовно виділити групу корпорацій і

підприємств, сферою діяльність яких є виробництво виробів різного призначення від мікрочіпів до складних високотехнологічних пристроїв, в які інтегровані елементи технології Industrial Internet of Things (IIoT) [22-24].

Забезпечення виробничого циклу високотехнологічних пристроїв неможливе без впровадження систем автоматизації на всіх рівнях виробництва і його супроводу, які в синтезі дозволяють створити єдиний інформаційний простір Smart Factory або Smart Manufacturung [25–28].

Національний інститут стандартів і технологій США (NIST) визначає поняття Smart Factory або Smart Manufacturung як повністю інтегровані корпоративні виробничі системи, здатні в реальному масштабі часу реагувати на мінливі умови керування процесами виробництва, мереж поставок і задоволення потреб клієнтів [29]. Для досягнення даної інтеграції необхідно використовувати дивергенцію інформаційно-комунікаційних технологій (ICT) [30–33], операційних технологій (OT) [34–38] і кібер-фізичних виробничих систем (CPPS) [32, 39–47] на всіх етапах виробництва високотехнологічної продукції.

Але якщо розглянути цю дивергенцію більш детально то можна зауважити, що вона включає в себе: хмарні технології (CC) [4, 48–49], великі дані (Big Data) [50–52], механізми штучного інтелекту (AI) [49, 53], аналіз даних на кордоні мереж (туманні і приграничні обчислення) [54], мобільну передачу даних [55, 56], мережеві технології [57, 58], інтерфейси користувачів (HMI) [59–61], SCADA системи [62–64], системи управління (c-MES, i-ERP) [65–69], програмовані логічні контролери (PLC) [63, 70–71], датчики і виконавчі механізми (MEMS) [72–74], автоматизовані робототехнічні процеси (RPA) [75], автономні роботи (AR) [76].

У роботах [77-78] запропоноване схематичне уявлення Smart Manufacturung в компонентах Industry 4.0, яке представлено на рис. 1.1.

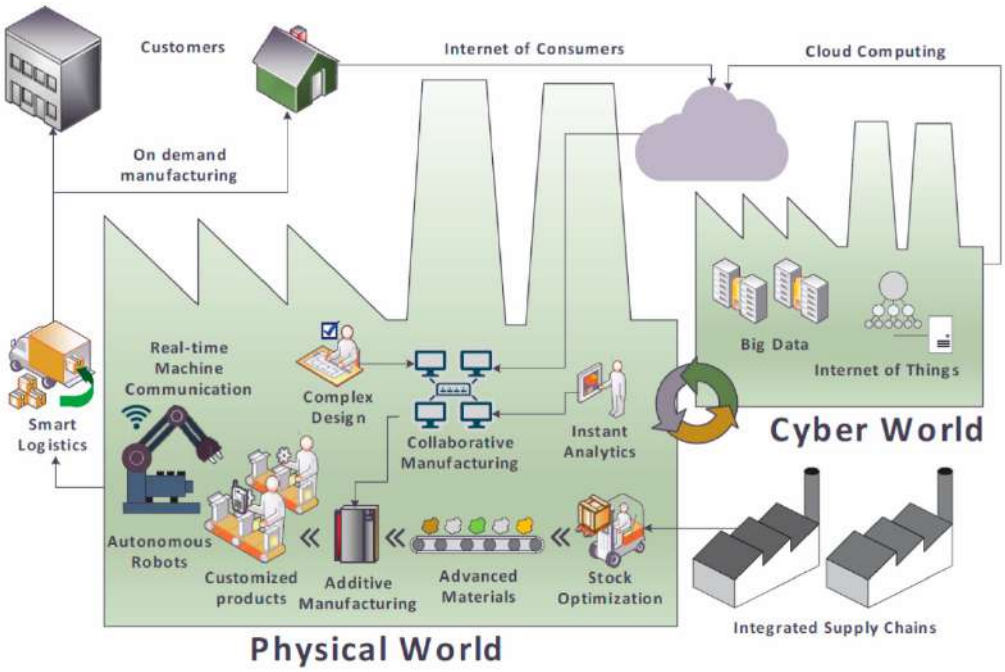


Рисунок 1.1 – Схематичне представлення Smart Manufacturing в компонентах Industry 4.0 [77, 78]

Представлене схематичне уявлення Smart Manufacturung повністю відображає основну концепцію німецького стандарту Reference Architecture Model Industrie 4.0 (RAMI4.0) [79]. Його основна концепція полягає у створенні правил опису інформаційних даних і параметрів технічного об'єкта у формі еталонної моделі архітектури Industry 4.0 (RAMI4.0), яка дає уявлення даного технічного об'єкта з усіма відповідними аспектами, від створення до виробництва і використання аж до утилізації. Це є компонент Industry 4.0, який фактично представляє її як об'єднання фізичного та кібернетичного світу з впровадженням технології їх комунікації між собою [80, 81].

Smart manufacturing-Reference Architecture Model Industrie 4.0 (RAMI4.0) [82] описує модель еталонної архітектури у формі моделі кубічного рівня, яка показує технічні об'єкти (активи) у вигляді шарів, що дозволяє їх описувати, відстежувати протягом усього терміну служби (або «vita»). У ньому також описуються структура і функції компонентів Industry 4.0, як найважливіших частин віртуального і фізичного представлення.

Аналізуючи [83] можна виділити основні вимоги до загальних концепцій і процедур специфікації технологій, орієнтованих на обслуговування еталонних архітектур Industry 4.0, на базі концепції IoT і Internet-of-Services (IoS).

У свою чергу, стандарт [84] описує основні концепції структури Smart Manufacturung.

Стандарт [85] описує область управління виробничими операціями (рівень 3) і її дії, а також вміст інтерфейсу та пов'язані транзакції на рівні 3, а також і між рівнем 3 і рівнем 4. Цей опис забезпечує інтеграцію між виробничими операціями, областю управління (рівні 3, 2, 1) і доменом підприємства (рівень 4). Його метою є підвищення одноманітності і узгодженості термінології інтерфейсів, а також зниження ризику, вартості і помилок, пов'язаних з реалізацією цих інтерфейсів.

Стандарт [86] є розширенням всієї серії стандартів OPC Unified Architecture, який визначає інформаційну модель, пов'язану з пристроями. Запропоновані три моделі, які базуються одна на одній:

– (базова) модель пристрою, призначена для забезпечення єдиного уявлення пристроїв;

– модель зв'язку пристроїв, яка представляє інформаційні елементи мережі і з'єднання, для створення топології зв'язку;

– нарешті, модель хоста інтеграції пристроїв, яка додає додаткові елементи і правила, необхідні хост-системам для управління інтеграцією всієї системи. Це дає змогу побачити топологію системи автоматизації з пристроями, а також з мережами зв'язку [87].

1.2 Дослідження структури кібер-фізичних виробничих систем в Industry 4.0

При визначенні поняття «виробнича система» головною вважалась її матеріальна (фізична) складова, яка визначалась через її матеріальну основу

(різноманітні процеси виробництва, спрямовані на виготовлення об’єкта виробництва), а кібернетична використовувалась з метою забезпечення функціонування системи як єдиного, цілісного утворення для органічного взаємозв’язку компонентів в різноманітних процесах та взаємодії з середовищем.

Впровадження концепції Industry 4.0 сприяє тому, що кібернетичний аспект виробничого процесу стає головним, що дозволяє впроваджувати новий підхід до реалізації виробництва, як синтезу кібернетичних і фізичних складових – кібер-фізичні виробничих систем керування (CPPS), які забезпечують: єдиний інформаційний простір керування, реалізацію технології «Digital Twins», самоадаптацію і само самодіагностику [88].

Їх постійний розвиток та ускладнення призвів до підвищення ступеня автоматизації і перерозподілу функцій між людиною і технікою, що загострило проблему взаємодії людини-оператора з системою керування (рис. 1.2).

Як можна побачити з рисунка 1.2, CPPS – синтез HPS (Human Physical system) та HCR (Human Cyber system), дозволяє реалізувати новий підхід до керування складними організаційно-технічними виробничими об'єктами, за рахунок виключення ручного керування процесами (Manual control). Це можливо завдяки використанню кібернетичної складової (Cyber system), з якою людина взаємодіє через людинно-машинний інтерфейс (Hunman Machine Interface), де відбувається передача команд керування та інформації про стан між кібернетичною та фізичною системами, за рахунок технології M2M [89].

У результаті дослідження інтеграції HPS та HCS (рис. 1.2) були детально розглянуті структурні зв'язки HPS, які представлені на рисунку 1.3.

Як можно бачити з рисунку 1.3, основні функції контролю та аналізу процесів, що відбуваються у фізичній системі, виконує людина, яка напряму, через виконавчі пристрої або в ручному режимі, впливає на керуваня технологічними процесами (ТП).

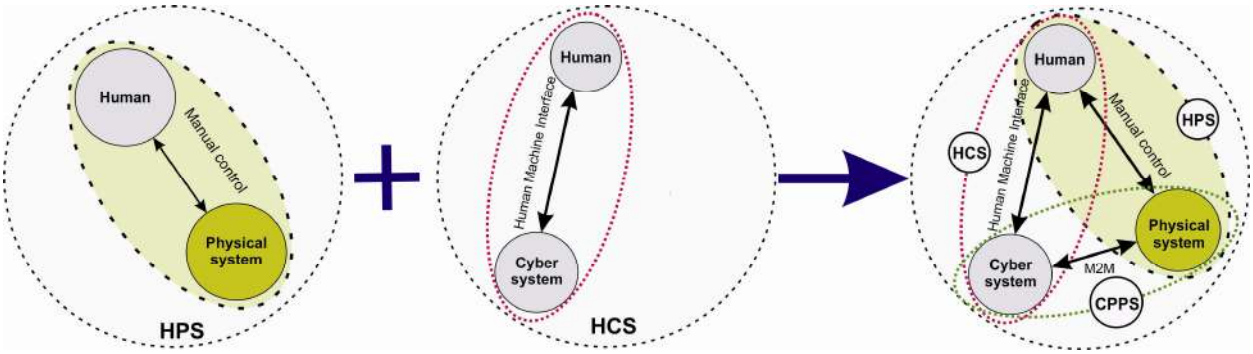


Рисунок 1.2 – Інтеграція HPS та HCS у CPPS

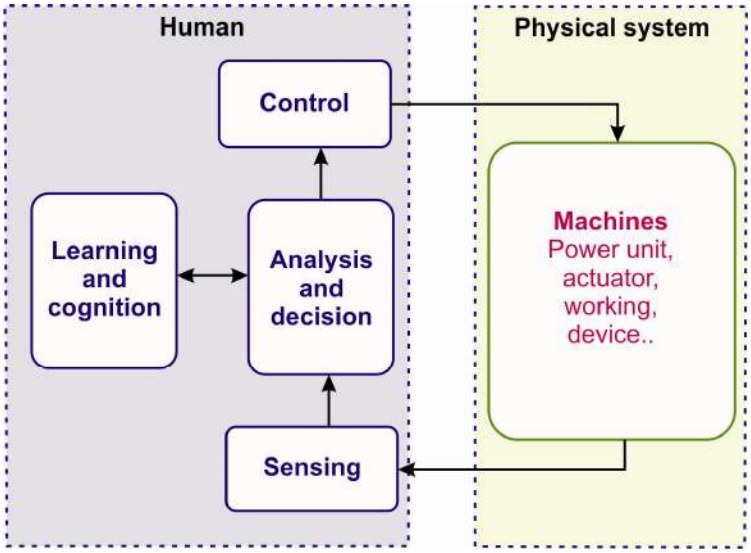


Рисунок 1.3 – Структура HPS

На сучасному рівні виробництва високотехнологічних виробів, в якому основними параметрами є швидкість, надійність та ритмічність, використання системи HPS є архаїчним та економічно невигідним.

Інтеграція в дану систему кібернетичної складової дозволяє автоматизувати керування виробничими процесами, завдяки використанню інтелектуальних мехатронних модулів, експертних систем та великих масивів данних для прогнозування виробництва.

В даному контексті людина керує процесами на фізичному рівні через кібернетичну систему. При цьому дана система з мінімальною участю людей дозволяє: автоматично контролювати ТП на фізичному рівні, аналізувати та приймати рішення у масштабі реального часу, накопичувати великий масив технологічних даних [90]. За рахунок цього відбувається вдосконалення

керування виробничими процесами, де основною функцією людини є моніторинг та керування експертними даними.

Структура зв'язків в CPPS представлена на рисунку 1.4.

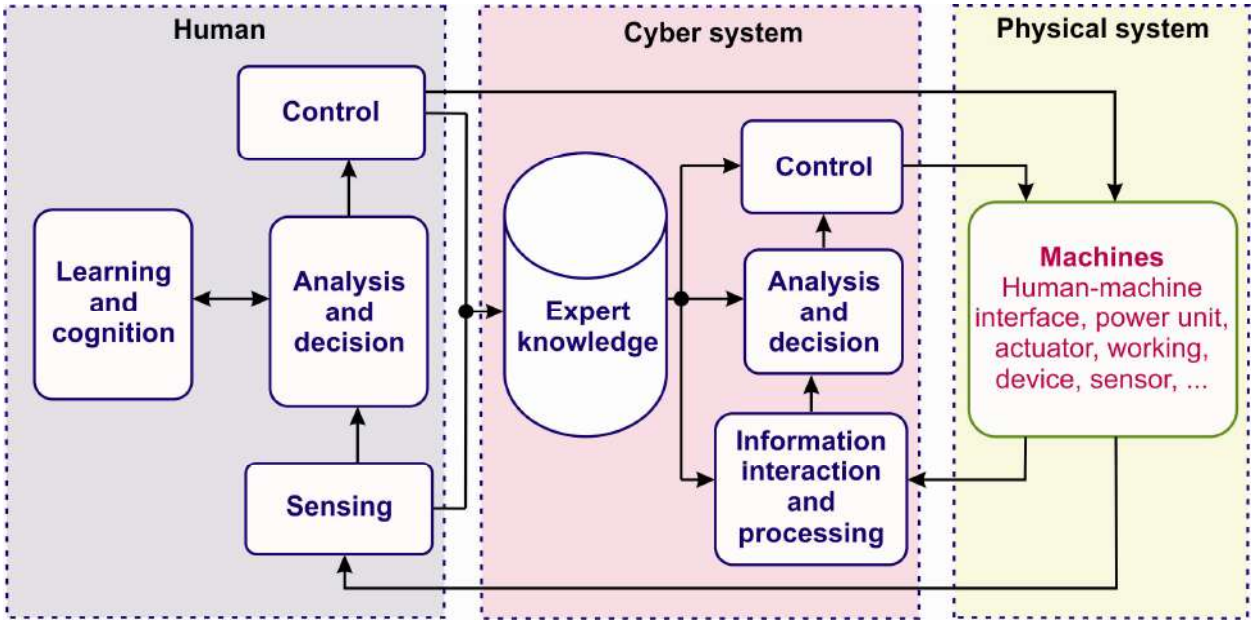


Рисунок 1.4 – Структура CPPS

З урахуванням теорії і технології існуючих кібер і фізичних систем, розробка та управління CPPS є складною науково-технічною і дослідницькою задачею [87].

Майбутні промислові системи можуть бути реалізовані з використанням CPPS, що є об'єднанням кібернетичної і фізичної складових в єдиному інформаційному просторі, за допомогою мережевої структури, для спільного виконання заданих функцій, незалежно від області їх застосування [91].

У [92] визначено, що основними факторами розвитку і впровадження CPPS – є скорочення витрат і часу процесу виготовлення виробів. Це також стосується аналізу типів систем і пов'язаних з ними процесів переходу від мехатроніки до CPPS і IoT систем.

Далі автори [92] розглядають вимоги щодо методології створення CPPS, в рамках якої розробники повинні зосередитися не тільки на окремих

фізичних і обчислювальних компонентах, але також на їх інтеграції та взаємодії.

Однією з найбільших проблем в розробці CPPS є їх внутрішня складність, неоднорідність і міждисциплінарний характер. Нові розподілені CPPS об'єднують широкий спектр різнорідних аспектів, таких як: фізична динаміка, керування процесами виробництва, машинне навчання і обробка помилок. Крім того, системні компоненти часто розподілені по декількох фізичних місцях розташування, апаратних платформ і мережах зв'язку [93].

В [94] розглядається проблема CPPS з точки зору безперервного генерування великих обсягів даних, який вимагає обробки і візуалізації для підвищення масштабованості, безпеки і ефективності CPPS, з метою удосконалення процесів керування організаційно-технічними виробничими об'єктами і досягнення повної автономії, в рамках технології Industry 4.0.

По суті, термін «кібер-фізичні виробничи системи» представляє собою архітектурну парадигму, в якій технології всеосяжного зондування є фундаментальною частиною. Започаткований в області комп'ютерних наук термін «кібер-фізичні системи» був адаптований до дуже різних галузей, таких як теорія управління або електронна інженерія. Деякі автори [95,96] розуміють CPPS як особливий сценарій IoT, заснований на усепроникаючому зондуванні та пропонують визначення CPPS яке б включало всі функції, описані в різних областях.

Запропоноване визначення CPPS, в рамках Industry 4.0, відповідає опису запропонованому міжнародними грантами:

– Trade and Invest «Industry 4.0» віднесено до технологічної еволюції від вбудованих систем до кібер-фізичних систем [97];

– McKinsey «Industry 4.0» – це наступний етап оцифровки виробничого сектора [98];

– Boston Consulting Group «Industry 4.0» відноситься до конвергенції і застосування дев'яти цифрових технологій: передової робототехніки, адитивного виробництва, доповненої реальності, моделювання, горизонтальної / вертикальної інтеграції, IIoT, хмарних даних, кібербезпеки,

Big Date і аналітики [99].

Виходячи з цього можна з упевненістю стверджувати, що дані системи є унікальними і їх математичне, інформаційне, функціональне і алгоритмічне забезпечення напряму залежать від специфіки виробництва, обладнання та вимог, які висуваються Smart Manufacturung.

Багато авторів, таких як Pericles Loucopoulos, Evangelia Kavakli, Natalia Chechina [100]; O. Cardin [101]; S. Vijayakumar, N. Dhasarathan, P. Devabalan, C. Jehan [102]; HermannMeissner, Jan C. Auricha [103]; Fazel Ansari, Robert Glawar, Tanja Nemeth [104] розкривають питання і завдання пов'язані з використанням CPPS, в рамках Smart Manufacturung, і визначають термін кібер-фізичних виробничих систем.

Зростаюча складність процесів керування організаційно-технічними виробничими об'єктами вимагає відповідних архітектур, які забезпечать їх гнучку адаптацію під час роботи.

CPPS надають засоби для подолання складності і гнучкості керування організаційно-технічними виробничими об'єктами, але інтеграція даних складових з існуючими системами керування все ще залишається проблемою. Термін CPPS визначає мехатронні системи (фізичний світ) в поєднанні з програмними об'єктами і цифровою інформацією (кібер-частина), та дозволяє реалізувати концепцію Smart Manufacturung для парадигми Industry 4.0 (I4.0).

Але все ж таки, дослідження проведені у [100–104], в основному, носять концептуальний характер, або знаходяться на ранній стадії та подаються лише пропозиції щодо еталонної архітектури.

Тому для досягнення практичної реалізації CPPS керування організаційно-технічними виробничими об'єктами потрібні систематичні моделі, методи керування процесами і технологіями, збору, обробки та застосування даних. Це пов'язано з тим, що CPPS може бути успішно реалізовані тільки тоді, коли розроблені всі критерії обробки і методи застосування різноманітних даних, визначена послідовність керування процесами в реальному часі через характер технології виробничих процесів.

Різні технології та системи були розроблені для збору необроблених даних цеху, але вони в основному спрямовані на автоматизацію виробництва.

Таким чином, роблячи висновок з аналізу публікацій, можна визначити існування складної науково-дослідної задачі систематизації знань кібер-фізичного керування процесами в складних організаційно-технічних виробничих об'єктах, яка буде слугувати ядром для створення систем керування процесами у виробничих об'єктах на базі CPPS для різних сфер виробничої діяльності людини [105–108].

В рамках даних досліджень пропонується прийняти наступне визначення поняття CPPS – це інформаційно-технологічна концепція, що передбачає інтеграцію обчислювальних ресурсів і фізичних сутностей будь-якого виду.

У CPPS обчислювальна компонента розподілена по всій фізичній системі, яка є її носієм, і синергетично ув'язана з її складовими елементами. Грунтуючись на даному визначенні можна зробити висновок що CPPS – це складна багаторівнева жорстко ієрархічна система, яка об'єднує в собі всі необхідні і достатні потоки інформації від апаратної складової (фізичної, мехартонної системи) до верхнього рівня візуалізації, аналізу і прийняття рішення (кібер системи).

Один з підходів використання теорії розподіленого управління є підхід, який заснований на багатоагентних системах (MAS – Multi-agent system) [109], що використовується для вирішення завдання кібер-фізичного керування організаційно-технічними виробничими об'єктами.

Для дослідження MAS була запропанованна наступня класифікація: від традиційних систем автоматизації керування організаційно-технічними виробничими об'єктами до CPPS. Тому авторам [110] запропонований шаблон, який складається з переліку критеріїв класифікації, затверджених експертами German Agent Systems committee (FA 5.15).

Оскільки всі підходи були створені для використання в різних областях і на різних рівнях піраміди автоматизації керування організаційно-технічними виробничими об'єктами, їх значна частина сконцентрована на

забезпеченні гнучкості або мінливості (Flexibility / changeability (FC)) системи. Інші концентруються на таких функціях: надійність (Reliability (RL)), адаптивність або спритність (Adaptability / agility (AA)), реконфігація (Reconfigurability (RC)) і надійність (Dependability (DP)). Опис характеристик MAS для CPPS представлений в таблиці 1.1.

Таблиця 1.1 – Основні характеристики MAS для CPPS

Характеристика	Опис
1	2
Flexibility/changeability (FC) [108,110]	Ступінь, з якою система буде ефективною, дієвою, вільною від ризику і задовільною в ситуаціях, що виходять за рамки початково зазначених вимог
Reliability (RL) [110]	Набір атрибутів, які впливають на здатність кібернетичної складової CPPS підтримувати свій рівень продуктивності, в зазначених умовах, протягом зазначеного періоду часу (чотири атрибути: зрілість, відмовостійкість, можливість відновлення, надійність)
Reconfigurability (RC) [111]	Атрибут швидкої зміни структури, компонентів фізичної та кібернетичної складових, для швидкого налаштування виробничих потужностей і функціональності, залежно від раптових змін вимог
Adaptability/agility (AA) [108]	Ступінь здатності системи «виживати» в конкурентному середовищі з безперервними і непередбачуваними змінами та ефективно реагувати на мінливі вимоги розроблені для клієнтів
Dependability (DP) [111]	Набір незалежних атрибутів виробничих подій (ES), який повністю визначає доступні процеси у виробничій системі

За результатами порівняння існуючих системних архітектур і їх основних напрямків, стосовно конкретних вимог до CPPS і RAMI 4.0 (табл. 1.2), а також порівняння структури підходів CPPS щодо класів типів механізмів управління і прийняття рішень, можна зробити висновки, що

майже всі архітектури концентруються на забезпеченні гнучкості систем керування організаційно-технічними виробничими об'єктами.

Для даного напрямку CPPS, згідно [112–117], існує п'ять ключових вимог: незалежність додатків (вимога 1.1), що означає, що MAS, його протоколи і повідомлення повинні бути незалежними від конкретного додатка.

Незалежність від рівня (вимога 1.2) вказує, що всі рівні автоматизації для ISA-95.00.01-2000 [118] доступні, в залежності від сценаріїв, в яких буде застосовуватися CPPS.

Незалежна від платформи реалізація (вимога 1.3) має на увазі, що модулі легко інтегруються з незалежною реалізацією (відкриті технології).

Стійкість до помилок (вимога 1.4) означає, що MAS повинні реагувати на несправності та динамічні умови відповідним чином, тобто вони повинні бути стійким до непередбачених обставин.

Децентралізація (вимога 1.5) означає, що MAS повинна мати справу з тимчасовою втратою мережевого з'єднання і критичні дані повинні бути розподілені між кількома вузлами.

Що стосується моделі RAMI 4.0, існують інші п'ять найважливіших вимог до концепції компонентів Industry 4.0 [82, 83, 119].

Підмоделі (вимога 2.1) повинні підтримувати різні інженерні дисципліни.

Границя системи (вимога 2.2) має на увазі, що підмодель описує відносини між рівнями RAMI 4.0.

Принцип вкладеності (вимога 2.3) для конкретної підмоделі повинен мати свої власні принципи організації для відповідних ресурсів (активи в ієрархічних вимірах).

Віртуальне уявлення (вимога 2.4), адміністративна оболонка, яка може позначати цифровий актив з його частинами.

Нарешті, функціональні властивості (вимога 2.5) вимагають, щоб маніфест мав доступний ззовні набір метамоделей, що описують його

функції.

Аналізуючи таблицю 1.2 можна виділити [115], в якій запропонована адаптована структурна модель CPPS на базі структурної моделі [120], яка представлена на рисунку 1.5.

Варто зауважити, що запропонована структурна модель CPPS є повністю гетерархічною, всі елементи знаходяться в різноманітних, але рівноцінних зв'язках, тому не існує переважаючого способу їх структурування. Будь-яка структура гетерархії сприймається розробником неповною, що супроводжується існуванням суперечливості, а отже, неприйнятною для вирішення задач процесів керування складними організаційно-технічними виробничими об'єктам на базі CPPS.

А як видно з таблиці 1.2 не існує системних архітектур, які б мали доступний набір метамоделей, що описують його функціональні і нефункціональні властивості та показує, що існуючі рішення мають суто декларативний опис і носять рекомендаційний характер.

Грунтуючись на результатах дослідження можна зробити висновок, що запропонована структурна модель, демонструє зв'язки, але не показує послідовність, правила процесів керування організаційно-технічним виробничим об'єктом на базі CPPS [121].

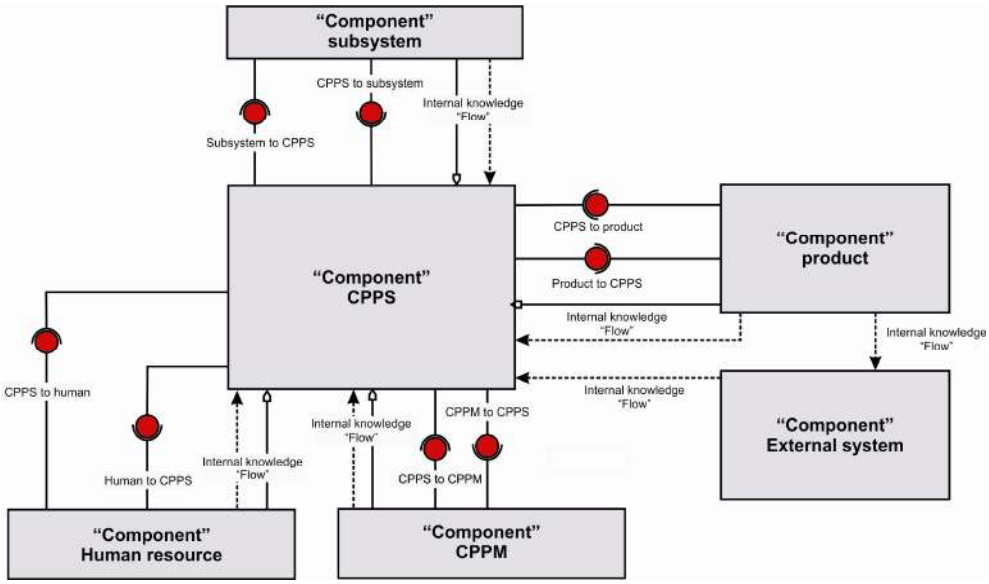


Рисунок 1.5 – Структурна модель CPPS [122]

Таблиця 1.2 – Порівняння існуючих системних архітектур і їх основних напрямків до CPPS і RAMI 4.0

Автор(и)	Характерні переваги					Класифікація		Вимоги CPPS					Вимоги RAMI 4.0				
	FC	RL	RC	AA	DP	Сфера	Тип	1.1	1.2	1.3	1.4	1.5	2.1	2.2	2.3	2.4	2.5
S. Cruz [116]	+	–	+	+	–	CPPS архітектура	II	+	+	+	+	+	–	–	+	–	–
J. Fischer	–	–	–	–	+	MFS на основі агентів	III	+	+	+	+	–	–	–	+	–	–
A. Lüder [113]	+	+	–	–	–	MAS для промисловості	III	+	+	+	–	–	–	–	+	–	–
S. Rehberger [114]	+	–	–	–	+	MAS для промисловості	III	+	+	+	+	–	–	–	+	–	–
L. Ribeiro [115]	+	+	+	+	–	CPPS архітектура	III	+	+	+	+	+	+	+	+	+	–

Примітка:

"+" – може застосовуватися;

"–" – не застосовують або описаний дуже слабо;

Тип II – напівгетерархічна система керування (наприклад, архітектура ADACOR);

Тип III – повністю гетерархічна система керування (наприклад, архітектура D-MAS).

1.3 Industry 4.0 і кібер-фізичні виробничі системи як засіб досягнення мети Lean Production

Основними ключовими технологіями Industry 4.0, в рамках Smart Manufacturung, є застосування CPPS, як результату замкнутого циклу: збору даних про фізичні процеси з мехатронних пристроїв (датчиків) у поєднанні з програмною (кібернетичною) обробкою даних про технологічні процеси та їх візуалізації (HMI) для прийняття рішення.

На основі елементів CPPS реалізуються зв'язки: збір даних, обробка даних, міжмашинні зв'язки (M2M) і взаємодієя людини з машиною (HMS), що дає можливість реалізації децентралізованого автономного керування процесами складними організаційно-технічними виробничими об'єктами.

Для впровадження технології Industry 4.0 в структуру виробництва, засновану на елементах CPPS, пропонується згрупувати їх у 3 кластера:

– збір і обробка даних;

– M2M;

– HMI.

До кластеру збору і обробки даних віднесено об'єднання датчиків і виконавчих механізмів в мехатронні пристрії для взаємодії з фізичним світом. Це, у поєднанні з підходами технології хмарних обчислень, дає можливість застосовувати інтелектуальні об'єкти CPPS оснащені мікроелектронікою, датчиками, модулями зв'язку і обробки [123].

В результаті продукти, ресурси, машини і обладнання набувають форму базового інтелекту. IIoT створює середовище для з'єднання таких інтелектуальних об'єктів в єдиний інформаційний простір. Всі отримані дані технологічного процесу, на основі інтелектуальних об'єктів, будуть зберігатися на великих платформах даних в якості бази даних для аналітичних додатків. Аналітичні додатки, що залежать безпосередньо від

датчиків і виконавчих механізмів, генерують дані інтелектуальних пристроїв і інтелектуальних машин.

Ці дані дозволяють аналізувати величезну кількість статистичних даних про технологічні процеси, для визначення нестабільних параметрів і уникнення зниження якості в межах встановленого діапазону [124–128]. Традиційні дерева key performance indicator (KPI) використовуються для управління, контролю і вимірювання ефективності виробничого процесу на різних рівнях CPPS [129–131]. Збір і розрахунок значень традиційним способом займають багато часу, що повністю суперечить принципам Smart Manufacturung, де ключовим параметром є можливість обробки і прийняття рішень в режимі реального часу.

Дане протиріччя вирішується за допомогою використання різних засобів автоматизації (PLC, SCADA), в залежності від рівня, на якому вони застосовуються у вертикальній структурі CPPS.

Кластер M2M дає можливість забезпечити автоадаптивне управління взаємопов'язаними машинами та обладнанням без участі людини [132,133]. Дана концепція поєднує в собі підхід вертикальної і горизонтальної інтеграції.

Вертикальна інтеграція з'єднує машини і дані на різних рівнях та дозволяє створити нерозривний зв'язок даних машинних процесів на фізичному рівні з MES і ERP. Дані з ERP-системи містять інформацію про параметри виробничого процесу кожного окремого продукту. Вертикальна інтеграція дозволяє здійснювати індивідуальний потік без ручного перемикання.

Підхід горизонтальної інтеграції визначає глобальний зв'язок між обладнанням на одному рівні. На підставі цієї інформації виробничий процес може бути змінений автономно, відповідно до автоадаптивного плану виробництва [134].

Ця концепція демонструє рішення, які дозволяють аналізувати велику кількість даних з датчиків, у поєднанні з можливістю екстреного втручання людини в протікаючі технологічні процеси для їх корекції під необхідні виробничі параметри [135], за допомогою графічних інтерфейсів, що базуються на отриманні даних в реальному часі.

В [136] проводиться аналіз House of Lean Production (HPS), запропонованого Toyota Production System, який є символом принципів Lean Production (LP). Грунтуючись на структурі HPS і його параметрах, автори провели аналіз впливу концепції Industry 4.0 і CPPS на підвищення LP.

У таблиці 1.3 наведена матриця оцінки впливу Industry 4.0 і кластерів CPPS, описаних вище, на параметри LP. Код оцінки «+» означає, що дана технологія Industry 4.0 і CPPS може надати незначний позитивний вплив на цей принцип. Два показника «++» показують високу оцінку впливу, а три показника «+++» означають максимально можливий вплив технології на відповідний принцип LP.

Матриця оцінки впливу Industry 4.0 і кластерів CPPS на параметри LP показує, що впровадження CPPS на підприємствах підвищить економічність і дозволить домогтися максимально бережливого виробництва.

Таблиця 1.3 – Матриця оцінки впливу Industry 4.0 і кластерів CPPS на параметри LP [123,136]

Параметри Lean Production [137]	Кластери							
	Збір і обробка даних				M2M		HMI	
	Sensors and Actuators [138]	Cloud Computing [139, 140]	Big Data	Analytics	Vertical Integration [141]	Horizontal Integration [142]	Vertual reality	Augmented Reality [142]
5S	+	+	+	+	+	+	++	+++
Kaizen	+	++	+++	+++	+++	+++	+++	+++
Just-in-Time	++	++	+++	+++	+++	++	+	++
Jiboka	+	+++	+++	+++	++	++	+	+
Heijunka	++	++	+++	+++	+++	++	++	+
Standardisation	+	+++	+++	+++	++	++	+++	+++
Takt time	+	+	+++	+++	+++	+++	+	+
Pull flow	++	+	+	+	+++	+++	+	+
Man-machine separation	+	+	+	+	+	+	+++	+++
People and teamwork	+	+	+	+	+	+	+++	+++
Waste reduction	+	+	++	+++	+++	+++	+	+

1.4 Еталонна архітектура кібер-фізичних виробничих систем в моделях RAMI 4.0 (DIN SPEC 91345) і ISO-95

В рамках форуму ЄС «Оцифровка європейської промисловості» [83] запропонована еталонна архітектурна модель CPPS (RAMI 4.0). Вона являє собою тривимірну модель, яка описує простір Industry 4.0. Дана архітектурна модель представлена на рисунку 1.6.

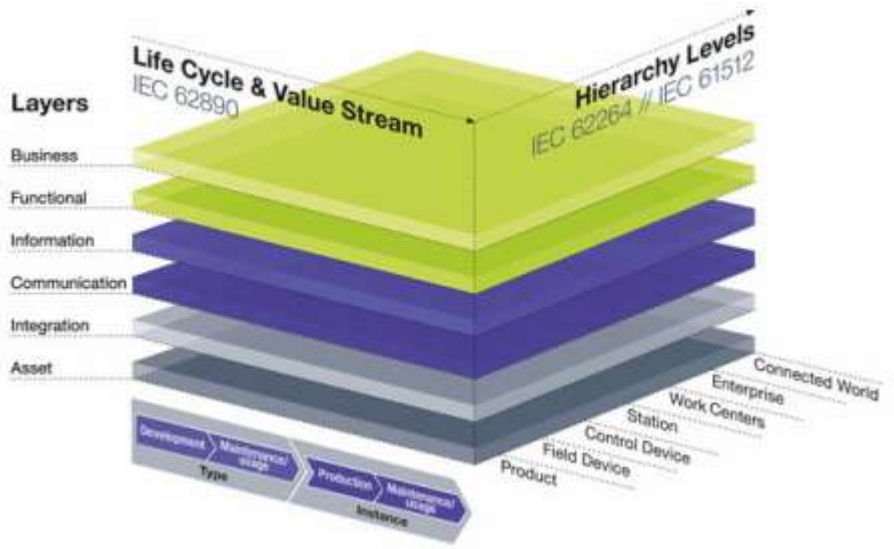


Рисунок 1.6 –Архітектурна модель Industry 4.0 (RAMI 4.0) [143,144]

Модель RAMI 4.0 складається з шести шарів по вертикальній і двох по горизонтальній осі. Перший шар по вертикалі має назву «Asset layer» і показує фізичні об'єкти: деталі, документи, архіви, діаграми, люди і т. д. [145].

На один рівень вище знаходиться «Integration Layer», на якому відбувається перетворення фізичних об'єктів в кібер об'єкти.

Компоненти «Asset layer» пов'язані на базі обробки інформації з кібер об'єктами на шарі «Integration Layer» і виконують роль сполучної ланки між фізичним і цифровим світом. Цей рівень включає комп'ютерний контроль процесу, системні драйвери, пристрої HMI, комутатори, концентратори і т. інш. [145,146].

Наступний рівень – «Communication layer» забезпечує стандартизований зв'язок між «Integration Layer» і «Information layer». Стандартизація досягається за допомогою єдиного формату даних, який використовується на «Information layer» та забезпечує управління «Integration layer».

«Information layer» зберігає дані в упорядкованному ієрархічному порядку і основною метою цього шару є надання інформації про обладнання та компоненти, які використовуються для створення виробу. «Information layer» заснований на програмному забезпеченні (форми додатків, даних, рисунків або файлів). У цьому шарі відбувається перетворення події у набір даних для більш високих рівнів.

Наступний шар на вертикальній осі «Functional layer» відповідає за виробничі правила, дії, обробку, систему, контроль та моніториг. Крім того він включає в себе інші види діяльності, такі як: координація компонентів виробничого процесу, включення / вимикання елементів системи, елементи управління. «Functional layer» реалізує концепцію віддаленого доступу і горизонтальну інтеграцію.

«Business layer» складається з бізнес-стратегії, бізнес-середовища, бізнес-цілей і управління взаємовідносинами, бюджетів і моделей ціноутворення [145,147].

На рис. 1.6 праворуч показана друга горизонтальна вісь, яка являє собою шар ієрархії, рівень якого заснований на міжнародних стандартах [148].

Крім чотирьох шарів: «Enterprise», «Work Centers», «Station» і «Control Device», існують два шари «Field Device» і «Product», які не включені до вище прелічених міжнародних стандартів, але є основними для процесів керування організаційно-технічними виробничими об'єктами на базі CPPS.

Шар «Field Device» дозволяє врахувати управління обладнанням або групою обладнання (при горизональній архітектурі) за допомогою інтелектуальних датчиків і виконавчих механізмів.

Шар «Product» забезпечує взаємозв'язок виробу і виробничих потужностей підприємства.

Верхній шар «Connected World» забезпечує можливість зв'язку із зовнішніми партнерами через сервісні мережі на базі технологій IoT [149–151].

Горизонтальна вісь (на лівій стороні рис. 1.6) показує життєвий цикл і значення потоків в процесі промислового виробництва (рис. 1.7). Вона складається з двох етапів: «Type» і «Instance».

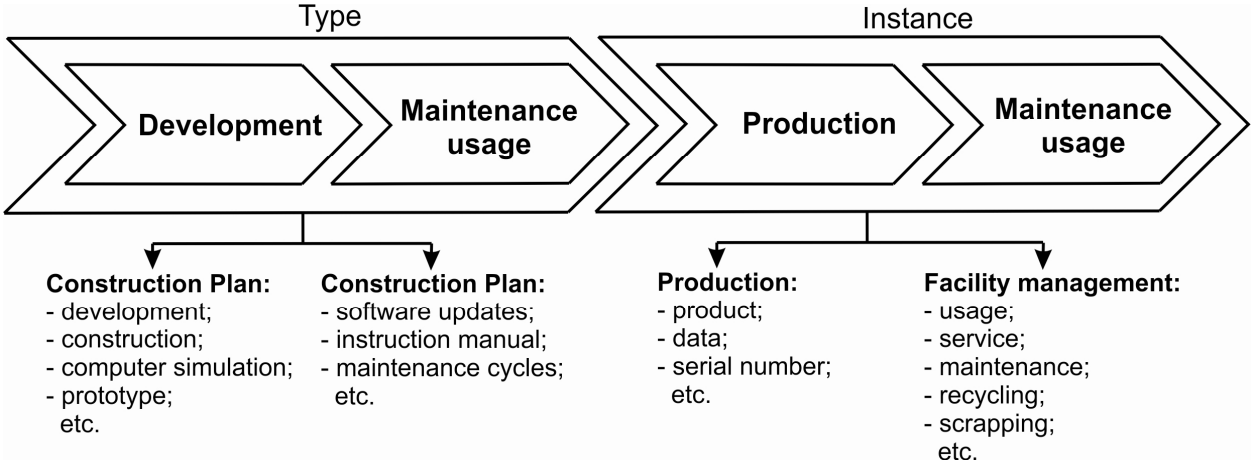


Рисунок 1.7 – Життєвий цикл і значення потоків в процесі промислового виробництва [151]

Коли виріб знаходиться на стадії розробки, він знаходиться в фазі «Type», а коли продукт знаходиться у виробництві – в фазі «Instance». Кожен раз, коли один і той же виріб переміщується в стадію розробки то він переміщується в фазу «Type» і цей цикл може повторюватись кілька разів [147,151].

На підставі проведеного аналізу еталонної моделі RAMI 4.0, яка включає в себе всі елементи вертикальної та горизонтальної архітектури, в рамках концепції Industry 4.0, можна виділити наступні недоліки:

– не існує точного визначення щодо того, яким чином окремі елементи всередині кожного шару з'єднані один з одним і з елементами розташованими на шарах вище і нижче даного;

– в еталонній архітектурній моделі RAMI 4.0 не описані шар «Field Device» і «Product» та не визначені елементи і типи зв'язків між ними, їх взаємодія з елементами на шарі «Control Device». Тому керування процесами організаційно-технічними виробничими об'єктами на базі CPPS, в рамках архітектурної моделі RAMI 4.0, має декларативний концепт, а не конкретні рішення у вигляді моделей, методів і послідовності керування процесами виробництва CPPS, в залежності від мети, вимог замовника, парку обладнання ділянок, цехів, підприємства і т.д.

В [152–156] проводиться аналіз існуючих архітектур Smart Manufacturung, в рамках концепції Industry 4.0. Автори досліджують базову архітектуру ISA-95 (S95), запропоновану Американським національним інститутом стандартів (ANSI), для спрощення процесу системної інтеграції підприємства в Smart Manufacturung. Міжнародний стандарт [85] визначає базову архітектуру ISA-95, як послідовність рівнів від 0 до 4.

Рівень 0 – це фізичний виробничий процес;

Рівень 1 – визначення і керування виробничим процесом за допомогою датчиків і виконавчих механізмів;

Рівень 2 – призначений для моніторингу, диспетчерського контролю та автоматичного управління виробничим процесом. Можливі об'єкти на цьому рівні: SCADA системи, розподілені системи керування (DCS) і PLC;

Рівень 3 – пов'язаний з робочим процесом і діяльністю з виготовлення виробу. Можливі об'єкти на цьому рівні: MES, система керування виробничою інформацією (PIMS), система керування складом (WMS) і комп'ютеризована система керування технічним обслуговуванням (CMMS).

Рівень 4 – пов'язаний з управлінням підприємством в цілому. Можливі об'єкти на цьому рівні: планування ресурсів підприємства (ERP), управління життєвим циклом продукту (PLM), управління людськими ресурсами (HRM), управління взаємовідносинами з клієнтами (CRM) і з постачальниками системи управління (SCM). У загальному вигляді архітектура ISA-95 (S95) представлена на рис. 1.8.

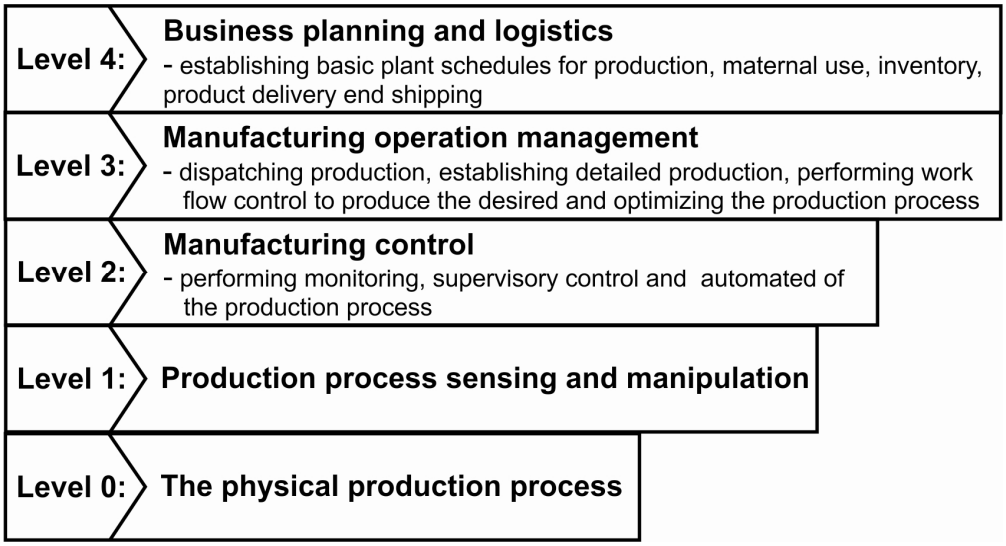


Рисунок 1.8 – Архітектура ISA-95 (S95) [157]

З появою поняття CPPS автори [158–159] запропонували п’ятирівневу архітектуру CPPS 5C, яка складається з рівнів: «Smart Connection», «Data-to-Information Conversion», «Cyber», «Cognition» і «Configuration». Загальний вигляд архітектури CPPS 5C представлений на рис. 1.9.

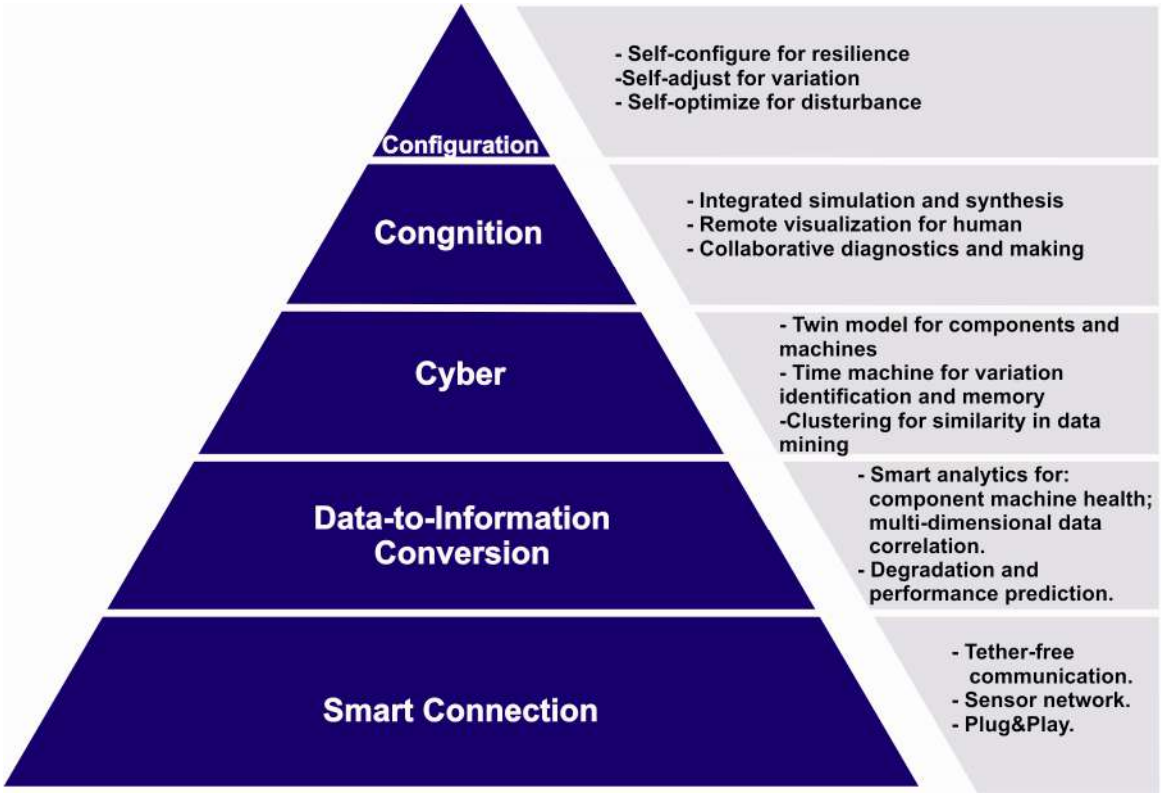


Рисунок 1.9 – Загальний вигляд архітектури CPPS 5C [157]

На першому рівні «Smart Connection» в [157] пропонується використовувати машини і їх компоненти для отримання точних і надійних даних на першому кроці кібер-фізичного керування. Датчики використовуються для збору різних параметрів протікання ТП виробництва виробів в режимі реального часу. Дані можуть також надходити від PLC, PAC, або виробничих систем ERP, MES, SCM і CMM. Технології IIoT використовуються для здійснення передачі даних і управління за обраними протоколами.

«Data-to-Information Conversion» призначений для перетворення отриманих даних в інформацію. В [158] розглядається цей рівень з позиції, що деякі пристрої можуть реалізовувати функції прогнозування і моніторингу зносу обладнання – це вносить поняття якості «інтелекту».

«Cyber» рівень представляє роль головного інформаційного центру, який збирає масив інформації з обладнання в промислову мережу. На цьому рівні додаткова аналітична інформація витягується із зібраної інформації. За отриманими даними продуктивність одного обладнання порівнюється і ранжується серед всіх однотипних типів обладнання, які знаходяться в промисловій мережі. Більш того аналітична інформація про роботу обладнання може бути застосована для прогнозування і планування виробництва, для досягнення необхідних заданих параметрів LP.

Рівень «Cognition» призначений для подання аналітичної інформації безпосередньо операторам для прийняття виробничих рішень. Цей рівень уможливлює дистанційну і спільну діагностику й прийняття рішень в умовах виробництва. Пріоритет завдань для процесу обслуговування може бути легко визначеним завдяки наявності порівняльної інформації та індивідуального стану групи або одиничного обладнання.

Рівень «Configuration» повертає зворотний зв'язок від «Cyber» рівня до «Smart Connection», тобто виконує диспетчерське управління, яке дозволяє зробити обладнання самоконфігурованим, самоналаштовуємим і самооптимізуємим. Рівень діє як система контролю стійкості (RCS), щоб

застосувати засоби управління для відповідних рішень на рівні контрольованого обладнання.

В роботі [157,158] розглядається архітектура CPPS 8C, яка утворюється шляхом додавання в архітектуру 5C граней 3C (рис. 1.10).

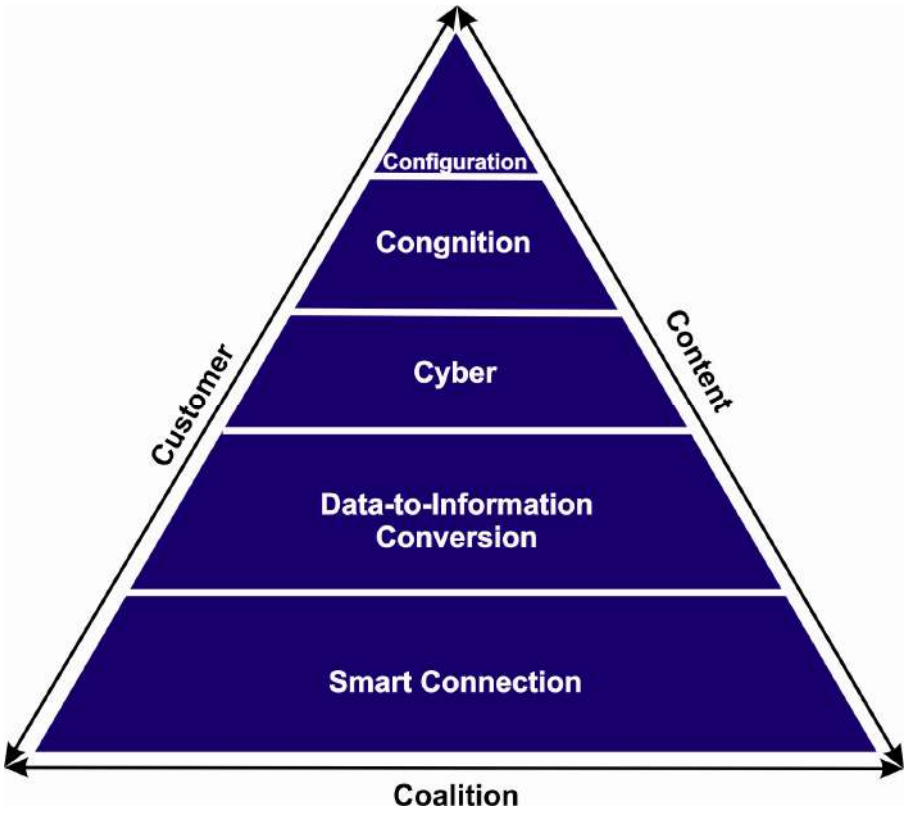


Рисунок 1.10 – Архітектура CPPS 8C запропонована в [157]

Аспекти запровадження граней 3C представлена в [157,158] як: інтеграція, клієнт і зміст. Автор підкреслює горизонтальну інтеграцію CPPS, як взаємозв'язок різних сторін і пов'язану з ними інформаційну складову (контент). Також грані дозволяють виділити найважливішу сторону, а саме участь клієнта в керування процесами виробництва. Проведемо аналіз кожної введеної грані 3C:

«Coalition» – грань фокусується на інтеграції ланцюжка створення вартості і інтеграції виробничого ланцюжка між різними сторонами, залученими у виробничий процес. Сторони можуть спільно побудувати ланцюжок поставок і планувати виробничі лінії для виробничих потоків.

Якщо є будь-які коригування у виробничому процесі, сторони можуть спільно реконструювати їх для досягнення максимального економічного ефекту.

«Customer» – грань орієнтована на роль, яку клієнти грають в процесі виробництва і післяпродажного обслуговування виробу. Smart Manufacturung можуть приймати різні замовлення в невеликих кількостях від різних клієнтів і виконувати замовлення вчасно. Клієнти або індивідуальний покупець, можуть брати участь при розробці і навіть у зміні технічних характеристик виробу в процесі виготовлення.

Це досягається за допомогою концепції виготовлення виробу. Тобто підприємство може автоматично підготувати матеріал, провести гнучке планування виробничого процесу, динамічно переналаштувати виробничі лінії і організацію зберігання і доставки виробу. Замовники можуть бути повідомлені про виробничі процеси, отримуючи звіти на електронну пошту.

Дані припущення, в аспекті клієнта, можуть відповісти на зрушення від звичайної «mass production» парадигми в парадигму «mass customization». Колишня парадигма призначалась для виробництва великої кількості продуктів однієї специфікації, в той час як нова – призначена для виробництва різноманітних товарів різних специфікацій. Крім того нова парадигма зачіпає доставку виробів замовнику, при цьому він може продовжувати отримувати післяпродажне обслуговування, інформацію про експлуатацію та використання виробу.

«Content» – грань спрямована на отримання, зберігання інформації про товари. Вся виробнича інформація (постачальники сировини, виробничі процеси, фізичні параметри навколишнього середовища, виробничі параметри) зберігаються в БД. Всі вироби контролюються на етапах післяпродажного обслуговування (технічне обслуговування, заміна деталей, усунення несправностей, утилізація, скарги, пропозиції та коментарі користувачів) вважаються важливими даними і зберігаються в базі даних (БД).

Положення в [157,158] для даної грані можуть допомогти досягти повного сервісного обслуговування виробу, аналізуючи всі отримані дані, а не тільки виробничий процес, що дозволяє замовнику та виробнику відстежувати тенденції ринку, проводити аналіз і прогнозувати його попит для кожного виробу.

За трьома додатковими гранями, архітектура 8C, запропонована в [157], орієнтована на вертикальну і горизонтальну інтеграцію, але це рішення приділяє велику увагу ЖЦ виробу на етапах сервісного обслуговування, що не спрощує розуміння і принципів рішення, з точки зору пославленних завдань в даній дисертаційній роботі.

Розглядаючи модель RAMI 4.0, в рамках першого і другого вимірів, необхідне чітке розуміння послідовності ієрархічних рівнів керування процесами на базі CPPS, а дана модель, в загальному вигляді, містить тільки аспекти та рекомендації. Моделі архітектури 5C, запропоновані в [158] і 8C, розроблені в [157], представляють загальну концепцію, парадигму та рекомендації до розробки, не пропонуючи послідовність процесів керування організаційно-технічними виробничими об'єктами на базі CPPS. При цьому не вказується і не обгрунтовується початкова точка (стартовий рівень) для створення кібер-фізичного керування процесами у організаційно-технічних виробничих об'єктах.

Грунтуючись на проведеному аналізі пропонується використовувати модель «Data, Information, Knowledge, Wisdom» (DIKW), яка дає можливість представити вертикальну інтеграцію керування процесами у CPPS.

У роботах [160–163] дана модель застосовується і широко використовується в різних сферах виробничої діяльності людини. На рис. 1.11 представлена інтерпретація моделі DIKW, для реалізації вертикальної інтеграцій ієрархічних рівнів CPPS, на базі моделей архітектур RAMI 4.0, 5C, ISA-95 (S95).

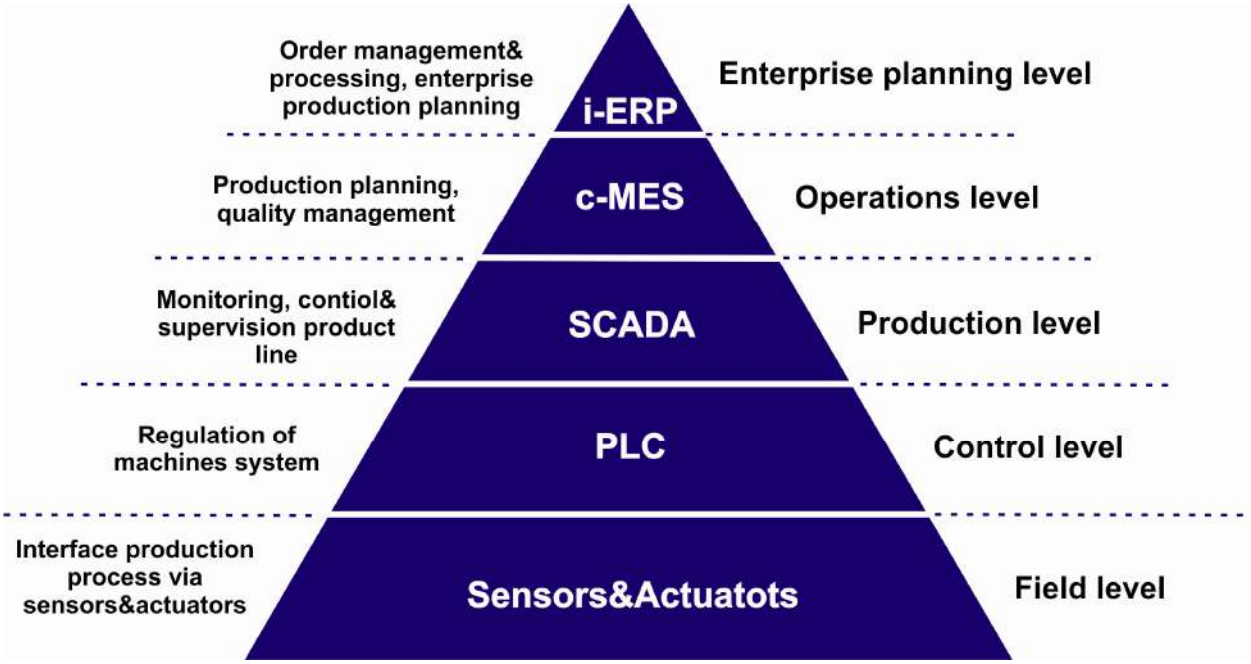


Рисунок 1.11 – Інтепретація моделі DIKW для вертикальної інтеграції ієрархічних рівнів CPPS

Грунтуючись на моделі DIKW інтеграція ієрархічних рівнів CPPS формується за принципом «знизу вверх». На першому польовому рівні (field level) представляється взаємодія з виробничим процесом за допомогою датчиків і виконавчих механізмів; на рівні управління (control level) здійснюватися регулювання і управління виконавчими механізмами з використання програмованих логічних контролерів (PLC); рівень технологічних ліній (production level) (фактично рівень виробничого процесу) виконує функцію контролю і моніторингу ТП виробництва; рівень операції (operations level) призначений для планування виробництва, управління якістю, і т.д.; рівень планування підприємства (enterprise level) забезпечує управління замовленнями та їх обробкою, загальне планування виробництвом і т.д.

Інтепретаційна модель DIKW дає можливість провести декомпозицію CPPS на ієрархічних рівнях за ключовими точками. Це може використовуватися як початкова базова конфігурація розробки моделей та методів кібер-фізичного керування на фізичному і кібернетичному рівні.

Грунтуючись на запропонованій в [157] 5C архітектурі CPPS, група вчених у складі: Petar Radanliev, David De Roure, Razvan Nicolescu, Michael Huth, запропонували імперичну архітектурну модель представлення інтеграції CPPS-IoE в архітектури 5C, фрагмент якої представлений на рис. 1.12.

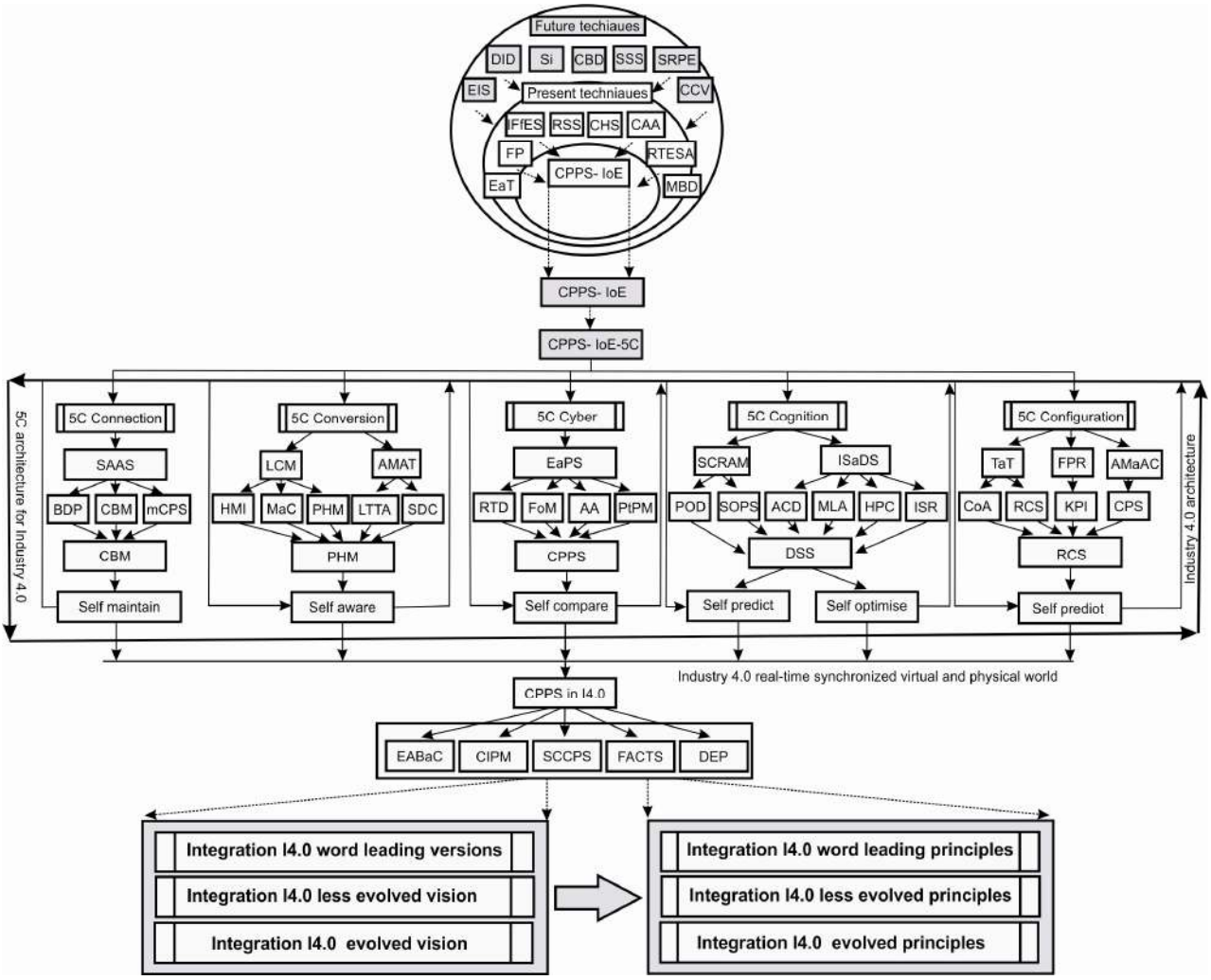


Рисунок 1.12 – Фрагмент імперичної архітектурної моделі інтеграції CPPS-IoE в архітектурі 5C для Industry 4.0 [164]

Автори стверджують, що запропонована модель дозволяє спростити процес інтеграції IIoT в I4.0, а в перспективі дозволить спростити процеси керування організаційно-технічними виробничими об'єктами на базі кібер-фізичних виробничих систем. Дана модель призначена для підтримки розробки нових національних стратегій Industry 4.0, що дасть можливість

поліпшити і переформулювати існуючі концепції і практичні ініціативи. Архітектурна модель також буде корисна для розробників в області IIoT, які прагнуть поліпшити і розвинути свою виробничу діяльність в концепції Industry 4.0.

При цьому автори звертають увагу, що архітектурна модель інтеграції CPPS-IoE-5C в Industry 4.0, вимагає подальшої перевірки та розмежування, шляхом застосування даної концепції при розробці реальних CPPS [164].

Проводячи аналіз запропонованої архітектурної моделі інтеграції CPPS-IoE-5C в Industry 4.0 можна виділити такі недоліки:

– дана архітектурна модель містить синтез усіх стратегій і теорій концепції Industry 4.0, які засновані на процесі каскадування і систематизації академічної літератури, при цьому запропонована модель не має чітко вираженого опису та подання рівнів і етапів кібер-фізичного керування, або не визначені ключові точки початку розробки CPPS і набір необхідних вхідних даних;

– модель містить рекомендаційний характер, в ній не прописані інформаційні зв’язки між елементами, змістом і параметрами, що обмежує її використання при розробці систем кібер-фізичного керування процесами у організаційно-технічних виробничих об'єктах [165].

1.5 Модель CMMI в розробці кібер-фізичних виробничих систем

Один з підходів до розробки кібер-фізичних систем керування організаційно-технічними виробничими об'єктами заснований на моделі Capability Maturity Model Integration (CMMI) [166]. Класичні методи враховують, в основному, окремі вироби або вже існуючі сімейства виробів. В них аналізується їх структура на рівні компонентів (фізичний рівень), для того щоб забезпечити підходи Industry 4.0 до розробки систем керування процесами на базі CPPS. Умовно їх можна розділити на дві групи:

– цілісні підходи (holistic approaches), які прагнуть оцінити і

використовувати максимально всі існуючі аспекти Industry 4.0 при розробці та впровадженні CPPS. В роботах [167–170] пропонується багатовимірна концептуальна модель CMMI у вигляді «Industry 4.0 toolbox», як центральний елемент для визначення відповідних областей застосування CPPS;

– специфічні підходи (specific approaches), де зосереджується увага на обмежену кількість аспектів, що стосуються Industry 4.0 і розглядається доцільність їх застосування у вирішенні задач кібер-фізичного керування.

Аналізуючи роботи [171–173] можна зробити висновок, що даний підхід показує очевидні обмеження, ізольованість фокусу і функціональних можливостей, в межах одного підприємства, нехтуючи взаємозалежностями. В результаті це призводить до зменшення потенціалу від розробки і впровадження CPPS, а також підвищує ризики некоректної розробки архітектури кібер-фізичного керування.

В [173] запропоновано нову модель зрілості, яка базується на наступних основних вимогах:

– операціоналізація абстрактних концепцій Industry 4.0 (кібер-фізична система, вертикальна / горизонтальна інтеграція), як практичних проблем застосування і оцінки цих концепцій, в рамках кібер-фізичного керування;

– перетворення результатів за термінами, якє включає висновки розробників за результатами оцінки впровадження, у зв'язку необхідністю задоволення вимог технічного завдання (ТЗ) на кожному рівні і етапі розробки CPPS.

Грунтуючись на вищенаведених вимогах, пропонуються наступні фази реалізації розробки і впровадження CPPS, на базі концептуальної моделі CMMI (рис. 1.12).

Як видно з рис.1.13 автор [166] визначає наступні фази:

– фази 1-3 фокусуються на розробці моделі зрілості CPPS (визначення сфери, дослідження структури і дизайну, оцінка обладнання та його структури, вимоги до виробу і його вихідні характеристики);

– фази 4-6 реалізують розробку остаточної моделі CPPS (визначення

процесу управління розробкою, знаходження шляхів, процедур і правил реалізації, розробки, тестування і моделювання отриманої моделі CPPS та її реалізація).

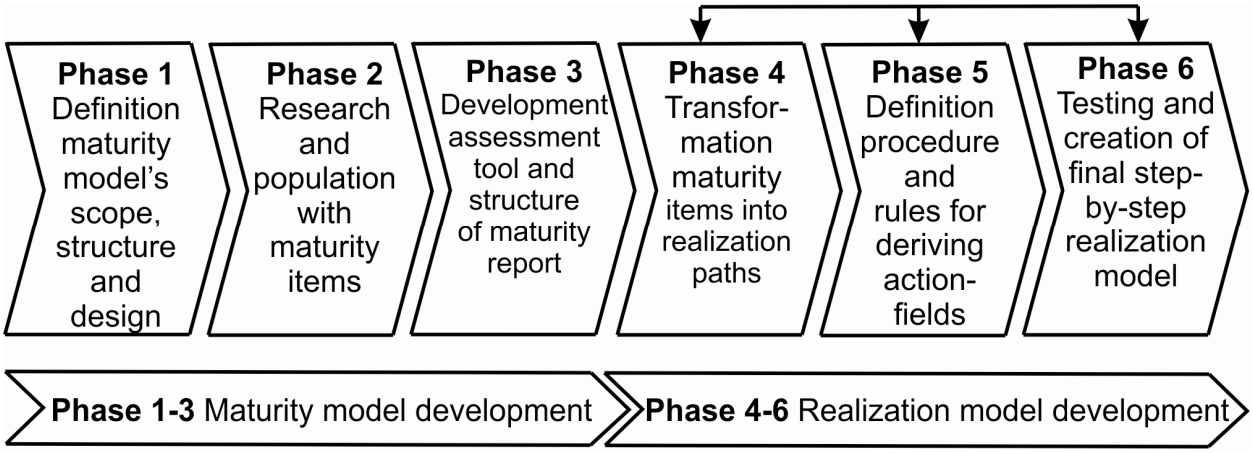


Рисунок 1.13 – Фази реалізації розробки і впровадження CPPS на базі концептуальної моделі CMMI [166]

На жаль модель, запропонована в [166,173], має узагальнений характер концептуальної моделі CMMI з точки зору застосування її для розробки кібер-фізичних систем керування. У свою чергу автори [174,175] запропонували розширення моделі CMMI, за допомогою розроблених рекомендацій щодо її поліпшення, відповідно до контекстуальних факторів:

– стратегічні мети;

– основні процеси;

– ключові показники ефективності.

В роботі [176] представлено дослідження реалізації CPPS реального виробництва, в ході якого розроблена всеосяжна описова модель подання відповідних систем, їх інтерфейсів, взаємозалежностей, параметрів. В ній автор приділив увагу необхідності визначення цілей системи (які включають технічні вимоги) на кожному рівні розробки CPPS керування процесами організаційно-технічних виробничих об'єктах. З точки зору даних досліджень, пропонується інтепретувати модель CMMI у вигляді послідовності декомпозиції головної мети розробки CPPS, як набору підцілей

і завдань для кожного ієрархічного рівня архітектури CPPS, які у сумі складових повинні задовольняти меті, зазначеній в ТЗ [177].

1.6 Дослідження сучасних життєвих циклів розробки програмного забезпечення для реалізації складних кібер-фізичних виробничих систем

В даному підрозділі проведено дослідження щодо можливості використання моделей ЖЦ розробки програмного забезпечення (ПЗ), з точки зору їх застосування для розробки адитивного кібер-дизайну кібернетичної складової CPPS.

Структура життєвого циклу ПЗ визначається і регламентується в міжнародному стандарті [178]. Відповідно до цього стандарту всі процеси моделі ЖЦ розробки ПЗ можна представити у вигляді етапів, показаних на рис. 1.14.

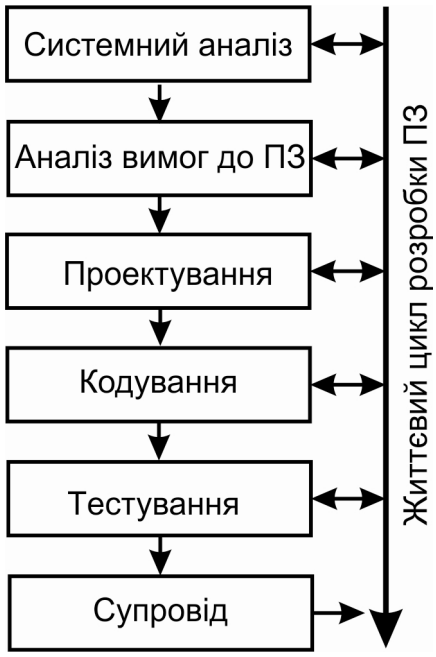


Рисунок 1.14 – Життєвий цикл розробки ПЗ

Наведемо коротку характеристику кожного етапу:

– системний аналіз – етап, в рамках якого йде визначення ролі кожного елемента, їх взаємодії при виконанні процесу планування проєкту, в ході

якого визначається обсяг роботи, ризик, необхідна трудомісткість і вартість;

– аналіз вимог до ПЗ – етап, на якому уточнюються і деталізуються функції, характеристики та створюється прототип інтерфейсу.

Необхідно відзначити, що запропоноване на початку розробки ПЗ технічне завдання, затверджене замовником і розробником, проходить стадію редагування і уточнення, що відповідно збільшує трудомісткість, вартість і складність проєкту. Це підтверджують аналітичні дослідження компанії Standish Group, яка проаналізувала роботу 364 американських корпорацій, а також підсумки виконання 23 тис. проєктів, пов'язаних з розробкою ПЗ і отримала наступні результати:

– тільки 16,2% проєктів завершені в строк і не перевищили розраховану трудомісткість і вартість, а всі функції, зазначені замовником, були реалізовані;

– 52,7% проєктів завершилися з запізненням, перевищивши при цьому запропоновану в ТЗ трудомісткість, з неповною реалізацією всіх функцій;

– 31,1% склали проєктів, які були закриті до завершення, так як вони не відповідали вимогам ТЗ і перевищили ліміт трудомісткості і вартість реалізації ПЗ.

В ході аналізу отриманих даних можна зробити висновок, що для проєктів, які завершилися з запізненням або були закриті до завершення, трудомісткість в середньому була перевищена на 89%, а термін виконання на 122% [179,180].

Основною причиною закриття проєктів, або перевищення їх терміну розробки, є нечітке і неповне формулювання вимог до програмного забезпечення, допущених на стадії створення ТЗ:

– розробка складається зі створення уявлень про розроблювану архітектуру, модульну структуру, алгоритмічну і функціональну структури ПЗ, структуру даних і розробку інтерфейсу. Однак, в реальних умовах, розробка HMI і реалізація основних візуальних компонентів (GUI) програмного забезпечення зазнає ряд змін, відповідно до побажань

замовника за інформативністю та взаємозв'язку робочих областей (робочих вікон, функціональних кнопок і т. д.) ПЗ, а також його специфіки та призначення;

– кодування полягає в перекладі результатів проєктування на мови високого рівня програмування;

– тестування – використання ПЗ для виявлення дефектів у функціях, логіці і в формі реалізації ПЗ;

– супровід – це підтримка у використанні ПЗ, а також зміни в ході виявлення помилок, вдосконалення за вимогами замовника або адаптація ПЗ в залежності від зміни операційних систем (ОС) і т. д.

Досліджуючи можливість застосування моделі ЖЦ розробки ПЗ для рішення завдань розробки адитивного кібер-дизайна кібернетичної складової CPPS дозволяє зробити наступні висновки:

– етапи системного аналізу та аналізу вимог ЖЦ розробки ПЗ не мають сенсу і не можливі до застосування тому, що системний аналіз і аналіз вимог до майбутнього адитивного кібер-дизайна кібернетичної складової CPPS, визначаються на етапі розробки фізичної складової і залежать від технічних характеристик обладнання, датчиків, виконавчих механізмів, а також вимог до технологічного процесу, що контролюється;

– етап проектування вже інтегрований в кібернетичну складову CPPS, яка має «жорстко» виражену ієрархічну архітектуру, прив'язану до процесів керування організаційно-технічними виробничими об'єктами (послідовність рівнів і етапів, об'єднаних логічною послідовністю, зв'язками і описами алгоритмів функціонування), який обирається в процесі розробки адитивного кібер-дизайна кібернетичної складової CPPS.

Найбільшого поширення набули такі моделі ЖЦ розробки ПЗ, які запропоновані в стандарті ISO / IEC 12207:2017 [178]:

– waterfall model (каскадна модель);

– v-shaped model (V-образна модель);

– prototype model (модель прототипування);

– rapid application development model або RAD-model (модель швидкої розробки додатків);

– incremental model (багатопрохідна модель);

– spiral model (спіральна модель).

Проведемо аналіз існуючих моделей ЖЦ розробки ПЗ для визначення можливості їх застосування в розробці адитивного кібер-дизайна кібернетичної складової CPPS.

Waterfall model (каскадна модель) – застосовувалась при розробці однорідних інформаційних систем в 1970-1980 роках, коли ПЗ являло собою єдине ціле. Основні етапи каскадної моделі представлені на рис. 1.15.

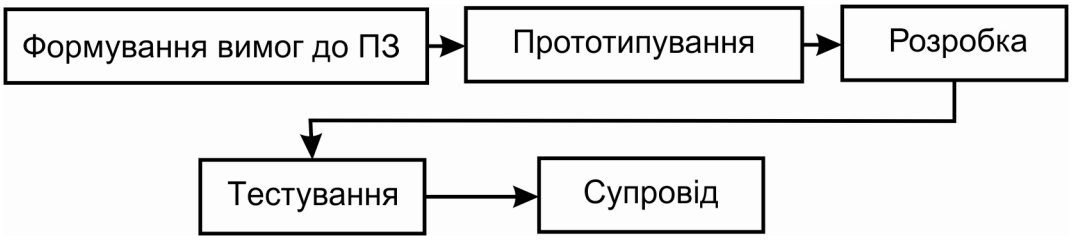


Рисунок 1.15 – Основні етапи каскадної моделі

Принципова особливість каскадної моделі полягає в тому, що перехід між етапами здійснюється тільки після повного завершення робіт на попередньому етапі моделі. Це пов'язано з тим, що результати, отримані при виконанні етапу, є вихідними даними для наступного етапу. При цьому вимоги, що пред'являються в ТЗ до ПЗ, суворо документуються і фіксуються на весь час виконання розробки.

Дана модель добре зарекомендувала себе при розробці ПЗ, для якого досить точно або повністю сформульовано ТЗ, а, отже, можна точно розрахувати трудомісткість і вартість розробки. Однак застосування даної моделі при розробці адитивного кібер-дизайна кібернетичної складової CPPS, неможливе тому, що реальний процес розробки ніколи не вкладається в жорстку схему каскадної моделі, результати чергового етапу часто викликають зміни в прийнятих рішеннях, розроблених на попередніх етапах.

Це змушує постійно повертатися до попередніх етапів і вносити зміни і уточнення в раніше прийняті рішення [181].

V-shapetd model (V-образна модель) – заснована на систематичному підході до розробки ПЗ і визначає чотири базових етапи: аналіз, проектування, розробка і огляд. Дана модель є різновидом каскадної, в якій приділяється велика увага верифікації та атестації ПЗ. Структура V-образної моделі представлена на рис. 1.16.

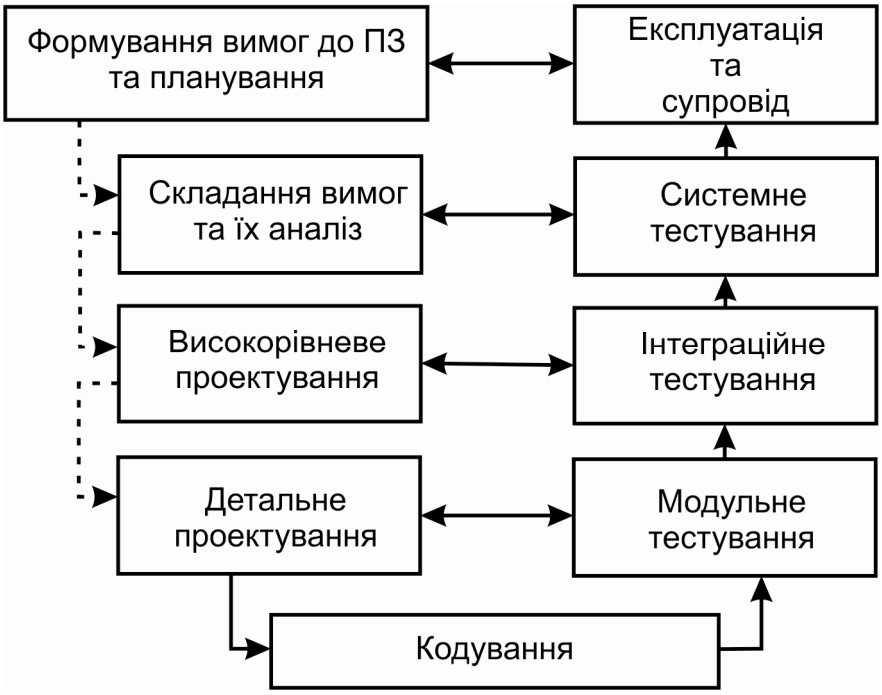


Рисунок 1.16 – Структура V-образної моделі

Штрихпунктирні стрілки (рис. 1.16) вказують на виконання даних етапів паралельно основним (системні вимоги до ПЗ і планування). Вони об'єднані з такими етапами: фомування вимог до продукту і їх аналіз (складання повного технічного завдання); високорівневе проектування (визначається структура програмного забезпечення і внутрішні зв'язки); детальне проєктування (визначається алгоритм роботи кожного компонента). При такому підході приділяється велика увага верифікації та атестації ПЗ з ранніх етапів проектування, що дозволяє легко відстежувати хід роботи, а також виконання кожного етапу, так як завершення кожного етапу є

контрольною точкою.

Однак основними недоліками даної моделі, з точки зору застосування її при розробці адитивного кібер-дизайнуа кібернетичної складової CPPS, є: до уваги береться інтеграція між етапами; немає можливості внесення змін на різних етапах ЖЦ; тестування вимог відбувається занадто пізно, а отже, і внесення змін впливає на графік виконання робіт [180].

Prototype model (модель прототипування) – дозволяє створити прототип ПЗ до або протягом етапу складання ТЗ. Потенційний замовник працює з прототипом, визначає його сильні і слабкі сторони, а результати повідомляє розробнику [178]. В результаті забезпечується взаємний зв'язок між розробником і замовником, який використовується для змін і коригування прототипу. Модель прототипування представлена на рис. 1.17.

Цей процес триває до тих пір, поки замовник не буде задоволений ступенем відповідності прототипу вимогам до ПЗ в ТЗ. Однак, крім зазначених переваг, при розробці адитивного кібер-дизайна кібернетичної складової CPPS, наприклад, етапи рішення складних технічних, технологічних і специфічних завдань, відсуваються на останній етап, що неприйнятно при кібер-фізичному керування, тому що вони є первинними функціями, які необхідні замовнику (замовник може віддати перевагу отриманню прототипу, а не закінченій повній версії ПЗ).

Розробка прототипу може невиправдано затягнутися у зв'язку з невизначеністю кількості виконаних ітерацій під час розробки, що збільшує трудомісткість адитивного кібер-дизайна кібернетичної складової CPPS [180,181].

Rapid Application Development model (RAD-model) – на відміну від попередніх моделей, в RAD-model замовник відіграє вирішальну роль, в тісній взаємодії з розробниками ПЗ він бере участь у формуванні технічного завдання та апробації прототипу.

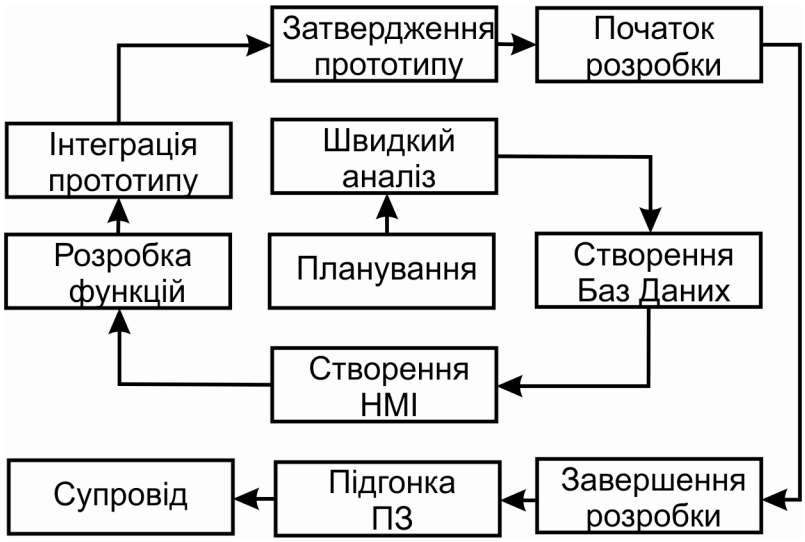


Рисунок 1.17 – Модель прототипування

Модель має ряд переваг: використання сучасних інструментальних засобів дозволяє скоротити трудомісткість розробки; залучення замовника на етапах складання вимог і розробки інтерфейсу користувача, що зводить до мінімуму ризик отримання програмного продукту, яким залишиться незадоволений замовник; повторне використання компонентів вже існуючих програм. Основні етапи RAD-model показані на рис. 1.18.

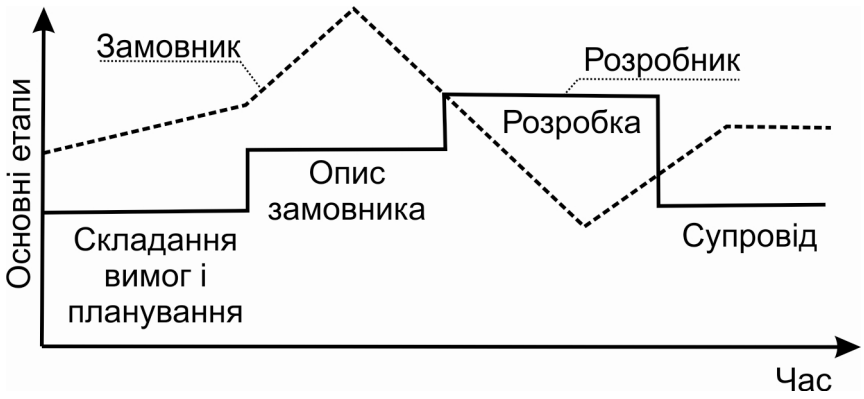


Рисунок 1.18 – Основні етапи RAD-model

Однак, дана модель має ряд недоліків, пов'язаних з розробкою адитивного кібер-дизайна кібернетичної складової CPPS: в залежності від специфіки ПЗ, використання компонентів існуючих програм неможливо або зведено до мінімуму; існує великий ризик, що розробка кібернетичної

складової CPPS ніколи не буде закінчена, у зв'язку з зацикленням робіт з її розробки.

Incremental model (багатопрохідна модель) – складається з об'єднання процесу побудови ПЗ, з додаванням на кожній інтеграції нових функціональних можливостей або підвищенням ефективності ПЗ. Дана модель передбачає, що на початковому етапі ЖЦ розробки виконується конструювання ПЗ в цілому і визначається число інкрементів та належних до них функцій. Далі кожен інкремент проходить через етапи ЖЦ, що залишилися (кодування і тестування) [178]. Багатопрохідна модель представлена на рис. 1.19.

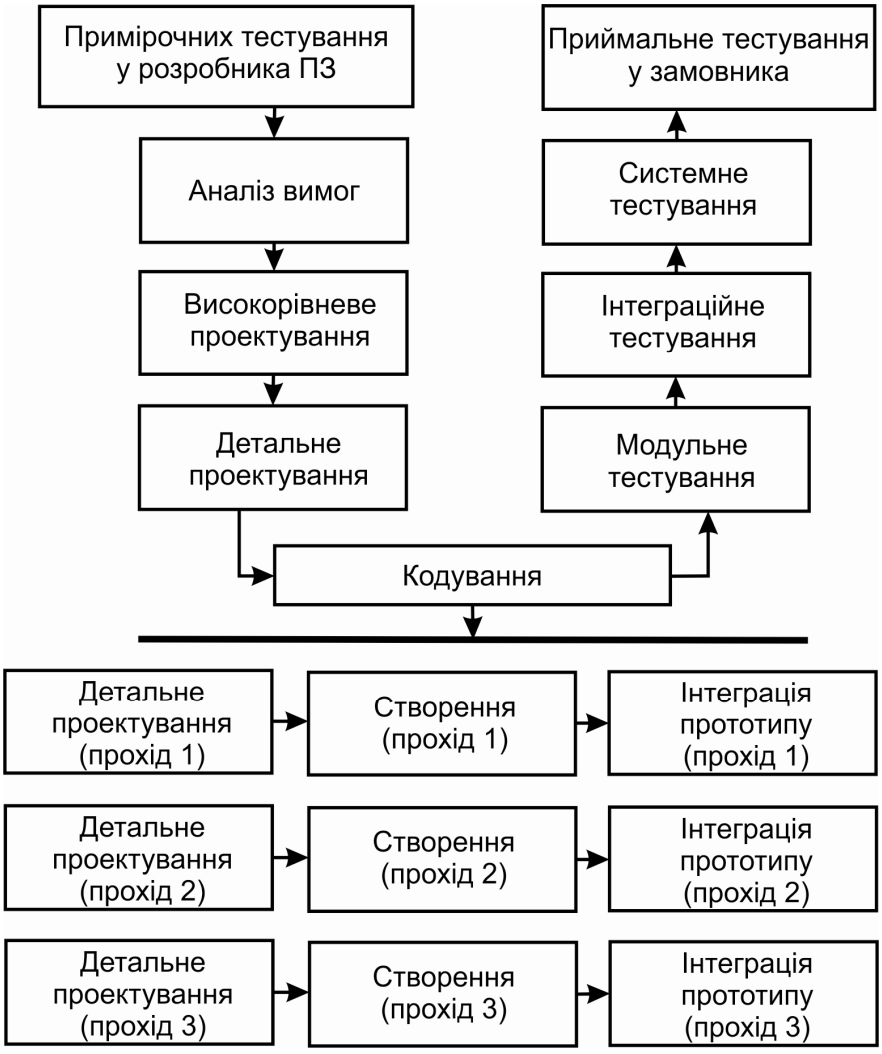


Рисунок 1.19 – Багатопрохідна модель

Однак, дана модель має такі недоліки: непередбачені етапи інтеграції всередині кожного інкремента; повна функціональність ПЗ повинна бути вказана на початку ЖЦ в технічному завданні; є можливість тенденції відтягування рішень складних завдань на завершальному етапі; загальні витрати трудомісткості і вартості ПЗ не знижуються у порівнянні з іншими моделями; обов'язковою умовою даної моделі є наявність гарного планування і проектування програмного забезпечення.

В результаті наведених вище недоліків дану модель рекомендують використовувати при проектуванні ПЗ за умови заздалегідь сформульованих вимог і виділення великого періоду часу.

Spiral model (спіральна модель) – особливість даної моделі полягає в тому, що прикладне ПЗ створюється не відразу, а частинами (модулями) з використанням методу прототипування. У даній моделі прототип – чинне ПЗ, що реалізує окремі функції і зовнішній інтерфейс користувача. Створення прототипу здійснюється за декілька ітерацій (витків спіралі), кожна ітерація відповідає створенню фрагмента або версії програмного забезпечення, на якій уточнюються мета і характеристики, оцінюється якість отриманих результатів, а також проводиться ретельний аналіз ризику перевищення трудомісткості і вартості ПЗ. Розробка ПЗ ітераціями відображає об'єктивно існуючий спіральний цикл, дозволяючи переходити на наступну стадію, не чекаючи повного завершення робіт на поточній стадії, оскільки при ітеративному способі розробки відсутні роботи можна зробити на наступній ітерації. Основні етапи спіральної моделі представлені на рис. 1.20.

До недоліків даної моделі можна віднести наступні: план роботи складається на основі статистичних даних, отриманих в попередніх проектах і з особистого досвіду розробника. Спіральна модель може тривати нескінченно, в результаті пропозицій замовника, які можуть породити нову спіраль, а, отже, збільшити трудомісткість, вартість і час розробки ПЗ, у зв’язку з чим дана модель не підходить для розробки адитивного кібер-дизайну кібернетичної складової CPPS.

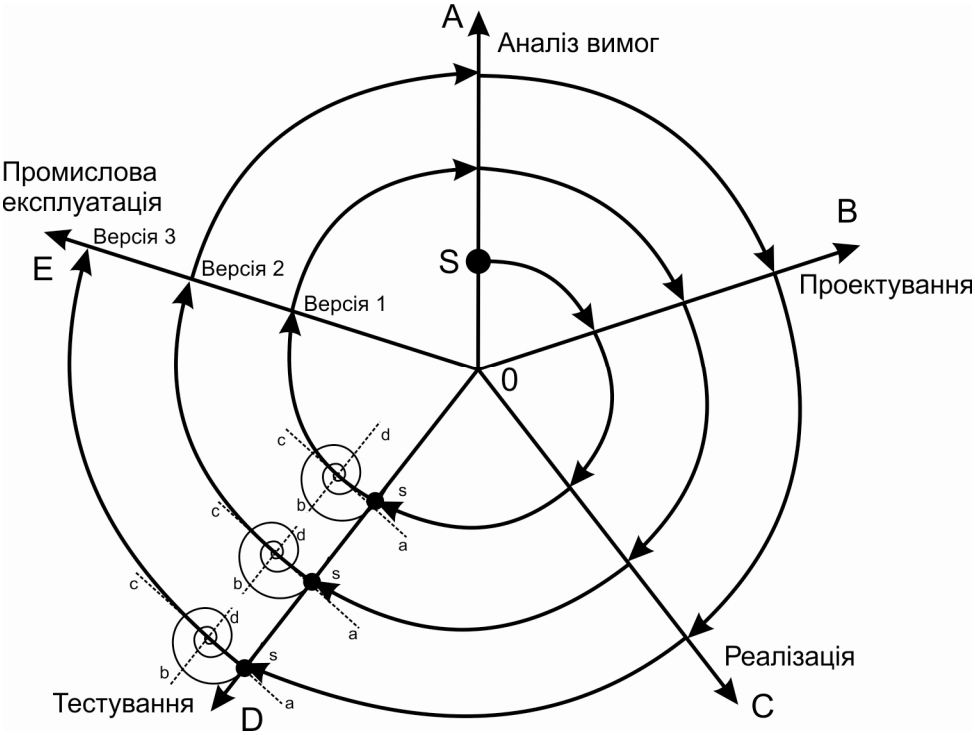


Рисунок 1.20 – Спіральна модель

1.7 Дослідження методологій розробки інформаційних систем для адитивного кібер-дизайна кібернетичної складової CPPS

Розглядаючи CPPS як складну взаємопов'язану багаторівневу архітектуру, яка об'єднує в собі кібернетичні і фізичні властивості та враховуючи специфіку процесів керування складними організаційно-технічними виробничими об'єктами на базі CPPS, можна визначити що доменантною для розробки адитивного кібер-дизайна кібернетичної складової CPPS, є інформаційна база, яка отримується з фізичного рівня ТП виробництва, а на верхньому рівні керування процесами, на відміну від класичних, має синтезовану форму, яку представлено на рис. 1.21.

Розробка кібернетичної складової CPPS, являє собою великий набір елементів, який складається з: програмних скриптів PLC, що обробляють дані на PLC з датчиків; програмних модулів керування на кожному етапі; HMI для візуалізації технологічної інформації, а також програмних рішень

для рівнів MES і ERP, пов'язаних в єдине цифрове середовище на базі сучасних інформаційних технологій. Тому, в рамках даного дослідження, необхідно провести аналіз існуючих методологій проектування інформаційних систем, з точки зору їх можливого часткового або повного застосування, при рішенні задач розробки адитивного кібер-дизайну кібернетичної складової CPPS.

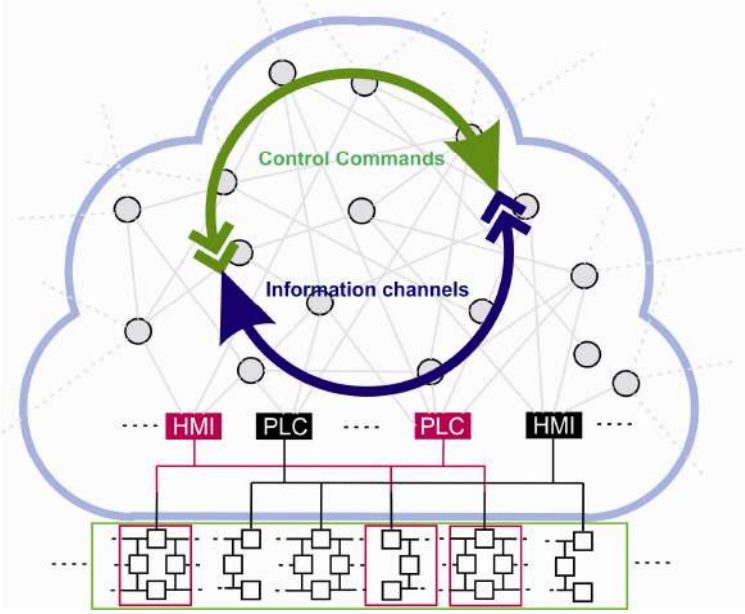


Рисунок 1.21 – Представлення кібернетичної складової CPPS

Для проведення аналізу визначимо наступні поширені методології проектування інформаційних систем:

- методологія функціонального моделювання SADT;
- методолгія RED;
- методологія RUP.

Проведемо дослідження кожної методології і можливість їх застосування для вирішення завдань розробки адитивного кібер-дизайна кібернетичної складової CPPS.

Методологія SADT (Structured Analysis and Design Technique) – методологія структурного аналізу і проєктування, запропонована Дугласом Т. Россом, яка базується на структурному аналізі систем і графічному

поданні організації у вигляді системи функцій, розділених на три класи структурних моделей:

– функціональна модель;

– інформаційна модель;

– динамічна модель.

Підхід до процесу моделювання заснований на наступних етапах:

– аналіз та збір інформації про предметну область;

– документування отриманої інформації;

– моделювання (IDEF0);

– коректура моделі в процесі інтерактивного рецензування.

Методологія заснована на формалізації процесу моделювання і базується на стадіях:

– аналіз;

– проєктування;

– реалізація;

– об'єднання;

– тестування;

– установки і функціонування.

Проектування на основі IDEF0 (Function Modeling) зводитися до декомпозиції основних функцій організації, а результатом є розробка ієрархічної моделі проектованої інформаційної системи при багатократній рівневій декомпозиції, для отримання чіткого і детального опису всіх процесів;

IDEF1 (Information Modeling) – методологія моделювання інформаційних потоків усередині системи, що дозволяє візуалізувати і аналізувати їх структуру і взаємозв'язки;

IDEF1X (Information Modeling Extended) – методологія розроблена на реляційних структурах і застосовується, в основному, для проектування БД і відноситься до типу ER (Entity – Relationship) «сутність - взаємозв'язок» ;

IDEF2 (Simulation Model Design) – методологія динамічного

моделювання систем. На сучасному етапі представлені алгоритмами і їх комп'ютерною реалізацією у вигляді бібліотек, які дають можливість конвертувати статичні діаграми IDEF0 в динамічні, за допомогою CPN (Color Petri Notes);

IDEF3 (Process Description Capture) – методологія документування процесів всередині системи, на базі сценаріїв і послідовності операцій для кожного процесу. Дана методологія напряму пов'язана з методологією IDEF0, у вигляді функціональних блоків кожного процесу засобами IDEF3;

IDEF4 (Object-Oriented Design) – методологія побудови об'єктно-орієнтованих систем, що дає можливість відображати структуру об'єктів і закладених в них принципів взаємодії та дозволяє аналізувати і оптимізувати складні об'єктно-орієнтовані системи;

IDEF5 (Ontology Description Capture) – методологія онтологічного дослідження систем, за допомогою певного словника термінів і правил, на основі яких формуються достовірні твердження про стани системи в деякий момент часу. На базі даних тверджень формуються висновки про подальший розвиток системи та її опітімізацію;

IDEF6 (Design Rationale Capture) – методологія проектних рішень, основним завданням якої є спрощення отримання «знань про здатність моделювання», їх уявлення і використання при розробці системи управління для Smart Manufacturung;

IDEF8 (User Interface Modeling) – метод розробки інтерфейсів користувача. Визначає увагу розробників на інтерфейс та поведінку користувача по:

- виконуваній операції;

- сценарію взаємодії, що визначається специфічною роллю користувача;

- деталям інтерфейсу, які визначають елементи управління для виконання операції та управління потоками даних.

IDEF9 (Business Constraint Discovery method) – метод дослідження і

аналізу бізнес обмежень, в яких працює Smart Manufacturung, їх характеристики і вплив на процес моделювання.

IDEF10 (Implementation Architecture Modeling), IDEF11 (Information Artifact Modeling), IDEF12 (Organization Modeling), IDEF13 (hree Schema Mapping Design) – ці методи були визначені як затребувані, але не були розроблені;

IDEF14 (Network Design) – метод проектування комп'ютерних мереж, який базується на аналізі вимог мережевих компонентів і забезпечує підтримку вирішення пов'язаних з раціональним управлінням матеріальними рісурсами.

Методологія SADT у більшості випадків підходить для опису моделей верхнього рівня, а наявність жорстких вимог забезпечує розробку моделей систем стандартного виду. При даних перевагах методологія SADT має ряд недоліків: складність сприйняття (велика кількість зв'язків (дуг) на діаграмі); велика кількість рівнів декомпозиції, що призводить до труднощів зв’язку декількох процесів, представлених в різних моделях, в рамках одного Smart Manufacturung. [182]

На базі методології SADT (IDEF0 + DFD) розроблена уніфікована мова моделювання UML, яка дає можливість реалізації специфікації, візуалізації, конструювання та документації складних інформаційно-насичених об'єктних систем. При всіх заявленим можливостям мови UML, вона має ряд недоліків, що не дозволяють її застосувати як базову методологію для розробки адитивного кібер-дизайну кібернетичної складової CPPS:

– надмірність мови обумовлена тим, що вона включає багато практично не використовуваних діаграм і конструкцій;

– нечітка семантика, обумовлена тим, що UML визначена комбінацією себе за допомогою абстрактного синтаксису, мовою опису обмежень (OCL) і Англійською мовою (детальна семантика). У деяких випадках абстрактний синтаксис комбінації UML, OCL і англійської мови суперечать один одному, в інших випадках вони неповні. Неточність опису UML однаково

відображаються на користувачах і розробниках інструментів, приводячи до несумісності розроблюваних інструментів під мову UML через унікальне трактування специфікації;

– UML розроблялася як мова моделювання загального призначення, в спробі досягти сумісnрсті з усіма можливими мовами розробки.

В контексті даних досліджень, для досягнення процесів керування організаційно-технічними виробничими об'єктами, повинні бути обрані та застосовні можливості UML, що накладає обмеження на її застосування, тому що на даний момент не існує повністю сформульованої формалізації опису моделей та методів розробки адитивного кібер-дизайну кібернетичної складової CPPS.

Тому застосування методології SADT, а саме мова моделювання UML, для розробки адитивного кібер-дизайну, неможливе без існуючих математичних моделей і методів, а також структури, яка формалізує всі рівні, етапи і послідовність розробки складних систем кібер-фізичного керування. Отже на даний момент часу вона носить описовий характер з нечіткою семантикою, що не припустимо у моделях та методах керування процесами в складних організаційно-технічних виробничих об'єктах.[182]

Методологія RED (Rapid Application Development) базується на принципах інкреметного прототипування, що дозволяє на ранній стадії проєктування інформаційної системи продемонструвати замовнику діючу інтерактивну модель прототипу, уточнити прийняті проєктні рішення та оцінити експлуатаційні характеристики. У даній методології швидкість розробки досягається за рахунок використання компонентно-орієнтованого конструювання і має сенс при:

– обмеженому бюджеті розробки;

– нечітко визначених вимогах до системи, що розробляється;

– мінімальному терміні розробки;

– декомпозиції проєкту на складові елементи за функціональним призначенням.

В процесі розробки за методологією RAD система проходить чотири фази:

– фаза аналізу і планування, в ході яких визначаються вимоги, функції додатку і його пріоритети, а також інформаційні потреби, вказані замовником, при безпосередній участі розробника. Визначаються рамки маштабу проєкту, тимчасові терміни і платформи для запуску системи;

– фаза проєктування проходить за участі потенційних користувачів системи під керівництвом розробників. Групи і підгрупи RAD на даному етапі використовують комбінацію технік спільної розробки додатків JAD і CASE (інструментів для реалізації вимог користувачів в моделях, щодо розробляються). Це дає можливість користувачам зрозуміти, модифікувати і визначити робочу модель системи, яка повинна відповідати вимогам ТЗ і при необхідності, створити частковий прототип системи, що розробляється.

В результаті даної фази розробник проєктує:

– загальну інформаційну модель системи;

– функціональні моделі системи і підсистеми;

– робочі прототипи інтерфейсу користувача, звітів та діалогових вікон. Особливістю даної фази в RAD моделі є те, що кожен прототип стає частиною розроблюваної системи;

– фаза побудови спрямована на швидку розробку системи за результатами отриманими на попередній фазі. При цьому замовник продовжує брати участь в розвитку системи і при необхідності пропонує внесення змін і поліпшень в систему, що розробляється. Особливістю даного етапу є те, що тестування проходить синхронно під час розробки;

– фаза впровадження дозволяє синхронізувати три ключові моменти: навчання користувачів розробленої системи; тестування функціональності на відповідність вимогам ТЗ; заміна старої системи на вдосконалену нову.

Розглядаючи методологію RAD, в контексті даних досліджень, і можливість її застосування при розробці адитивного кібер-дизайну кібернетичної складової CPPS можна виділити наступні недоліки:

– відсутність маштабування (аналіз показав, що методологія RAD переважно використовується для швидкої розробки невеликих і малих систем, невеликими і середніми проєктними командами);

– фокусування методології RAD на прототипах в, більшості випадків, призводить до проблеми постійного внесення дрібних змін в окремі елементи системи і при цьому ігноруються проблеми системної архітектури, що є недопустимим для процесів керування організаційно-технічними виробничими об'єктами;

– в реальних проєктах неможливо дотримати баланс між гнучкістю і контролем процесу розробки, а в більшості випадків обирається один з них. В останньому випадку методологія RAD не буде життєздатною;

– методологія RAD не підійде для систем в яких: основними вимогами є якість і контроль всіх етапів розробки; створення крупномаштабних проектів (час реалізації яких більше 90 днів); критично важливим є високий рівень планування і жорстка архітектура; притаманне суворе дотримання розробленим протоколам і інтефейсу.

Грунтуючись на вище перечелічених недоліках методології RAD, можна зробити висновок, що її застосування не доцільне при вирішенні задач розробки адитивного кібер-дизайну кібернетичної складової CPPS [182].

Методологія RUP (Rational Unified Process) – заснована на інтератівній моделі розробки систем на базі ЖЦ «Спіральної моделі». В кінці кожної ітерації розробники повинні досягти заплановані мети, створити або доопрацювати «проєктні артефакти» і отримати проміжну, але функціональну версію кінцевої системи. Основними перевагами методології RUP є швидкість реагування на мінливі вимоги до розроблюваної системи, виявлення та усунення ризиків на ранніх стадіях проєкту, а також ефективність контролю якості створюваного продукту.

ЖЦ розробки системи складається з чотирьох основних фаз:

– Inception (початкова фаза), в ході якої формується початкове бачення і межі проекту; розроблюється економічне обґрунтування; визначаються

основні вимоги, обмеження і ключові функції продукту; формується базова версія моделі прецедентів і проводитися оцінка ризиків. Як результат цієї фази розробник оцінює досягнення етапу життєвого циклу мети (LOM) і узгодженності між замовником і розробником;

– Elaboration (уточнення), на цій фазі проводяться дослідження, аналіз предметної області та побудова архітектури системи, яка включає детальний опис прецедентів, на базі яких буде проведено тестування розробленої архітектури і уточнюються терміни та вартість розробки системи. Успішне виконання фази Elaboration означає досягнення етапу життєвого циклу архітектури (LAM);

– Construction (побудова), на даній фазі ЖЦ розробки здійснюється реалізація частини функцій системи і завершується розробка першої бета-версії системи;

– Transutuon (впровадження), фаза, в ході виконання якої створюється фінальна версія системи і передається від розробника до замовника для повного тестування та оцінки якості. Якщо якість не відповідає вимогам і критеріям ТЗ, встановленим на фазі Inception, то дана фаза повторяється знову поки всі умови зазначені замовником в ТЗ не будуть задоволені.

Грунтуючись на вищевказаному можна виділити основні риси методології RUP:

– інтерактивний процес розробки (controlled interactive);

– наскрізне застосування апарату Use Case Driven;

– приділення особливої уваги розробці Architecture Centric;

– управління вимогами та змінами (Requirements Configuration and Change Management);

– використання концепції Component Based Development;

– базування на принципах Visual Modeling Nechniques.

Аналізуючи [183] можна виділити наступні загальні ризики при використанні методології RUP:

– планування нереалістичних термінів за якими неможливо виконати

повний обсяг роботи, а отже профіцит бюджету;

– неповне враховування функціональності, що неприпустимо і критично при розробці адитивного кібер-дизайну кібернетичної складової CPPS на базі HMI та GUI, так як функціональність є головним пріоритетом для процесів керування організаційно-технічними виробничими об'єктами;

– розробка некоректного візуального інтерфейсу користувача, що не дозволить реалізувати синтез інформаційних потоків та потоків керування вертикальної інтеграції ієрархічних рівнів кібер-фізичного керування;

– відведення великої кількості часу на оптимізацію;

– базування методології на «Спіральній моделі» ЖЦ, яка вносить на кожній ітерації новий потік змін, що в рамках розробки адитивного кібер-дизайну кібернетичної складової CPPS недопустимо і призводить до порушення архітектури, алгоритмів функціонування, а отже ускладнює процес управління розробкою CPPS;

– брак інформації про зовнішні GUI компоненти, які визначають оточення системи, що призводить до зменшення продуктивності, а це суперечить основним принципам кібер-фізичного керування в умовах «реального часу».

З огляду на вищеперераховані ризики використання методології RUP можна зробити однозначний висновок про недоцільність її застосування при розробці адитивного кібер-дизайну кібернетичної складової CPPS. Так само, в рамках даної роботи, були досліджені і проаналізовані наступні методології: ASD (Adaptive Software Development), DevOps (Development і Operations), DAD (Disciplined Agile Delivery), DSDM (Dynamic Systems Development Method), FDD (Feature Driven Development), LeSS (Large Scale Scrum), MDD (Model-Driven Development), MSF (Microsoft Solutions Framework), PSP (Personal Software Process), OpenUP (Open Unified Process), SAFe (Scaled Agile Framework), Scrum (SCRibing Unified Methodology), TSP (Team Software Process), UP (Unified Process), XP (Extreme Programming), а також наступні основні парадигми і моделі: Agile, Cleanroom, моделей ЖЦ

«Інтерактивної», «Спіральної», «Каскадної», «V-Model», «Dual Vee Model». В результаті їх аналізу зроблені висновки про можливість застосування до вирішення завдань при розробці адитивного кібер-дизайну:

– методологія DevOp базується на автоматизації розробки, яка заснована на стандартизації і схожа з моделлю Agile. Метою даної методології є використання архітектурних стилів мікросервісів. Дослідження компании «F5 Labs» показав, що тільки 17% розробників із 2200 опитаних обрали методологію DevOp базовою в своїх розробках. З огляду на специфіку розробки CPPS, складність і багатогранність архітектури, а також унікальність розробки під кожну конкретну задачу і мету, можна зробити висновок про неможливість використання стандартизації мікросервісів для розробки адитивного кібер-дизайну кібернетичної складової CPPS;

– методологія DAD доповнює методологію Scrum за допомогою гібридних доповнень: XP, UP, Канбана, ощадної розробки ПЗ, які були запропоновані компанією IBM. В її основі лежить підхід заснований на меті розробки, і представляє можливість вибору і модифікації фреймворку, в залежності від вимог до кожної розробки та обраної стратегії. Розглядаючи методологію DAD, з точки зору вирішення завдань данніх досліджень, можна зробити висновки, що дана методологія допускає масштабування, але не дозволяє його реалізувати в повній мірі, обмеження фреймворку накладає рамки його використання, що вимагає доопрацювання його кожен раз під новий тип завдань і вимоги, які накладаються на адитивний кібер-дизайн і доступний тільки на верхніх рівнях його ієрархії;

– методологія DSDM заснована на концепції методології RAD, тобто інтерактивному підході до розробки, недоліки якої, з точки зору даних досліджень, повторюють недоліки методології RAD, що описані вище;

– методологія FDD заснована на моделі гнучкої розробки (Agile) і використовує модульний підхід до розробки ПЗ. Основна парадигма даної методології заснована на обмеженні тимчасових рамок розробки функції (2 тижні), при неможливості їх досягнення в заданий термін, функції

разбиваються на підфункції і т.д. Основним недоліком методології FDD є залучення невеликих команд розробників певних функції, що ускладнює видимість основних цілей розробки, повної струкури розробки адитивного кібер-дизайну кібернетичної складової CPPS і зв'язків між рівнями та етапами, регулярна збірка розроблюваних функцій кожного разу коли виявляються помилки при їх об'єднанні. Дані недоліки, у відповідності до вимог з адитивного кібер-дизайну, можуть спричинити зміну архітектури, функціональних алгоритмів і привести до відхилення або недосягнення головної мети розробки CPPS;

– методологія MDD заснована на абстрактному описі ПЗ у вигляді моделей. Моделі тут представлені за допомогою ПОМ (предметно-орієнтованої мови). Модель визначається нотацією (сукупністю графічних елементів) і метамоделлю (інформація у вигляді метаданих), які в сумі використовуються у вигляді каркасів, які описують ПЗ, що розробляється. На базі даної методології розроблено Eclipse Modeling Framework, але має він має ті ж недоліки що і методологія DAD, з токи зору його застосування для розробки адитивного кібер-дизайну кібернетичної складової CPPS;

– методологія MSF використовується у гібридному зв'язку з методом Agile, і являє собою загальну базу методів, які відображають загальні методологічні підходи до інтерактивного проектування, що тягне за собою наступні недоліки їх застосування при розробці адитивного кібер-дизайну: немає певних принципів управління проєктами та пояснень різноманітних методів роботи, з самого початку не проводиться декомпозиція CPPS за рівнями, що тягне «проблеми» розробки на рівнях Fild level, Control level (рис. 1.4), а дані рівні є основними при вирішенні задач розробки кібер-фізичного керування організаційно-технічними виробничими об'єктами, у зв’язку з чим можна стверджувати про неможливість повного використання методології MSF для реалізації розробки адитивного кібер-дизайну кібернетичної складової CPPS;

– методологія PSP, за визначенням, не підходить з точки зору її

застосування для розробки адитивного кібер-дизайну кібернетичної складової CPPS, так як основним її аспектом є застосування принципів моделі зрілості процесів до практики одного розробника;

– методологія OpenUP заснована на ітеративно-інкрементальній моделі розробки і базується на легкому і гнучкому варіанті методології RUP, у зв'язку з чим вона успадкувала всі недоліки методології RUP;

– методологія SAFe являє собою гнучкий фреймворк на базі Agil і розглядає процес розробки з точки зору керівників для аналізу та управління поточними проектами, а отже дана методологія не підходить до вирішення завдань розробки адитивного кібер-дизайну кібернетичної складової CPPS.

Як можна бачити, всі проаналізовані методології, методи і моделі, в загальному випадку, відносяться до розробки інформаційних систем, що не враховує фізичну складову CPPS, яка є основоположною для розробки на рівнях Fild level, Control level (рис. 1.11) і впливає на структуру і реалізацію інформаційних потоків (на рівнях Production, Operations level) і механізмів математичного, функціонального і алгоритмічного опису інформації (на атамарних рівнях (датчики, PLC)), які впливають на процес управління, архітектуру і функціональні алгоритми кібер-фізичного керування складними організаційно-технічними виробничими об'єктами, як єдиної інформаційної еко-системи.

В результаті чого можна зробити висновки, що досліджені методології, методи та моделі разрахунку вартості [184-189], мають узагальнені підходи до розробки інформаційних систем і програмних продуктів і частково можуть використовуватися для розробки програмних модулів, і налаштувань для кібернетичної складової, у вигляді окремих програмних рішень певних типів завдань, що повністю не відповідає задачам які ставляться перед системами керування процесами керування організаційно-технічними виробничими об'єктами на базі кібер-фізичних систем.

1.8 Постановка мети і завдань дослідження

Процеси керування організаційно-технічними виробничими об'єктами на базі кібер-фізичних виробничих систем, являють собою складний «жорстко» ієрархічний об'єкт, який поєднує в собі природу як фізичних так і кібернетичних процесів, які, в колобарації, дозволяють створити цифрового близнюка реального виробничого процесу високотехнологічних виробів з можливістю автоматизації керування в режимі реального часу.

Розробка нових і модернізація існуючих CPPS керування складними організаційно-технічними виробничими об'єктами є складною науково-технічною задачею, яка носить індивідуальний характер та залежить від мети, застосування, обладнання, які використовуються в даному операційному процесі.

При цьому використання аналогічних CPPS керування неможливе через використання різних параметрів сенсорів, актюаторів, PLC, навіть в схожих операційних процесах, що веде до зміни кібернетичних потоків контролю фізичних величин і алгоритмів та функцій їх виконання, що в подальшому буде впливати на забезпечення ритмічності, продуктивності.

Тому одним з ефективних варіантів розробки нових або модернізації існуючих CPPS є застосування нових методів і моделей автоматизації процесів керування складними організаційно-технічними виробничими об'єктами, як синтезу фізичних і кібернетичних складових в єдиний цифровий інформаційний простір, який дозволить об’єднати всі інформаційні потоки в цілісну систему.

Основною науковою гіпотезою наукової роботи є підвищення продуктивності і ритмічності виробництва, за рахунок розробки нових методів і моделей автоматизації процесів керування складними організаційно-технічними виробничими об'єктами на базі кібер-фізичних виробничих систем. Для обгрунтування наукової гіпотези, на етапі експерементальних досліджень, в роботі поставлена мета – підвищення

ефективності виробничого процесу шляхом розробки методів, моделей і нової технології, алгоритмічного і програмного забезпечення керування організаційно-технічними об'єктами на базі кібер-фізичних виробничих систем.

Для досягнення поставленої мети в роботі розроблена узагальнена схема теоретичних, експериментальних і практичних досліджень, яка представлена на рис. 1.22.

На першому *теоретико-методологічному етапі* проводитися аналіз об'єкта дослідження – процеса кібер-фізичного керування складними організаційно-технічними виробничими об'єктами, як синтезу фізичних і кібернетичних параметрів в єдиному інформаційному середовищі.

На даному етапі досліджені і проаналізовані основні етапи зрілості CPPS, на базі моделі CMM, інтепретовані вимірювання моделі еталонної архітектури CPPS (RAMI 4.0), з точки зору мети і завдань досліджень.

Проводиться дослідження вертикальної інтеграції ієрархічних рівнів CPPS на базі моделі DIKW. Об'єднуючи результати аналізу архітектури та функціонування складних CPPS, а також ряд вимог, які висуваються до Smart Manufacturung, з точки зору автоматизації процесів керування організаційно-технічними виробничими об'єктами. До даних вимог віднесено: наявність єдиного інформаційного простору, реалізація технології «Цифрових близнюків», присутність якостей самоадаптації, самодіагностики. Та аналіз існуючих методологій, парадигм і моделей розробки ПП і ПЗ, показує тільки часткову можливість їх використання при реалізації адитивного кібер-дизайну кібернетичної складової CPPS.

Таки чином виявляється і обгрунтовується науково-прикладна проблема розробки ефективної стратегії автоматизації управління складними організаційно-технічними виробничими об'єктами, шляхом реалізації комплексу моделей, методів процесів керування і технології на базі кібер-фізичних систем.

Для досягнення поставленої мети, в даному дослідженні, необхідно

вирішити наступні завдання:

– провести аналіз сучасного стану існуючих проблем концепцій, архітектури, методологій, моделей і методів процесів керування організаційно-технічними об'єктами на базі кібер-фізичних виробничих систем;

– розробити архітектурно-логічну модель керування процесом розробки CPPS, з урахуванням вимог висунутих технологіями «Digital Twins»;

– розробити методи прийняття рішень на кожному етапі архітектури і технології процесів управління розробкою CPPS;

– запропонувати технологію розробки кібер-фізичних виробничих систем;

– розробити метод синтезу алгоритмів функціонування CPPS і формалізації структурних системних моделей;

– розробити модель формалізації кібернетичної складової на базі параметрів і подій GUI (графічних елементів), які реалізують HMI (людино-машинний інтерфейс) системи керування організаційно-технічними об'єктами;

– розробити синтаксичну та семантичну моделі мови визначення і опису параметрів, необхідних і достатніх для автоматизації процесів розробки CPPS;

– розробити програмне забезпечення для реалізації розроблених моделей і методів та провести експериментальні дослідження ефективності отриманих теоретичних результатів.

На останьому етапі практичної реалізації результатів дослідження і наукових розробок проводиться оцінка техніко-економічної ефекту (продуктивності і ритмічності виробництва). Для вирішення поставлених завдань, в рамках даної роботи, запропонована наступна методологія кібер-фізичного керування процесами організаційно-технічних виробничих об'єктів, яка представлена на рис. 1.23.

<table>
<tr><td rowspan="3">І етап</td><td colspan="2">Теоретико-методологічні дослідження</td></tr>
<tr><td colspan="2">Аналіз об'єкта дослідження – процес кібер-фізичного керування складними організаційно-технічними виробничими об'єктами</td></tr>
<tr><td>Проблема – ефективність стратегії автоматизації керування складними організаційно-технічними виробничими об'єктами, шляхом реалізації комплексу моделей, методів процесів керування і технології на базі кібер-фізичних систем.</td><td>Проблема, як потенційна можливість розробки нових методів, моделей і технологій процесів керування складними організаційно-технічними виробничими об'єктами на базі кібер-фізичних виробничих систем</td></tr>
<tr><td rowspan="4">ІІ етап</td><td colspan="2">Експериментальні дослідження</td></tr>
<tr><td colspan="2">Визначення рівнів і етапів процесу керування складними організаційно-технічними виробничими об'єктами у вигляді синтезу кібернетичних і фізичних параметрів на базі теорії метасистем і методів декомпозиції</td></tr>
<tr><td>Мета досліджень – підвищення ефективності виробничого процесу шляхом розробки методів, моделей і нової технології, алгоритмічного і програмного забезпечення керування організаційно-технічними об'єктами на базі кібер-фізичних виробничих систем.</td><td rowspan="2">- розробити архітектурно-логічну модель і методи декомпозиції кібер-фізичного керування процесами в складних організаційно-технічних об'єктах;
- розробити методи процесу керування організаційно-технічним об'єктом, як логічно узгоджені послідовності прийнять рішень;
- розробити технологію безперервного процесу керування організаційно-технічним об'єктом;
- розробити метод синтезу алгоритмів функціонування кібер-фізичного керування;
- розробити модель формалізації кібернетичної складової організаційно-технічного об'єкта;
- розробити синтаксичну та семантичну модель декларативного мови визначення і маніпулювання даними;</td></tr>
<tr><td>Завдання дослідження:
– розробити архітектурно-логічну модель керування процесом розробкою CPPS, з урахуванням вимог висунутих технологіями «Digital Twins»;
– розробити методи прийняття рішень на кожному етапі архітектури і технології процесів управління розробкою CPPS;
– запропонувати технологію розробки кібер-фізичних виробничих систем;
– розробити метод синтезу алгоритмів функціонування CPPS і формалізації структурних системних моделей;
– розробити модель формалізації кібернетичної складової на базі параметрів і подій GUI (графічних елементів), які реалізують HMI (людино-машинний інтерфейс) системи керування організаційно-технічними об'єктами;
– розробити синтаксичну та семантичну моделі мови визначення і опису параметрів, необхідних і достатніх для автоматизації процесів розробки CPPS;</td></tr>
<tr><td colspan="3">«Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом»</td></tr>
<tr><td rowspan="2">ІІІ етап</td><td colspan="2">Практична реалізація результатів досліджень і наукових розробок</td></tr>
<tr><td>Оцінка техніко-економічної ефекту (продуктивності і ритмічності виробництва) від впровадження результатів дослідження</td><td>Висновки за результатами досліджень. Отримання свідоцтва про реєстрацію авторського права. Промислове впровадження отриманих результатів наукової роботи.</td></tr>
</table>

Рисунок 1.22 – Узагальнена схема теоретичних, експериментальних і практичних дослідженнь

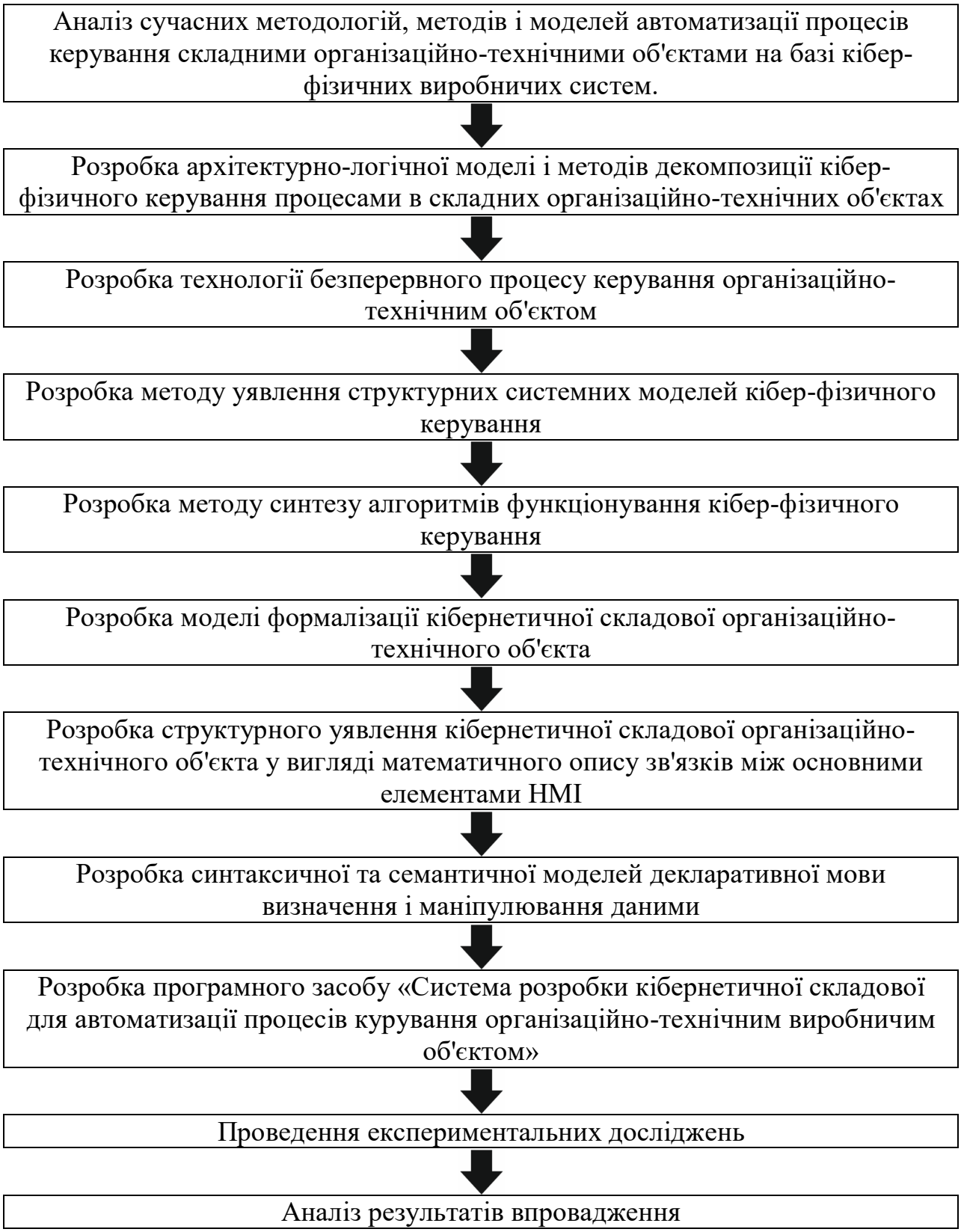


Рисунок 1.23 – Методологія дослідження кібер-фізичного керування процесами в організаційно-технічних виробничих об'єктах

Висновки до розділу 1

Розробки адитивного кібер-дизайну кібернетичної складової процесів керування організаційно-технічними виробничими об'єктами на базі CPPS є цифровими близнюками (Digital Twins) реальних виробничих процесів і вирішують великий спектр завдань пов'язаних з автоматизацією керування процесами, що протікають при виробництві високотехнологічних виробів на всіх етапах їх життєвого циклу. Впровадження таких систем дозволить керувати і контролювати виробництво в режимі реального часу, зробить її самоадаптованою, самодіагностованою та дасть можливість оптимізувати виробничий процес і життєвий цикл виробу, що є одним з критеріїв для досягнення цілей Lean Production (LP).

Дослідження в цій галузі є досить новим напрямком, який виник з появою концепції Industry 4.0 і технології IIoT, тому аналіз останніх публікацій показав, що CPPS є складним об'єктом, який об'єднує в собі фізичні і кібенетичні складові, архітектура і структура яких залежить від вимог (конструкторські, технологічні парметри, вимоги до обладнання, PLC, датчиків, виконавчих механізмів і т.д.). Також виявлена залежність вибору методів контролю і керування процесами, способів моніторингу, аналізу і зберігання великих обсягів даних, які повинні бути узгоджені в режимі реального часу для уникнення появи браку або простою обладнання, що зменшує економічну ефективність виробництва.

Незважаючі на сучасні дослідження в даній області і велику кількість науково-дослідних робіт, залишається невирішеною проблема відсутності ефективної стратегії автоматизації керування складними організаційно-технічними виробничими об'єктами, шляхом реалізації комплексу моделей, методів процесів керування і технології на базі кібер-фізичних систем.

Тому розробка нових методів, моделей і нової технології, алгоритмічного і програмного забезпечення керування організаційно-технічними об'єктами на базі кібер-фізичних виробничих систем є

актуальним завданням.

За результатами проведеного аналізу сформована мета і завдання дисертаційної роботи. Запропоновано методологію та розроблена узагальнена схема теоретичних, експериментальних і практичних досліджень, які передбачають розробку нових і вдосконалення існуючих математичних методів, моделей, технології, алгоритмічного і програмного забезпечення управління організаційно-технічними об'єктами на базі кібер-фізичних виробничих систем, для розв'язання задачі підвищення продуктивності і ритмічності виробництва від впровадження результатів дослідження.

Список використаних джерел в розділі наведено в повному списку використаних джерел [1–189].

2 МЕТОДИ ТА МОДЕЛІ КЕРУВАННЯ ПРОЦЕСАМИ В СКЛАДНИХ ОРГАНІЗАЦІЙНО-ТЕХНІЧНИХ ВИРОБНИЧИХ ОБ'ЄКТАХ

2.1 Архітектурно-логічна модель керування процесами в складних організаційно-технічних виробничих об'єктах

Сучасні кібер-фізичні виробничі системи, в рамках Industry 4.0, можна розглядати як складні багаторівневі комп'ютерно-інтегровані системи на всіх рівнях керування. Прикладами таких систем є системи, що застосовуватися в таких сферах діяльності людини: хостінг (Amazon, IBM, Oracle, Bosch, ATAT, Rockspoce, і.т.д.), промислові IoT платформи (Samsung, OLEX, PTC , Siemens, Exosite, Cisco, Atlizon, і. т.д.), аналітика (SAR, Mnubo, Seeq, Smart Cloud, ARM Treasure data, MAPR, Relayr, і т.д.), мікросхеми (ARM, Micron, ST , NVidia, Intel, і т.д.), датчики (ABB, JUMO, FESTO, SICK, KISTLER, TE і .т.д), обладнання для підключання (ABB, DELL, MOXA, LOGIC, KEB, iEi, ADELINK), кібербезпека (ARGUS, ARM, Arilou, Bastille, CyberBit, Indegy, Senrio, MOCANA і т.д.), інтеграція систем (Accenture, Atos, CGI, Cognizant, HCL, InfoSys, Kalypso, T-Systems і т.д.), адитивне виробництво (3D System, AddUp, AutoDesk, ArcamEBM, Arkema, DesktopMetal, HP, DMG MORI, ExOne, і т.д.), розширена та віртуальна реальність (Sony, HTC, Atheer, Brother, UBiMAX, і т.д.), колоборативні роботи (ASS, KUKA, YASKAWA, ROBOTIQ, MAGAZINO, і. т.д.), підключене машинне бачення (BASLER, COG NEX, FESTO, FLIR, OMRON, SICK), безпілотні літальні апарати (дрони) (Kespry, DJI, Skycatch і т.д.), самохідні машини (AETHON, AmazonRobotics, BALYO, STILL, GreyOrange, Omron і т.д.) [21].

Особливістю керування процесами в складних організаційно-технічних виробничих об'єктах є їх складний багаторівневий характер, що визначає великі труднощі пов'язані з нескінченними варіантами їх побудови як на

кібернетичному так і на фізичному рівнях, з урахуванням інтеграції всіх можливих зв'язків між ними.

Іншою особливістю систем керування процесами в складних CPPS є їх індивідуальність і одиничний характер, тобто не існує прототипів і аналогів, з яких можна частково або повністю перенести структуру кібер-фізичної системи. В результаті чого необхідно знаходити спільні закономірності процесів керування при розробці таких систем і проводити узагальнення не на типових рішеннях, а на рівні складних багаторівневих організаційно-технічних виробничих об'єктів.

Третьою особливістю є слабка методологічна база їх розробки, що пояснюється відсутністю адекватних засобів формального опису таких складних об'єктів і засобів алгебраїчного перетворення цих описів, що стримує розвиток методів структурно-параметричного синтезу CPPS і їх аналізу у вигляді системних складних моделей.

Четвертою особливістю є синхронний процес розробки організаційно-технічних виробничих об'єктів, як самого об'єкта, так і його системи керування, який спочатку носить паралельно-послідовний характер «зверху-вниз» і суворо відповідає логічній послідовності.

Модель керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS включає в себе два основних підходи до кожного рішення:

– підхід до вибору конкретного рішення (синтезу) цільових, функціональних, структурних, інформаційних і алгоритмічних рішень;

– аналіз кожного прийнятого рішення за якістю, змістом і методами моделювання.

Модель об'єднує в собі складний комплекс логічно пов'язаних методів прийняття рішень. Спочатку вводяться і визначаються такі поняття: «дерево», «рівень декомпозиції» і «етапи». Далі, можна сказати, що керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS – це структура, розподілених по «рівню декомпозиції»

та «етапам» методів рішень, а «древо» представляє собою деревоподібну ієрархічну структуру з «рівнів декомпозиції».

Взявши до уваги, що складні організаційно-технічні виробничі об'єкти, в рамках технологій Industry 4.0, є складними *n*-рівневими об'єктами, то розробку даної моделі пропонується почати з системного аналізу методом декомпозиції і розбиття організаційно-технічного виробничого об'єкту на складові компоненти за класифікаційними властивостями, або ознаками, що дозволять визначити рівень його ієрархії, а отже визначити його системний вид. Одним з ключових моментів багаторівневої декомпозиції CPPS є виявлення і виділення їх властивостей, які визначаються метою і завданнями створення CPPS, для керування процесами в складних організаційно-технічних виробничих об'єктах. Дане рішення впливає на всі етапи процесу керування розробкою CPPS, так як визначає не тільки структуру, а й «дерево».

Ґрунтуючись на вищевказаному можна зробити висновок, що на ранніх етапах розробки CPPS необхідно використовувати цільову декомпозицію, тобто на першому кроці процесу керування розробкою CPPS неможливо обійтись без системного аналізу мети і завдання, які дозволять окреслити і виділити пріоритетні ознаки і властивості, за якими є можливість провести декомпозицію на підцілі нижнього рівня.

Дані ознаки і властивості можна отримати за допомогою аналізу близьких за метою та завданнями існуючих CPPS, а також мети і задачам CPPS, що розробляється. Варто звернути увагу, що рівнева декомпозиція дає можливість визначити функціонал, структуру і алгоритм функціонування CPPS на базі мети і завдання керування процесами в складних організаційно-технічних виробничих об'єктах. Отже, мета та завдання розробки є доменатними, так як централізують в собі всі інші властивості CPPS.

Ґрунтуючись на базі теорії великих систем, [190,191] введемо в дане дослідження наступні поняття:

– атомарний елемент системи ($AEofS_i$) – це неподільна найелементарніша частина системи, яка не складається з будь-яких частин, де $i=0…n$. Тобто атомарні елементи системи, які за своїми властивостями, що характеризують їх фізичну або інформаційну природу, можуть бути як однаковими так і різними. Звідси можна записати умову, яка на самому нижньому елементарному рівні складатиметься з безлічі $AEofS_i$. У результаті ліквідним є запис $CPPS \subset AEofS_i$. Даний нижній рівень визначимо як «рівень декомпозиції першого рівня» CPPS. Тоді уявлення CPPS через елементи першого рівня можна описати у вигляді $CPPS\_AEofS_i$ при $i=1…n$;

– група елементів ($G\{AEofS\}$) *та* $AEofS_i$, які об'єднані за принципом володіння однаковими властивостями на атомарному рівні. Введення ($G\{AEofS\}$) дозволить представити CPPS через ($G\{AEofS\}$), а отже дозволить її описати у вигляді $CPPS \subset G\{AEofS\}_j$, де $j=1…m$, що відповідає другому рівню складності уявлення CPPS і в даній методології визначається як декомпозиція другого рівня;

– підсистема (Sub_S) – це уявлення CPPS на базі ($G\{AEofS\}$), які об'єднані в групу і володіють властивими їм природними властивостями. Тоді уявлення CPPS, на базі підсистеми, дасть можливість визначити третій рівень складності об'єкта $CPPS \subset Sub\_S_k$, де $k=0…d$;

– мультисистема (MS'') – об'єднання (Sub_S) складних об'єктів, які мають свої властивості. Це дає можливість представити об'єкт як четвертий рівень декомпозиції CPPS, а отже її можна описати виразом $MS'' \subset Sub\_S_k$;

– метасистема (MS') – об'єднання MS'' за властивостями відповідно до вище описаної логіки.

Використовуючи запропонований метод подання CPPS через рівні і глибину розробки з’являється можливість провести декомпозицію моносистеми до четвертого порядку, а з четвертого – як мультисистеми, з п'ятого – як метасистеми з нескінченним числом рангів. Дане рішення дозволить провести дослідження керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS, визначити рівень

декомпозиції, а також рівень розробки структури на кожному етапі, використовуючи метод «зверху-вниз». Запропоновані в даній роботі рішення дають можливість будувати вертикальний каркас декомпозиції всіх рівнів процесу управління розробкою CPPS відповідно до поставленої мети.

Проведений аналіз ISO 10303-235:2019 [192], ISO 18828-3:2017 [193], ISO 18828-5:2019 [194] показав, що розробці методів керування процесами в складних організаційно-технічних виробничих об'єктах в даних стандартах не приділено достатньо уваги. В них наведені лише загальні відомості, основні принципи та послідовність завдань на різних етапах, у зв’язку з чим їх неможливо застосувати до сучасних кібер-фізичних систем і тому вони не можуть бути прийняті за основу.

Для подальших досліджень керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS необхідно провести класифікацію прийняття рішень за певними властивостями, які дозволять досягти мети розробки CPPS.

В ході аналізу існуючих методів, які дозволяють провести класифікацію, був обраний метод стратифікації. Даний метод включає в себе певний набір рішень за різними властивостями об'єкта. Виходячи з цього, для визначення властивостей об'єкта, необхідно використовувати методи системного аналізу.

Дослідження керування процесами в складних організаційно-технічних виробничих об'єктах на базі кібер-фізичних систем, можна в загальному випадку згрупувати за такими рішеннями:

Загальні (стратегічні):

– мета керування;

– функції керування;

Кібернетична складова:

– інфологічна структура;

– алгоритми управління;

– інформаційна структура;

– програмне забезпечення;

Фізична складова:

– функції управління;

– організаційно-технічна структура;

– комплекс технічних засобів у вигляді мехатронних пристроїв (ПЛК, датчики, технологічне обладнання, мережі і т.д.).

При цьому необхідно врахувати, що процес керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS, може протікати як в синхронному так і в асинхронному тимчасових режимах, що вимагає встановлення жорсткої послідовності виконання рішень, так як їх невиконання призведе до зміни термінів і підвищення вартості впровадження CPPS на підприємстві.

Як обґрунтування існування суворої логічної послідовності процесу керування розробкою складних CPPS можна навести такі протиріччя:

– не маючи проектних рішень з алгоритмічного забезпечення і комплексу технічних засобів неможливо розробити PLM, MES, ERP;

– необхідне алгоритмічне забезпечення функціонування організаційно-технічних виробничих об'єктах на базі CPPS відбувається на базі математичного забезпечення;

– математичне забезпечення розробляється на базі завдань керування, які повинні обиратися залежно від мети і завдання розробки кожної CPPS для окремих організаційно-технічних виробничих об'єктів.

Тому процес керування в організаційно-технічних виробничих об'єктах, на базі сучасних кібер-фізичних систем, будується на базі цільових алгоритмів керування та інформаційних потоків взаємодії структурних елементів об'єкту керування, які залежать від їх природи і фізичних властивостей. Простий приклад вищенаведених протиріч показує, що кожне рішення строго залежить від попереднього і існує «жорстка» логічна послідовність всіх рішень при розробці тієї чи іншої CPPS керування складними організаційно-технічними виробничими об'єктами.

Виходячи з цього, для архітектурно-логічної моделі керування процесами в складних організаційно-технічних виробничих об'єктах, що

розробляється, необхідно визначити послідовність прийняття рішень, для чого пропонується наступна ієрархія:

– стратегічний етап;

– функціональний етап;

– організаційно-технічний етап;

– інфологічний етап;

– інформаційний етап;

– етап алгоритмів функціонування;

– рівень фізичного і імітаційного моделювання елементарних завдань;

– рівень математичного опису елементарних завдань.

Запропонована послідовність прийняття рішень не є повною так як вона не охоплює завдання керування CPPS, що розробляється.

Для забезпечення повного кортежу необхідних цілей і параметрів, для досягнення завдань розробки сучасних CPPS, пропонується розширити його за допомогою таких установок керування:

– цільова система керування;

– функціональна система керування;

– організаційна система керування;

– інформаційна система керування;

– фізична система керування;

– математична;

– алгоритми керування.

Ґрунтуючись на запропонованих параметрах керування і ухвалення рішень в даній послідовності, можна зробити висновки, що етапи носять складний багаторівневий характер, їх об'єднання дозволить окреслити дерево розробки складних CPPS керування процесами в складних організаційно-технічних виробничих об'єктах з розбиттям їх на рівні декомпозиції [195]. Архітектурно-логічна модель керування процесами в складних організаційно-технічних виробничих об'єктах представлена на рис. 2.1

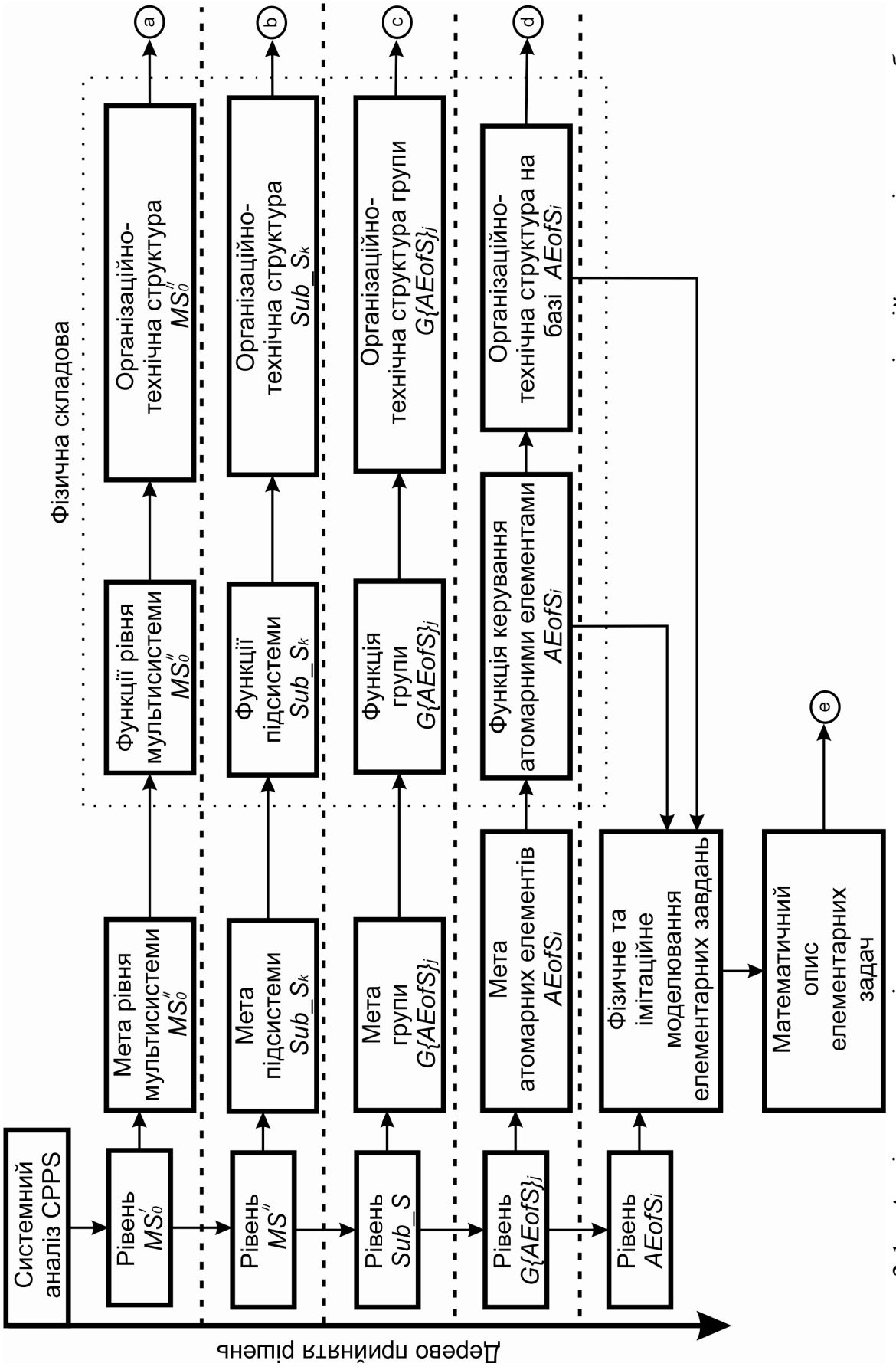


Рисунок 2.1 – Архітектурно-логічна модель керування процесами в складних організаційно-технічних виробничих об'єктах

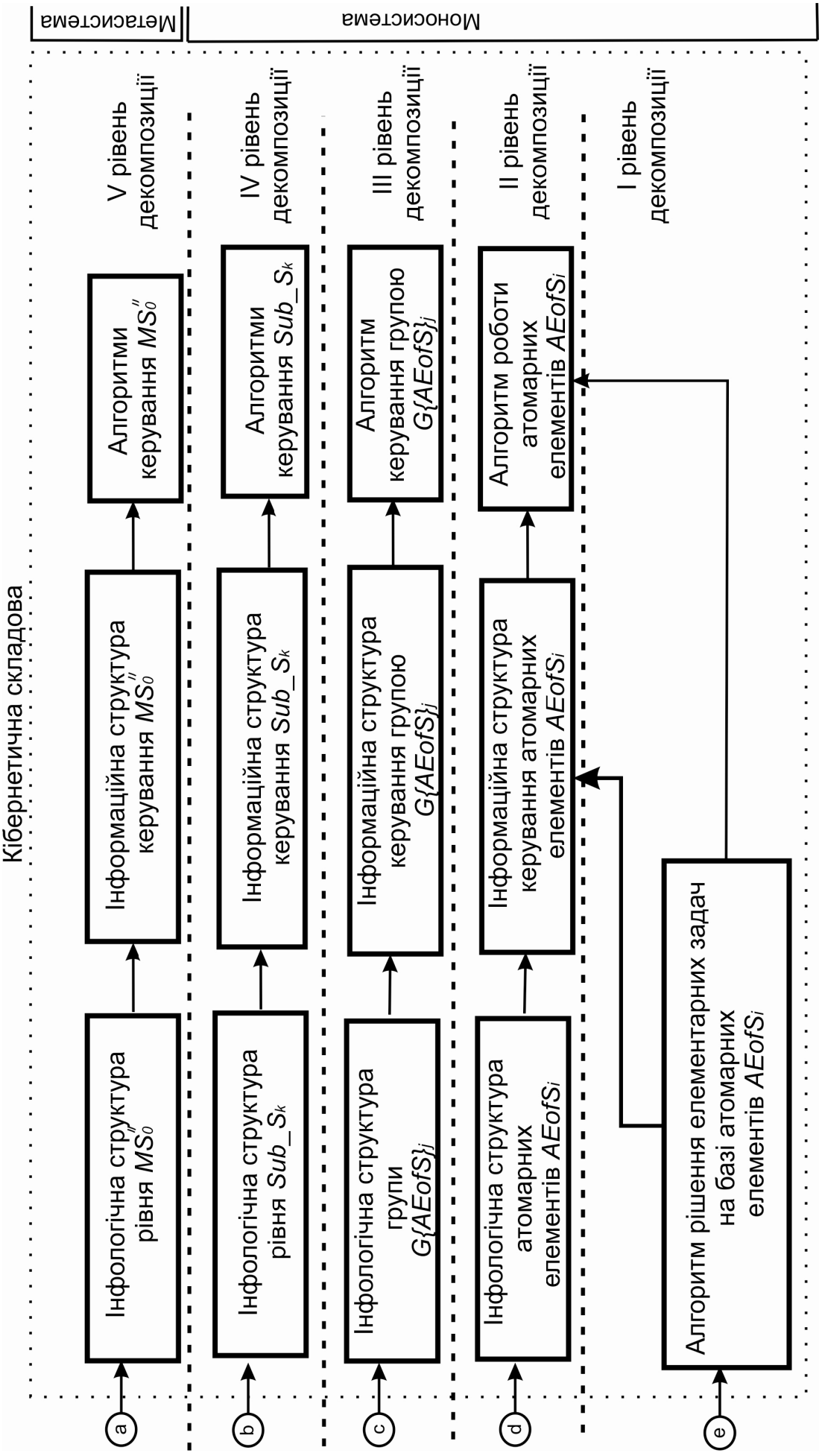


Рисунок 2.1, аркуш 2

2.2 Метод декомпозиції архітектурно-логічної моделі керування процесами в складних організаційно-технічних виробничих об'єктах на початковому етапі

Кібер-фізична виробнича система являє собою інтеграцію тісно взаємопов'язаних властивостей груп об'єктів: стратегічна (мета, завдання), функціональна, структурна, інфологічна і алгоритм їх функціонування на кожному рівні декомпозиції. В даному дослідженні пропонується за вище перерахованими ознаками проводити декомпозицію метасистем в системи, між якими можуть існувати зв'язки з більш сабо вираженим ефектом.

Проводячи аналіз реального керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS можна провести декомпозицію гнучкого автоматизованого заводу, за функціонально-технічними ознаками, на автоматизовані цеха з верстатами з ЧПУ, які тісно пов'язаними властивостями: технологія виробництва, технологічне обладнання і комп'ютерно-інтегроване керування. Однак, варто зауважити, що зв'язки між цехами, у порівнянні з внутрішньоцеховими, мають не настільки яскраво виражений ефект.

У свою чергу, декомпозицію автоматизованого цеху можна провести за такими ознаками: дільниці, які мають свої функціональні цілі; технологічні процеси і функціонування. Використовуючи метод діалектичного перенесення, з наведеного прикладу ознак декомпозиції метасистем, можна виділити ознаки, які дозволять декомпозувати метасистеми в системи:

– стратегічні;

– функціональні;

– організаційні;

– інфологічні;

– функціонування.

Основне завдання розробника CPPS – провести повний системний аналіз сучасних аналогів та виділити базові об'єкти, виходячи з головної мети розробки CPPS, визначити дерево прийняття рішень.

В даному дослідженні фіксуються такі рівні декомпозиції розробки CPPS по MS'_0, MS''_0, Sub_S_k, $G\{AEofS\}_j$, $AEofS_i$, що дасть можливість провести повний аналіз і отримати древо побудови будь-якої CPPS, з урахуванням як кібернетичних так і фізичних складових.

На початковому етапі розробки CPPS, залежно від головної мети, необхідно побудувати древо прийняття рішень. Для цього розробник повинен провести декомпозицію головної мети розробки CPPS на підцілі. З цією метою проводиться повнофакторна науково-дослідна робота з аналізу головної мети, яка у більшості випадків містить якісні та кількісні характеристики, у вигляді масиву вимог до технічних і інформаційних складових.

У відповідності до запропонованої архітектурно-логічної моделі автоматизації процесу управління розробкою складних CPPS, наведеної на рис. 2.1, можна визначити, що на 5 рівні розробки необхідно провести декомпозицію головної мети метосистеми (MS'_0) на підцілі, які формують цілі для кожної мультисистеми (MS''_0), за умов $MS'_0 \subset MS''_0$, які повністю відповідають головній меті розробки. Методика декомпозиції MS'_0 наступна:

– проводиться аналіз властивостей MS''_0;

– вивчаються і аналізуються зв'язки між групами MS''_0;

– досліджується можливість досягнення головної мети MS'_0;

– визначаються закономірності відносин між MS'_0 і MS''_0;

– аналітичним методом розраховуються якісні і кількісні показники вимог для кожної підцілі $(QandQIR\_MS''_0)$, при цьому вони можуть бути як поодинокі так і об’єднані в групи;

– розглядається можливість існування співвідношень в групах MS''_0, які володіють своїми цілями. Дані співвідношення аналізуються і розраховується $QandQIR\_MS''_0$;

– будується графова модель за принципом вузли – це значення $QandQIR\_MS''_0$ графа мети MS'_0, а співвідношення – ребра графа;

– проводиться перевірка правильності побудови $QandQIR\_MS''_0$ для MS''_0 і $QandQIR\_\ MS'_0$ для MS'_0, за допомогою розв'язання оберненої задачі суперпозиції параметрів;

– якщо в ході перевірки система параметрів підцілей MS''_0 складе кількісне значення параметрів головної мети MS'_0, то розробник може вважати, що рішення з декомпозиції головної мети MS'_0 розробки CPPS на цілі MS''_0 є правильними.

Узагальнена структура запропонованого методу представлена на рис. 2.2. Для зручності розуміння позначимо процес моделювання за допомогою графа цілей – через системне моделювання CPPS, а побудову графа цілей системи – через системну цільову модель CPPS.

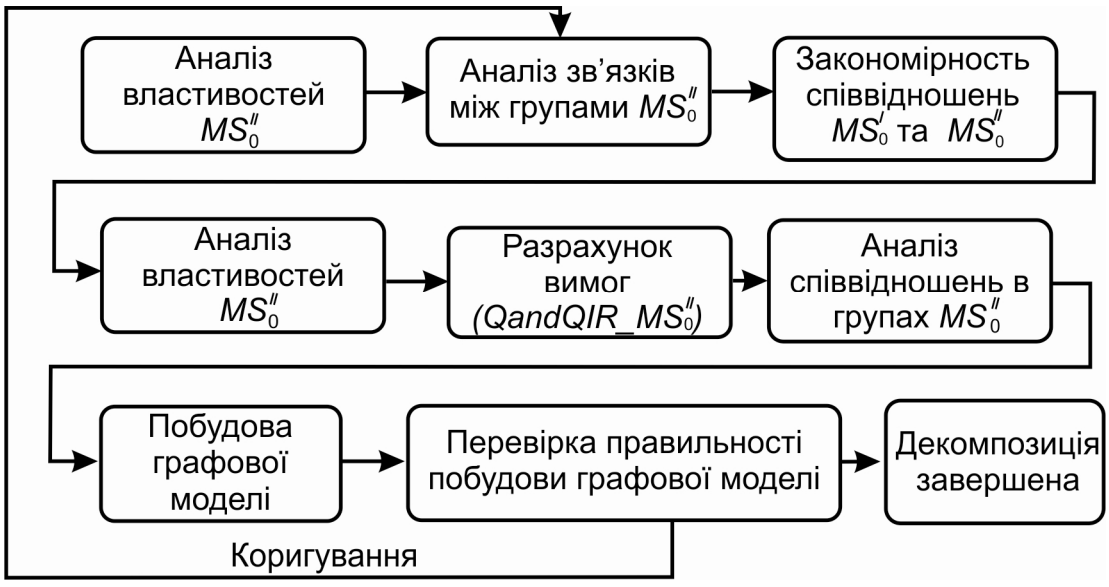


Рисунок 2.2 – Узагальнена структура методу декомпозиції кібер-фізичних виробничих систем як метасистеми (MS'_0)

Логічно, що запропоновані рішення повинні проводитися на кожному рівні процесу управління розробкою CPPS: системному (MS''), підсистемному (Sub_S), груповому ($G\{AEofS\}$) і за елементарними цілями на рівнях атомарних елементів системи ($AEofS$). Варто зауважити що уявлення MS'_0,

через декомпозицію кожного рівня, підвищить розмірність графа цілей, що в свою чергу ускладнить зв'язки між усіма елементами.

Звідси виникає завдання адекватної обробки і аналізу цільової декомпозиції, особливо це буде видно на атомарних нижніх рівнях. Для цього необхідним буде застосування методів системного аналізу та комп'ютерних системних моделей, а графові системні цільові моделі будуть необхідні тільки для наочного уявлення процесу моделювання.

2.3 Розробка методів прийняття рішень на рівні фізичної складової CPPS

2.3.1 Розробка методу прийняття рішень на функціональному етапі

Відповідно запропонованій архітектурно-логічній моделі керування процесами в складних організаційно-технічних виробничих об'єктах (рис. 2.1) першим йде етап подання та опис процесів керування розробкою фізичної складової CPPS. На цьому етапі розробляється комплекс завдань для досягнення головної мети розробки CPPS, який формується на базі апаратних засобів. Логічно зауважити, що всі рішення, які ухвалюватимуться на функціональному рівні будуть відбуватися на всіх рівнях декомпозиції і будуть дзеркальним відображенням головної мети розробки організаційно-технічного виробничого об'єкта. В рамках даних досліджень пропонується наступний метод рішень на функціональному рівні:

– фіксується перший рівень декомпозиції на атомарному рівні ($AEofS_i$) (рис. 2.1);

– проводяться дослідження галузі знань (засоби досягнення мети функціонування фізичної складової) на цьому рівні;

– визначаються завдання або група завдань досягнення поставленої мети. Тобто для кожної мети розробляються завдання, які дозволяють досягти мети, з урахуванням їх ефективності та запобігання надлишковості даних.

Введемо Aim_i, як мету одного з рівнів декомпозиції CPPS. Тобто для кожної мети (Aim_i) розробляються завдання ($Task_j$), з урахуванням їх ефективності та запобігання надлишковості даних. Це можна представити у вигляді бінарних відношень $AT \subset \{Aim_i\} \times \{Task_j\}$:

$$Aim_i \cong Task_j \Leftrightarrow (Aim_i, Task_j) \in AT . \quad (2.1)$$

У випадку існування декількох $Task_j$, для досягнення Aim_i, таку множину завдань (TA_i) можна представити:

$$TA_i = \{t \mid (Aim_i, t) \in AT\} . \quad (2.2)$$

Для кожного кількісного параметра мети *QandQIR* обираються атомарні елементи $AEofS_i$, для яких розраховується тактико-технічні характеристики завдання (*PCofT*). Дані характеристики повинні повністю відповідати заданим цілям та враховувати необхідність проведення розрахунку за правилами опису явищ предметної області, а отже функціонування об'єкта даного рівня розробки;

– проводиться побудова системної графової функціональної моделі CPPS за наступними принципами: вузли графа розташовуються у відповідності послідовності розроблених завдань ($Task_j$), а зв'язком між $Task_j$ виступають Aim_i у вигляді ребер графа. У результаті чого розробник отримує орієнтований функціональний граф досягнення мети, у вигляді завдань ($FCofTask_j$). Запропоновані рішення необхідні для контролю правильності ухвалення рішень на функціональному етапі даного рівня декомпозиції;

– проводиться перевірка на функціональну повноту $Task_j$, для вирішення всіх поставлених Aim_i. Для цього порівнюються графи заданого рівня розробки і завдань цього рівня. Отже можна стверджувати, що граф мети для

MS''_0 (Aim_MS'') є «батьківським» для графа завдань MS''_0 ($Task\_MS''_0$), що дозволяє стверджувати про гомоморфізм графів $Aim\_MS''_0$ і $Task\_MS''_0$;

– розмічається системний функціональний граф параметрами $PCofI\_MS''_0$, що дозволить зробити системне моделювання декомпозиції $PCofI\_MS''_0$, на відповідність тактико-технічним вимогам мети $QandQIR\_MS''_0$, за критеріями ефективності та специфічних параметрів даної MS'_0. Варто зауважити, що особливістю моделювання зазначених критеріїв, є те що його можливо провести на функціональному графі системи. Якщо результати моделювання $PCofI\_MS''_0$ повністю задовольняють заданим $QandQIR\_MS''_0$ і головній меті MS'_0, тоді можна з упевненістю вважати рішення на функціональному етапі системного рівня процесу управління розробкою CPPS правильним. Узагальнена структура методу представлена на рис. 2.3.

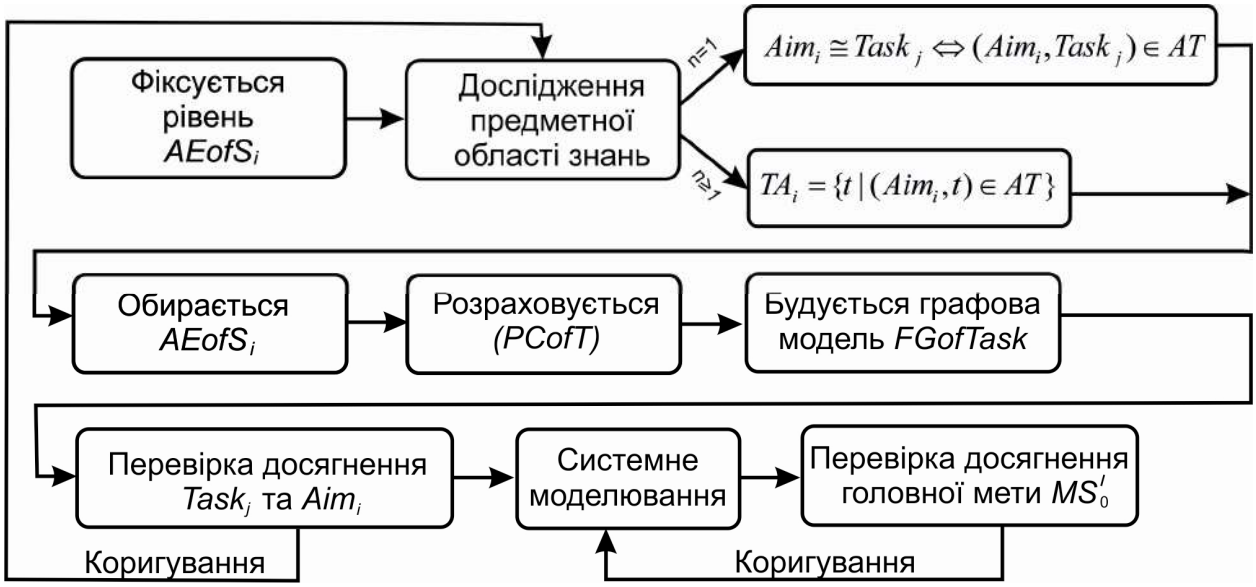


Рисунок 2.3 – Узагальнена структура методу прийняття рішень на функціональному етапі

2.3.2 Розробка методу прийняття рішень на організаційно-технічному етапі

Розглянемо наступний етап розробки фізичної складової CPPS. Як можна бачити з рис. 2.1 даний етап визначає засоби (апаратні), на базі яких вирішується завдання, або комплекс завдань досягнення головної мети розробки CPPS. Позначимо їх як організаційно-технічну структуру CPPS, на певному рівні декомпозиції (*Pattern_MS″₀*), для рівня *MS″₀*, що складається з елементів (*1…m*) даного рівня декомпозиції об'єкта. Уявімо запропонований структурний елемент *Pattern* у вигляді абстрактного процесу, який призначений для функціонального рішення *Task*, або групи $Task_j$ функціонування CPPS на даному рівні декомпозиції.

Як можна бачити з цього твердження розробник створює прообраз комплексу технічних засобів CPPS, на рівні фізичної складової, які функціонально здатні вирішувати необхідні завдання ($Task_j$). Для досягнення проектних рішень на організаційно-технічному етапі пропонується наступний метод:

– проводиться дослідження $Task_j\_MS''_0$ системного рівня розробки на предмет можливості бути реалізованим і досягти $Aim_i\_MS''_0$, а також аналізуються зв'язки у *FCofTask_MS″₀* на предмет «сильних» і «слабких» зв'язків;

– на базі ознак «реалізованості» і «слабкості» зв'язків $Task_j\_MS''_0$ групуються, з метою реалізації на своєму структурному елементі (*StrE_MS″₀*) даного рівня розробки об'єкта. Це дасть можливість знайти відповідності між $Task_j\_MS''_0$ і *StrE_MS″₀*, на яких вона може бути реалізована. Далі проводиться структурно-функціональний синтез об'єкту для досягнення необхідного рівня за специфічними особливостями (технічними даними) властивих $AEofS_i$ або $G\{AEofS\}_j$. За результатами будується бінарне співвідношення:

$$Task_j\_MS_0'' \cong Pattern\_MS_0''. \quad (2.3)$$

– на даному етапі методу проводиться з'єднання $Pattern\_MS''_0$ в систему, за допомогою ототожнення зв'язків між $Task_j\_MS''_0$, як всередині $AEofS_i$ так і між ними. Існуючі зв'язки між $Task_j\_MS''_0$ різних структурних елементів ($AEofS_i$), визначають зв'язки $Pattern\_MS''_0$, у результаті чого розробник отримує системну модель організаційно-технічної структури CPPS, яка відповідає заданому рівню процесу керування;

– аналізуються $PCofI\_MS''_0$ (технічні характеристики), на рівні MS''_0, які закріплені за кожним $StrE\_MS''_0$. За результатами даного аналізу розробник розраховує всі параметри технічних характеристик $Pattern\_MS''_0$ структурних елементів, що реалізують Aim_i. Модель розрахунків параметрів для $PCofI\_MS''_0$ і для будь-якого рівня визначаються закономірностям предметної області знань, до якої належить використаний розрахунковий процес, аналогічно розраховуються параметри продуктивності, надійності, точності для кожного *Pattern*;

– на останньому етапі проводиться моделювання $PCofI\_MS''_0$ на відповідність $TrandTR\_Aim\_MS''_0$ та критеріям ефективності декомпозиції структурних елементів. Отриманий результат моделювання повинен показати відповідність показникам продуктивності, точності та надійності елементів, що піддані декомпозиції: $StrE\_MS''_0$ і $TrandTR\_Aim\_MS''_0$ і, як наслідок, можна зробити висновок, що прийняті рішення структурно-функціонального синтезу даного рівня декомпозиції відповідають Aim_i на CPPS.

Узагальнена структура запропонованого методу представлена на рис. 2.4

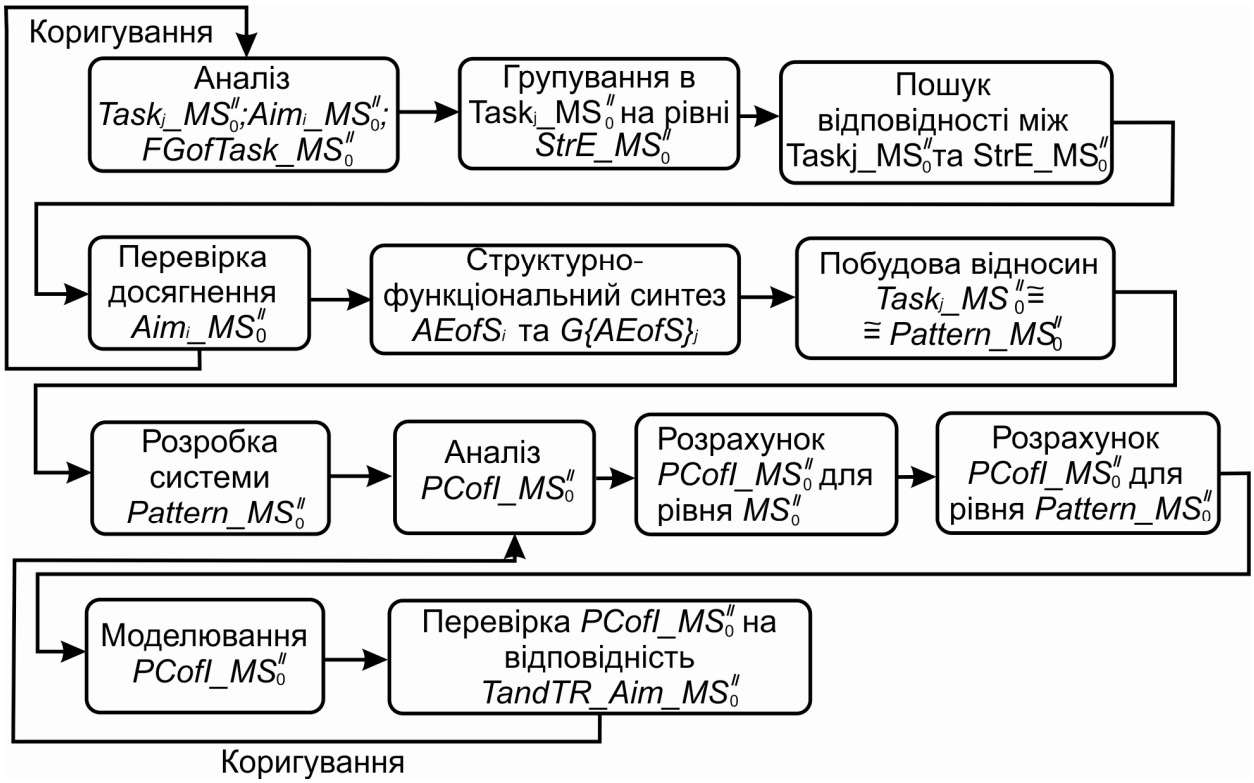


Рисунок 2.4 – Узагальнена структура методу прийняття рішень на організаційно-технічному етапі

2.3.3 Розробка методу прийняття рішень на атомарному рівні

Сучасні організаційно-технічні виробничі об'єкти на базі CPPS – це складна метасистема, яка має на нижньому рівні декомпозиції (рівень $AEofS_i$ на рис. 2.1) множини технічних приладів (ПЛК, датчики, верстати з ЧПУ, і т.д), здатних вирішити елементарні завдання, сукупність яких задовольняє вимогам ТЗ на CPPS. Наочним прикладом можуть виступити: верстат з ЧПУ, транспортний робот або робот-маніпулятор, отже можна відзначити, що елементарна задача – це виконання технологічної операцій (основної, допоміжної, транспортної і т.д.), або технологічного переходу. Тому дослідження атомарних елементів (*AEofS*), які можуть вирішувати елементарні завдання в даній методології позначимо через побудову і аналіз фізичних моделей.

У зв'язку з цим, для виконання прийнятих рішень на атомарному рівні декомпозиції «фізичного і імітаційного моделювання елементарних завдань» необхідно проводити відразу після прийняття рішень по *Aim* елементів

об'єкта. Далі йде аналіз *Aim* для *AEofS* і її *TrandTR*, в ході якого визначається галузі фізики, до яких відносяться процеси в технічній системі по кожній елементарній *Aim*.

Будується фізична модель процесу по кожному атомарному елементу, з визначенням граничних умов, які представлені в специфікації та тактико-технічних вимогах, для досягнення елементарної мети при рішенні елементарних завдань. Вивчається мета атомарного елемента (*Aim_AEofS*), його технічні вимоги (*TrandTR_AEofS*) на даному рівні процесу керування та проводиться ідентифікація фізичної моделі процесу, який протікає всередині *AEofS*. Проводиться ряд експериментів за допомогою імітаційного, а потім і фізичного моделювання, за результатами якого уточнюються параметри *AEofS*. Таким чином отримується масив даних, який містить технічні характеристики фізико-інформаційної моделі атомарного елемента (*PCofI_PIM_AEofS*). Ґрунтуючись на запропонованому методі розробнику необхідно розробити фізико-інформаційну модель (*PIM*) для всіх *AEofS* і елементарних *Aim*. Узагальнена структура методу прийняття рішень на атомарному етапі процесу управління розробкою CPPS представлена на рис. 2.5.

Отже при підвищенні рівня розробки можна об'єднати елементарні *Aim* в систему групових цілей (*G{Aim}*), що дозволить об'єднати фізико-інформаційні моделі в групову систему ($PIM\_G\{AEofS\}_j$). Для зручності візуалізації позначимо вузлами графа атомарні фізико-інформаційні моделі (*PM_AEofS*), а ребрами графа виступатимуть зв'язки (фізичні та / або інформаційні) між *AEofS*.

Для представлення даних зв'язків вводяться такі типи: фізичний зв'язок (*Pcom*), інформаційний зв'язок (*Icom*), фізико-інформаційний зв'язок (*PIcom*), які описуються закономірностями, властивими предметній області знань.

Системна $PIM\_G\{AEofS\}_j$, з її технічними характеристиками, дає можливість контролю правильність розробки, за допомогою технічних

характеристик *AEofS*, підданих декомпозиції і їх відповідників для досягнення графової мети на даному рівні розробки CPPS.

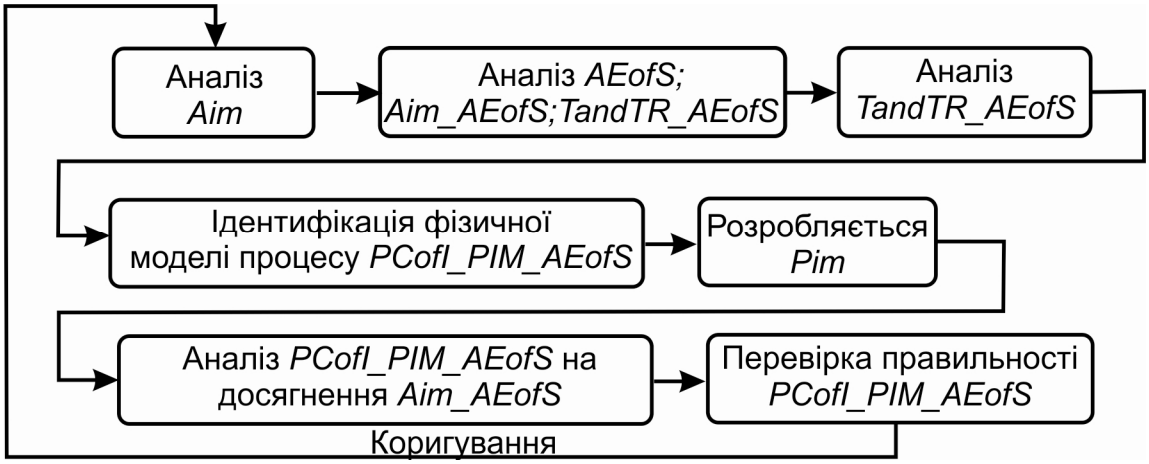


Рисунок 2.5 – Узагальнена структура методу прийняття рішень на атомарному етапі розробки CPPS

Фізичні моделі елементарних цілей є основою для розробки математичних моделей елементарних цілей (MM_Aim_i), елементарних завдань (E_Task_j) і інформаційного алгоритму роботи (LAW).

2.4 Розробка методів прийняття рішень на рівні кібернетичної складової

2.4.1 Розробка методу прийняття рішень на інфологічному етапі

Проводячи дослідження існуючих CPPS, які спеціалізуються на повній автоматизації складних технологічних процесів виробництва, велика увага приділяється її кібернетичній складовій, яка об'єднує всі етапи технології і керування в єдине інформаційне середовище. В рамках даних досліджень пропонується процес функціонування CPPS розглянути як завдання, які вирішуються на структурних елементах, для досягнення головної мети CPPS, зазначених в ТЗ. Зробімо припущення, що ці завдання мають вихідні ($InputIP_i$) і вихідні ($OutIP_j$) інформаційні параметри.

Розглядаючи завдання верхніх ієрархічних рівнів можна сказати, що це будуть тільки інформаційні потоки, а на нижніх рівнях це будуть параметри матеріальних потоків атомарних елементів ($AEofS_i$), або групи атомарних елементів ($G\{AEofS\}_j$). Виходячи з особливостей розробки і функціонування CPPS для керування процесами в складних організаційно-технічних виробничих об'єктах пропонуються наступні рішення на цьому етапі:

– проводиться аналіз і дослідження $InputIP_i$ і $OutIP_j$ для кожної $Task\_MS''_0$ системного рівня. Залежно від рівня MS''_0 параметри будуть носити «укрупнений» характер. Розробник визначає об'ємні і тимчасові характеристики $InputIP_i$ і $OutIP_j$, а також об'ємні і тимчасові характеристики $Pattern\_MS''_0$. Відстежується перетворення параметра $InputIP_i$ в $OutIP_j$ на базі рішення кожної $Task_j\_MS''_0$. Обираються математичні моделі і методи перетворення інформаційних параметрів, властивих предметній області $Task_j$, які адекватно описують даний процес;

– кожен $AEofS_i$ представляється у вигляді інформаційного перетворювача (*IC*). Обирається і розраховується швидкодія, обсяги оброблюваної інформації, функціональна точність і необхідність перетворення інформації;

– проводиться об'єднання $InputIP_i$ всіх завдань, що вирішуються на одному $AEofS_i$, у вигляді вхідного інформаційного каналу (*InputCanal*), а $OutIP_j$ у вигляді вихідного інформаційного каналу (*OutCanal*) з $AEofS_i$;

– об'єднуючи *InputCanal* і *OutCanal* різних $AEofS_i$ за ознаками системного зв'язку отримуємо, що вихідними параметрами $OutCanal_i\_Task_j$ будуть служити $InputCanal_{i+1}Task_j$, що дозволить розробнику сформувати системну інфологічну модель CPPS на системному рівні декомпозиції;

– проводиться аналіз параметричної сумісності всіх вихідних і вихідних параметрів $Task_j\_G\{AEofS_i\}$, розв'язуваних на *IC*, для перевірки правильності побудови системної інфологічної моделі;

– на останньому етапі рекомендується провести моделювання інфологічної моделі CPPS системного рівня за наступними мінімальним

критеріям: продуктивність, точність, надійність, а також обов'язково на відповідність $TrandTR\_Aim_i\_G\{AEofS_i\}$. Якщо отримані результати повністю задовольняють головній меті і ТЗ на CPPS, то всі прийняті рішення на інфологічному етапі є правильними. Узагальнена структура методу прийняття рішень на інфологічному етапі представлена на рис. 2.6.

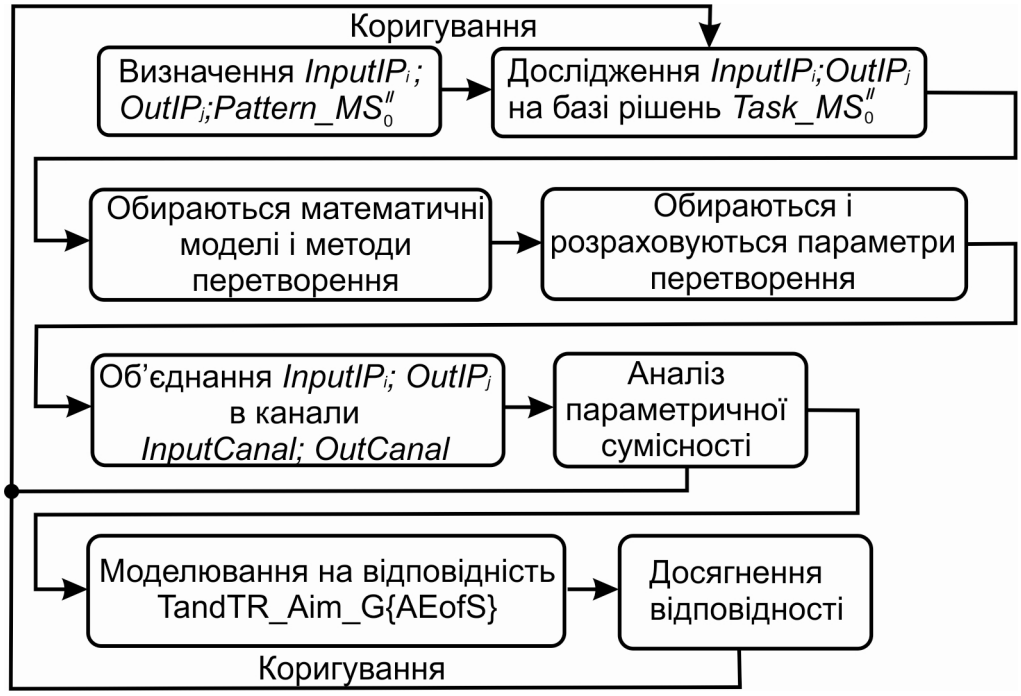


Рисунок 2.6 – Узагальнена структура методу прийняття рішень на інфологічному етапі

2.4.2 Розробка методу прийняття рішень на інформаційному етапі

Розробка прийняття рішень на інформаційному етапі є одним з важливих етапів керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS, який забезпечує функціонування та інформаційну взаємодію між інфологічим і алгоритмічним рівнями. На базі даного етапу розробником приймаються рішення, які напряму впливають на адекватність і якість розробки всієї кібернетичної складової CPPS. Даний рівень забезпечує функціонування і розвиток інформаційного простору CPPS, а також засобів інформаційної взаємодії не тільки на кожному рівні архітектурно-логічної

моделі, але і несе в собі рішення щодо візуалізації і відображення (HMI, GUI) всієї необхідної інформації.

На даному етапі необхідно реалізувати всі інформаційні потоки на кожному рівні щодо забезпечення досягнення головної мети розробки CPPS, відповідно до вимог зазначених в ТЗ. Як результат, в рамках даного дослідження, пропонується на даному етапі зробити декомпозицію методів «знизу-вгору», що обумовлено наступними завданнями:

– розробник повинен володіти повними технічними характеристиками кожного атомарного елемента системи ($PCofI\_AEofS_i$), з точки зору його інформаційної складової, а також кількості $OutIP_j$, $InputIP_i$, $PCofI\_OutIP_j$, $PCofI\_InputIP_i$;

– робочий алгоритми вирішення елементарних завдань базується на всіх $AEofS_i$, які досягають $Aim_i\_AEofS_i$ для кожного атомарного елемента;

– інформація вхідних $InputCanal\_AEofS_i$ і вихідних $OutCanal\_AEofS_i$ каналів забезпечує зв'язки всередині групової $G\{AEofS_i\}$, що дозволяє досягти $Aim_i\_G\{AEofS_i\}$ і т.д.

Виходячи з вищеперерахованих завдань в даному дослідженні пропонується наступний метод ухвалення рішень на даному етапі:

– проводиться аналіз $OutIP_j$, $InputIP_i$, $PCofI\_OutIP_j$, $PCofI\_InputIP_i$, для кожного $AEofS_i$;

– перевіряють і моделюють інформацію вхідних $InputCanal\_AEofS_i$ і вихідних $OutCanal\_AEofS_i$ каналів $AEofS_i$;

– $AEofS_i$ об'єднують у необхідну групу $G\{AEofS_i\}$ атомарних елементів $AEofS_i$, які відповідають $Task_j\_G\{AEofS_i\}$ і дозволяють досягти $Aim_i\_G\{AEofS_i\}$, а також перевіряють інформаційну сумісність $PCofI\_G\{AEofS_i\}$ всередині кожної $G\{AEofS_i\}$;

– проводиться моделювання досягнення вирішення задач, або завдань для кожної $Aim_i\_G\{AEofS_i\}$ і перевіряється досягнення правильності рішення, точності і швидкодії необхідних (Sub_S_k) на вході наступного рівня;

– за отриманими результатами моделювання розробник приймає рішення про успішність правильності побудови інформаційної структури рівня Sub_S_k;

– використовуючи отримані результати розробник синтезує послідовність інформаційних зв'язків елементів рівня Sub_S_k для досягнення вимог до вхідної інформації на рівень MS''_0, яка є необхідною для досягнення $Aim_i\_MS''_0$, $Task_j\_MS''_0$ на даному рівні розробки;

– на останньому рівні розробнику необхідно розробити всі інформаційні зв'язки між елементами MS''_0, врахувати всі канали зв'язків і їх технічні характеристики, що в сумі дасть змогу досягти головної мети розробки CPPS, з урахуванням вимог ТЗ.

2.4.3 Розробка методу прийняття рішень на алгоритмічному етапі

Алгоритмічний етап є одним з важливих складових при розробці CPPS, в ході якого, на базі отриманих результатів, розробляється комплекс програмного забезпечення (ПЗ) (від програм управління ПЛК, SCADA систем до ERP і MES), який в сукупності повинен забезпечити повний рівень автоматизації на кожному етапі виробництва. Розробник повинен повністю розробити алгоритми роботи на кожному рівні і забезпечити інформаційну сумісність кожного елемента, від атомарного до елементів візуалізації і прийняття рішень на найвищому рівні ієрархії, при цьому не допустити втрати інформаційних каналів або їх несумісності і досягти головної мети розробки CPPS, заданої в ТЗ. Для опису прийняття рішень на алгоритмічному етапі, в даному дослідженні, пропонується вербально-формульний опис послідовності дій.

Позначимо під алгоритмом функціонування CPPS наступну послідовність виконання: $Task_j\_MS''_0$ на $StrE\_MS''_0$, з урахуванням $InputCanal\_MS''_0$ і $OutCanal\_MS''_0$, для виконання $Aim_i\_MS''_0$ функціонування CPPS на даному рівні розробки. Представимо алгоритм функціонування у вигляді наступного виразу:

$$AF\_MS_0'' = f(Task_j\_MS_0'', StrE\_MS_0'', InputCanal\_MS_0'', OutCanal\_MS_0'', t), \quad (2.4)$$

де $AF\_MS''_0$ – алгоритм функціонування CPPS на рівні MS_0'';

$Task_j\_MS''_0$ – завдання на рівні MS''_0;

$StrE\_MS''_0$ – структурні елементи на рівні MS''_0;

$InputCanal\_MS''_0$ і $OutCanal\_MS''_0$ – вхідні і вихідні системні канали рівня MS''_0;

t – час.

На початковому етапі розробки $AF\_MS''_0$ проводиться декомпозиція CPPS по:

– $Aim_i\_MS''_0$ (мета на рівні MS''_0);

– $PCofI\_Aim_i\_MS''_0$ (технічні характеристики цілі на рівні MS''_0);

– $Task_j\_MS''_0$ (завдання на рівні MS''_0);

– $PCofI\_Task_j\_MS''_0$ (технічні характеристики завдань на рівні MS''_0);

– $StrE\_MS''_0$ (структурні елементи рівня MS''_0);

– $PCofI\_StrE\_MS''_0$ (технічні характеристики елементів рівня MS''_0);

– $InputCanal\_MS''_0$ і $OutCanal\_MS''_0$ (вхідні та вихідні системні канали рівня MS''_0 і зв'язки між ними);

– $PCofI\_InputCanal\_MS''_0$ і $PCofI\_OutCanal\_MS''_0$ (технічні характеристики вихідних і вихідних каналів рівня MS''_0).

Основною і єдиною метою розробки є синтез алгоритму функціонування (AF) досягнення $Aim_i\_MS''_0$ на рівні MS''_0. В даному дослідженні пропонується наступна послідовність рішень:

– аналізується початковий стан: MS''_0, $StrE\_MS''_0$, $InputCanal\_MS''_0$, $OutCanal\_MS''_0$ і по $PCofI\_StrE\_MS''_0$, $PCofI\_InputCanal\_MS''_0$, $OutCanal\_MS''_0$ і визначається їх готовність до функціонування;

– аналізується $Aim_i\_MS''_0$, $PCofI\_Aim_i\_MS''_0$ та їх взаємодія для досягнення генеральної мети розробки CPPS, яка вказана в ТЗ. На базі аналізу розробник будує граф цільової моделі;

– проводиться аналіз $Task_j\_MS''_0$ і визначається послідовність досягнення $Aim_i\_MS''_0$;

– перевіряється сумісність $PCofI\_Aim_i\_MS''_0$ і $PCofI\_StrE\_MS''_0$ при послідовному їх проходженні по структурних елементах;

– аналізується і перевіряється сумісність $PCofI\_InputCanal\_MS''_0$ та $PCofI\_OutCanal\_MS''_0$ з $PCofI\_StrE\_MS''_0$;

– розробляється послідовність виконання $Task_j\_MS''_0$, на базі $StrE\_MS''_0$, з урахуванням $InputCanal\_MS''_0$ та $OutCanal\_MS''_0$ і просторових характеристик $PCofI\_Task_j\_MS''_0$, $PCofI\_StrE\_MS''_0$, $PCofI\_InputCanal\_MS''_0$ і $PCofI\_OutCanal\_MS''_0$;

– будується граф алгоритму функціонування рівня MS''_0, де вузлами виступають $Task_j\_MS''_0$. Зв'язком між $Task_j\_MS''_0$ виступають $InputCanal\_MS''_0$ та $OutCanal\_MS''_0$, які будуть ребрами даного графа. Таким чином розробник отримує графову модель алгоритму функціонування CPPS даного рівня.

– розробник проводить аналіз графу алгоритму функціонування, за допомогою статичного моделювання, за критеріями ефективності розроблювального CPPS рівня MS''_0 і перевіряє досягнення $Aim_i\_MS''_0$;

– використовуючи методи імітаційного моделювання розробник перевіряє розроблену графову модель алгоритму функціонування CPPS рівня MS''_0 під навантаженням, якщо результати моделювання задовольняють умовам головної мети і вимогам ТЗ, прийняті рішення по розробці графової моделі вважаються вірними.

Розроблений метод дає можливість комплексувати рішення за цільовою, функціональною, інформаційною, алгоритмічною сумісністю з метасистемою, яка представляє собою складну CPPS.

Розглядаючи CPPS, як складну багаторівневу метасистему, пропонується використовувати вищепредставлені вербально-формульні описи розробки рішень на алгоритмічному етапі для всіх рівнів декомпозиції і розглядати їх як певну кількість підсистем (Sub_S_k), за умови що $MS''_0 \subset Sub\_S_k$. Отже з цього можна визначити, що прийняті розробником рішення здійснюються так

само як за цільовою, функціональною організаційно-технічною структурами на інфологічному і алгоритмічному етапах. Ґрунтуючись на (2.4) можна провести декомпозицію $Aim_i\_MS''_0$ на підцілі $Aim_i\_Sub\_S_k$ з побудовою графа алгоритму $Aim_i\_MS''_0$ на алгоритмічному етапі. Для цього розробник може уявити $Aim_i\_MS''_0$ як:

$$AF\_Aim_i\_MS''_0 = f(Aim_i\_Sub\_S_k, t), \tag{2.5}$$

де $AF\_Aim_i\_MS''_0$ – алгоритм досягнення мети на рівні MS''_0;

$Aim_i\_Sub\_S_k$ – мета на рівні підсистем Sub_S_k;

t – час.

Для цього розробнику необхідно знати всі $PCofI\_Aim_i\_Sub\_S_k$, розрахунок критеріїв ефективності кожної складової, надійність і результати системного моделювання всіх прийнятих рішень на кожному підсистемному рівні. Так само можна уявити алгоритмічний опис і на функціональному етапі кожної підсистеми:

$$AF\_Aim_i\_Sub\_S_k = f(Task_j\_Sub\_S_k, t), \tag{2.6}$$

де $AF\_Aim_i\_Sub\_S_k$ – алгоритм досягнення мети підсистеми Sub_S_k;

$Task_j\_Sub\_S_k$ – завдання підсистеми Sub_S_k;

t – час.

Розробник повинен провести розрахунок основних критеріїв ефективності $PCofI\_Aim_i\_Sub\_S_k$ за розробленим алгоритмічним описом і на функціональному етапі, за допомогою моделювання, довести правильність прийнятих рішень, які відповідають головній меті CPPS.

На організаційно-технічному етапі розробник визначає структурні елементи підсистеми ($Pattern\_Sub\_S_k$), аналізує і розраховує

PCofI_Pattern_Sub_S$_k$, на базі яких будує структурну модель системи підсистем:

$$AF\_Patten\_Sub\_S_k = f(Patten\_Sub\_S_k, t), \qquad (2.7)$$

де *AF_Pattern_Sub_S*$_k$ – алгоритм моделі підсистеми рівня *Sub_S*$_k$;

Pattern_Sub_S$_k$ – організаційно технічна структура рівня *Sub_S*$_k$;

t – час.

Розробник, на базі (2.7), проводить системне моделювання правильності прийнятих рішень, які показують досягнення необхідних цілей MS''_0, за допомогою сукупності *Sub_S*$_k$, на базі їх *PCofI_Pattern_Sub_S*$_k$ і з урахуванням *PCofI_InputCanal_Sub_S*$_k$, *PCofI_OutCanal_Sub_S*$_k$.

Для забезпечення керування розробкою кібернетичної складової CPPS, в даному дослідженні, пропонується наступний метод прийняття рішень на інфологічному етапі:

– розробником аналізуються і визначаються підсистемні канали *InputCanal_Sub_S*$_k$ та *OutCanal_Sub_S*$_k$;

– виділяються і аналізуються необхідні *PCofI_InputCanal_Sub_S*$_k$, *PCofI_OutCanal_Sub_S*$_k$;

– перевіряються *PCofI_InputCanal_G{AEofS}*$_j$ і *PCofI_OutCanal_G{AEofS}*$_j$ кожної групи;

– будується інфологічна модель системи на базі:

$$\begin{aligned} &AF\_InputCanal\_Sub\_S_k = \\ &f(InputCanal\_G\{AEofS\}_j, PCofI\_InputCanal\_G\{AEofS\}_j, t) \end{aligned}, \qquad (2.8)$$

$$\begin{aligned} &AF\_OutCanal\_Sub\_S_k = \\ &f(OutCanal\_G\{AEofS\}_j, PCofI\_OutCanal\_G\{AEofS\}_j, t) \end{aligned}, \qquad (2.9)$$

де $AF\_InputCanal\_Sub\_S_k$ – алгоритм функціонування *InputCanal* підсистеми рівня Sub_S_k;

$InputCanal\_G\{AEofS\}_j$ – вхідний канал $G\{AEofS\}_j$ з притаманними йому каналами рівня атомарних елементів $G\{AEofS\}_j$;

$PCofI\_InputCanal\_G\{AEofS\}_j$ – технічні характеристики вхідного каналу групи атомарних елементів $G\{AEofS\}_j$;

$AF\_OutCanal\_Sub\_S_k$ – алгоритм функціонування *OutCanal* підсистеми рівня Sub_S_k;

$OutCanal\_G\{AEofS\}_j$ – вихідний канал $G\{AEofS\}_j$ з притаманними йому каналами рівня атомарних елементів $G\{AEofS\}_j$;

$PCofI\_OutCanal\_G\{AEofS\}_j$ – технічні характеристики вихідного каналу групи атомарних елементів $G\{AEofS\}_j$;

t – час.

Після побудови інфологічної моделі розробником розраховується критерій ефективності підсистеми каналів і проводиться моделювання правильності побудови інфологічної структури рівня Sub_S_k.

– на останньому етапі розробником CPPS будується алгоритм функціонування на підсистемному рівні MS''_0.

В рамках даних досліджень пропонуються наступні рішення:

$$\begin{aligned} AF\_MS''_0 = f(&Aim_i\_Sub\_S_k, StrE\_Sub\_S_k, \\ &OutCanal\_Sub\_S_k, InputCanal\_Sub\_S_k, t) \end{aligned}, \qquad (2.10)$$

де $AF\_MS''_0$ – алгоритм функціонування підсистеми рівня MS''_0;

$Aim_i\_Sub\_S_k$ – досягнення мети підсистеми рівня Sub_S_k;

$StrE\_Sub\_S_k$ – вхідні канали рівня Sub_S_k;

$OutCanal\_Sub\_S_k$ – вихідні канали рівня Sub_S_k;

$InputCanal\_Sub\_S_k$ – вхідні канали рівня Sub_S_k.

t – час.

Після побудови $AF\_MS''_0$ обов'язково проводиться його дослідження з метою виявлення досягнення $Aim_i\_MS''_0$, шляхом моделювання за просторово-тимчасовими характеристиками і критеріям ефективності при навантаженні.

Виходячи із запропонованих рішень (2.1)–(2.10) розробник CPPS на підсистемному рівні декомпозиції, проводить комплексування в єдину систему по всім рівням, з урахуванням технічних характеристик їх підсистемних рівнів, які повинні відповідати технічними умовам мети та завданню даного рівня. Варто звернути увагу, що запропоновані методи прийняття рішень можна поділити на два типи:

– моделювання декомпозованих елементів нижнього рівня на відповідні елементи вищого рівня з урахуванням їх технічних характеристик;

– побудова моделі MS'_0, як сукупності елементів підсистемних рівнів, від першого до четвертого, для перевірки досягнення головної мети розробки CPPS та відповідності ТЗ.

Ґрунтуючись на вище запропонованих методах прийняття рішень, для отриманням на кожному етапі відповідних тактико-технічних характеристик досягнення задач і мети, розробнику необхідно провести декомпозицію від п'ятого до першого атомарного рівня процесу керування розробкою CPPS. Відповідні тактико-технічних характеристики повинні задовольняти вимогам даного рівня і прагнути до досягнення головної мети і параметрів, зазначених в ТЗ.

2.4.4 Розробка методу прийняття рішень на рівні математичного опису елементарних завдань

Виходячи з запропонованої архітектурно-логічної моделі керування процесами в складних організаційно-технічних виробничих об'єктах (рис 2.1), можна бачити, що рішення блоку «Математичного опису елементарних завдань» впливають на кібернетичну складову CPPS.

Визначимо під математичною моделлю атомарного елемента $AEofS_i$ математичне представлення фізичного процесу, що протікає в ньому, в

термінах математичних рівнянь і їх послідовності різних балансів фізичних моделей. Це пов'язано з тим, що математичний опис елементарних завдань, розв'язуваних на атомарних елементів ($AEofS_i$), залежить від апаратних складових (ПЛК, датчиків, одноплатних комп'ютерів і т.д.).

Результати математичних моделей можуть бути використані розробником при проєктуванні технологічного процесу, який напряму може бути пов'язаний з інформаційною складовою у вигляді кодів для програм з ЧПУ, які забезпечують вимоги даної задачі.

Позначимо елементарну задачу через найпростіший одиничний перехід (основний, допоміжний, транспортний і т.д.), який є невід'ємною частиною технологічної операції. З чого можна з упевненістю заявити, що математичні моделі, які використовують фізичну природу свого процесу, можуть бути представлені у вигляді математичного уявлення, яке можна перевірити за допомогою математичного моделювання з використанням пакетів MatCAD, MatLab і т.д.

Результати моделювання показують можливість досягнення мети, що поставлена перед елементарними завданнями, а отримані математичні описи виступатимуть основою для розробки алгоритму і програм керування, моніторингу перебігу технологічного процесу в часі. Ґрунтуючись на цьому пропонується наступний метод прийняття рішень на даному блоці:

– на першому етапі розробник повинен провести аналіз і дослідити результати фізичного та імітаційного моделювання елементарних завдань (Elm_Task_j) на наявність досягнення елементарної мети (Elm_Aim_i);

– за отриманими результатами розробник виявляє деякі фізичні закономірності, при яких об'єкт досягає заданих вимог Elm_Task_j;

– на базі виявлених закономірностей аналізуються і підбираються математичні апарати, від найпростіших (системи лінійних рівнянь) до найскладніших (інтегро-диференціальних рівнянь). Розробнику необхідно підібрати ефективний математичний апарат, який відповідає вимогам точності, надійності, часу виконання і збіжності результатів;

– проводиться ідентифікація математичної моделі на параметри фізичної моделі об'єкта, проводиться моделювання при різних допустимих значеннях і вимог тактико-технічних характеристик, що дає можливість визначити фізичний діапазон вхідних і вихідних параметрів;

– розробник ідентифікує всі математичні моделі і виявляє фізичний діапазон їх вхідних і вихідних параметрів;

– ґрунтуючись на ідентифікованих математичних моделях, розробник формує список апаратних атомарних елементів ($AEofS_i$), які за технічними характеристиками дозволяють вирішувати Elm_Task_j, при цьому на один $AEofS_i$ може вирішуватися кілька Elm_Task_j, кожна з яких досягає Elm_Aim_i;

– для кожного обраного $AEofS_i$ розробляється алгоритм вирішення елементарних завдань (рис. 2.1, продовження), які в майбутньому дозволяють розробити інформаційну структуру і алгоритм роботи ПЗ, в залежності від типу і вимог $AEofS_i$.

2.5 Технологія процесу керування розробкою кібер-фізичних виробничих систем

Запропоновані в підрозділах 2.2-2.4 методи декомпозиції основних етапів процесу керування розробкою CPPS для складних організаційно-технічних виробничих об’єктів дозволяють прийняти рішення на кожному кроці, але при цьому не показують загальну технологію підходу до вирішення мети даного дослідження, а лише вирішують завдання розробки на етапах.

Отже, наступному кроком, в цій роботі, є розробка технології керування процесом розробки CPPS. Цей крок є логічно пов'язаною послідовністю ухвалення рішень розробником, на основі методів декомпозиції, у відповідності з древом і етапами розробки (рис. 2.7.).

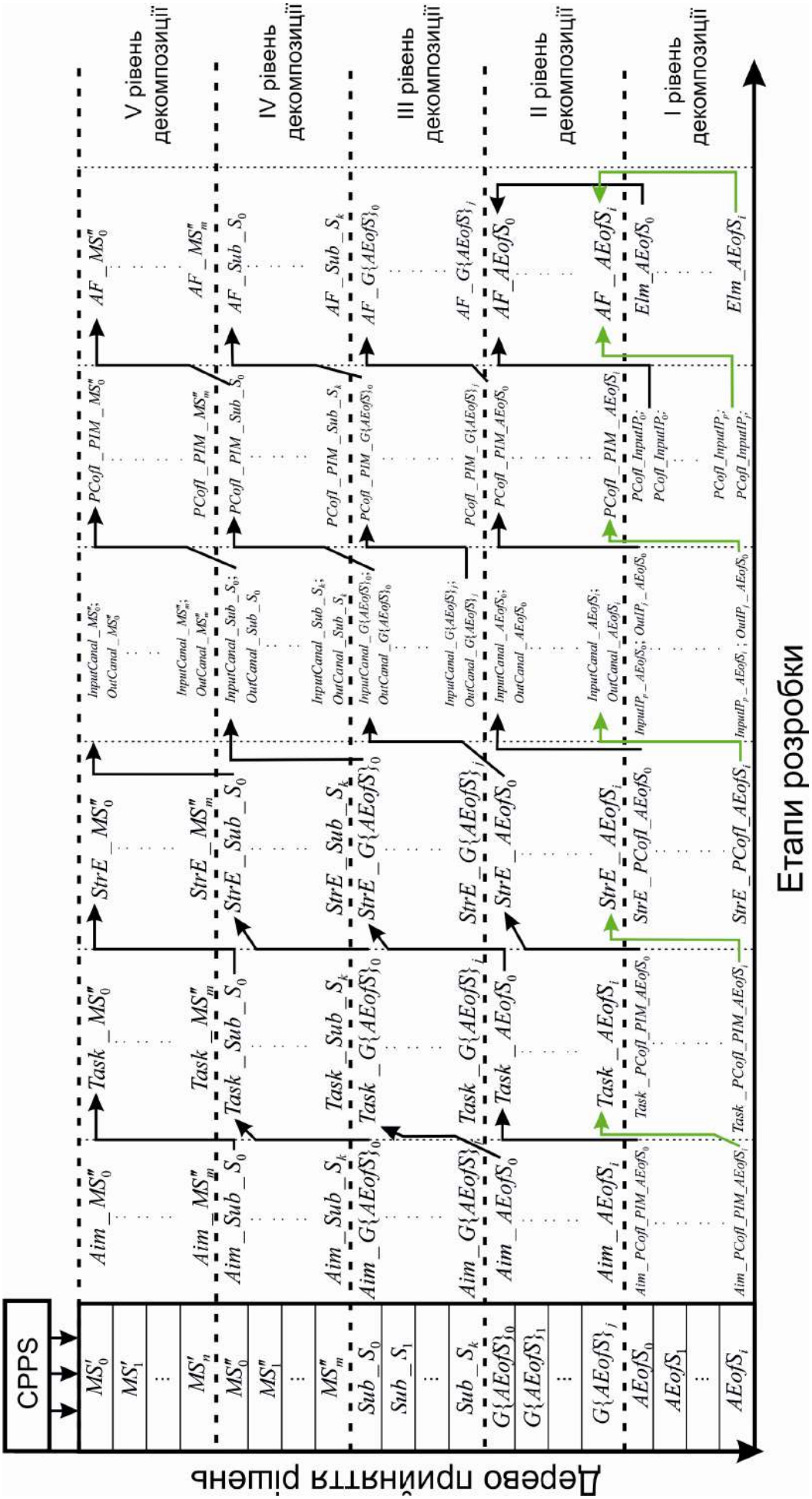


Рисунок 2.7 – Технологія процесу керування розробкою CPPS

В основі запропонованої технології лежить принцип «зверху-вниз» і «зліва-направо». Такий вибір обумовлено складною ієрархічною структурою будь-якої сучасної CPPS. Отже на початковому етапі необхідно вирішити завдання системного підходу до технології процесу керування розробкою CPPS [196].

Ґрунтуючись на даному припущенні розробнику необхідно визначити п'ятий рівень ієрархії CPPS, який ґрунтується на критеріях головної мети і вимогах ТЗ.

Вибір критеріїв визначення п'ятого рівня декомпозиції, в ході проведення системного аналізу, проводиться на базі методу запропонованого в підрозділі 2.2.

Розробник виділяє необхідний рівень ієрархії процесу керування розробки і проводить декомпозицію метасистеми MS'_0 на мультисистеми MS''_0 і фіксує цей рівень.

Це дозволяє приймати рішення на даному рівні в суворій послідовності, використовуючи принцип «зліва-направо», у відповідності з запропонованими етапами (рис. 2.1) та врахуванням послідовності рішень від m до $m+n$. Отже структуру технології процесу керування розробки CPPS можна візуалізувати у вигляді систематизованого дерева рішень на етапі декомпозиції MS'_0, який представлений на рисунку 2.8.

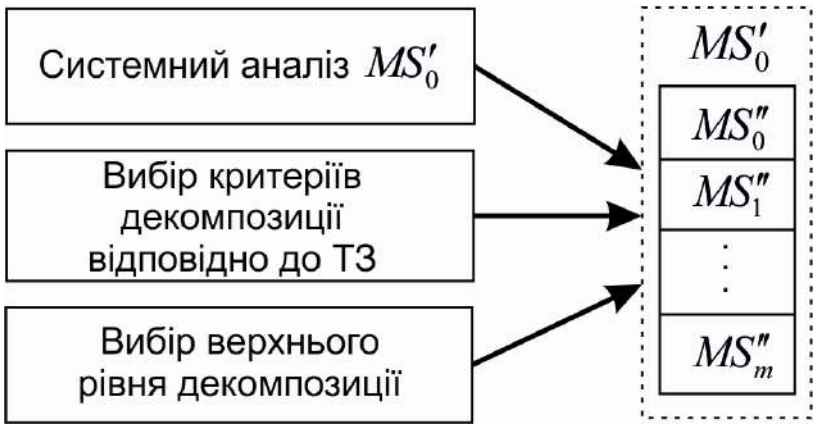


Рисунок 2.8 – Древо рішень на етапі декомпозиції MS'_0

Після виділення рівня декомпозиції розробки MS'_0, розробник проводить декомпозицію MS'_0 на $Aim_i\_MS''_0$, на базі головної мети ТЗ. Наступним кроком є виявлення і обґрунтування всіх $PCofI\_MS''_0$, на базі яких здійснюється побудова цільової моделі MS'_0 і проводитися перевірка правильності і оцінка ефективності прийняття рішень за допомогою динамічного моделювання, яке повинно показати досягнення головної мети розробки MS'_0.

У відповідності до запропонованої технології процесу керування розробки CPPS (рис. 2.7) розробник може перейти до наступного етапу – розробка функціональної моделі. На цьому етапі знаходиться відповідність для кожної мети ($Aim_i\_MS''_0$) свого завдання ($Task_j\_MS''_0$), або групи завдань. Для кожного завдання $Task_j\_MS''_0$ розробляються $PCofI\_Task_j\_MS''_0$, на базі яких будується функціональна модель досягнення $Aim_i\_MS''_0$. Дерево рішень декомпозиції на функціональному етапі представлене на рис. 2.9.

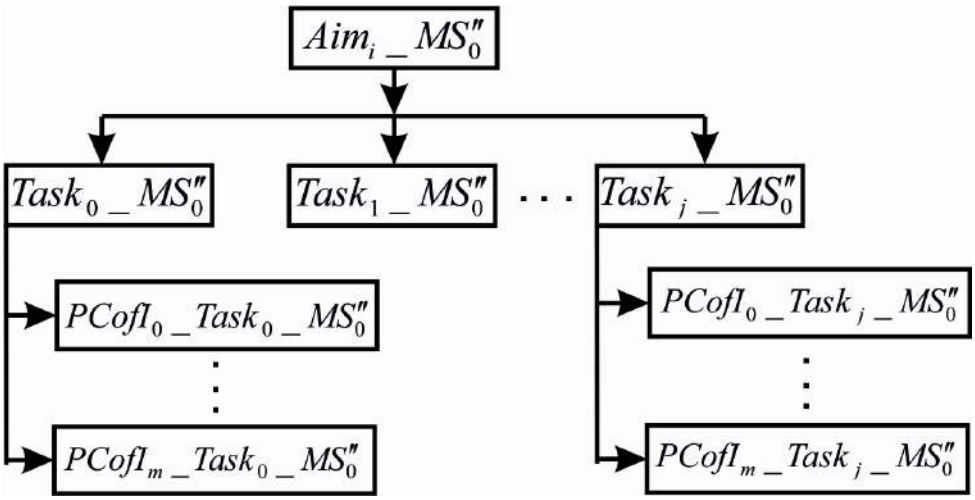


Рисунок 2.9 – Древо рішень декомпозиції на функціональному етапі

Методом системного моделювання на $Task_j\_MS''_0$ розробник проводить статистичне моделювання $PCofI\_Task_j\_MS''_0$ функціональних елементів MS''_0, які піддавались декомпозиції. Якщо результати моделювання відповідають головній меті MS'_0, то можна вважати, що рішення на функціональному етапі декомпозиції завдань і тактико-технічних характеристик, розроблені

адекватно і розробник може переходити до етапу організаційно-технічної розробки.

На цьому етапі розробник розробляє структурні елементи системного рівня $StrE\_MS''_0$, проводить дослідження і вибір їх $PCofI\_StrE\_MS''_0$, в залежності від вимог $PCofI\_Task_j\_MS''_0$. Будується системна структурна модель CPPS рівня MS''_0 і проводитися моделювання $PCofI\_StrE\_MS''_0$ для досягнення $Aim_i\_MS''_0$.

Дерево рішень декомпозиції на організаційно-технічному етапі представлене на рис. 2.10.

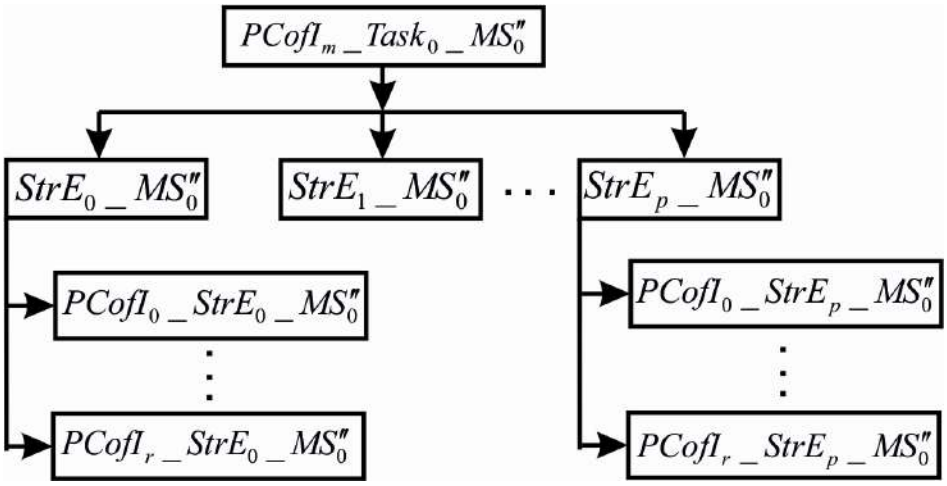


Рисунок 2.10 – Дерево рішень декомпозиції на організаційно-технічному етапі

Якщо результати моделювання задовольняють $PCofI\_StrE\_MS''_0$ і $Aim_i\_MS''_0$ – можна вважати, що рішення на даному рівні відповідають головній меті розробки CPPS і вимогам ТЗ. Отже розробник може приступити до наступного інфологічного етапу.

Ґрунтуючись на методі розробки, представленого в підрозділі 2.4.1, розробник проводить аналіз $StrE\_MS''_0$ і визначає $InputIP_i$ і $OutIP_j$ для кожного структурного елемента, що вимагає дослідження інфологічних перетворювачів параметрів $IC\_MS''_0$ і системних каналів зв'язків між ними

$InputCanal\_MS''_0$ і $OutCanal\_MS''_0$, в тому числі $PCofI\_IC\_MS''_0$, $PCofI\_InputCanal\_MS''_0$, $PCofI\_OutCanal\_MS''_0$.

На базі проведеного аналізу будується інфологічна модель CPPS системного рівня MS''_0 і проводиться моделювання на відповідність $PCofI\_IC\_MS''_0$, $PCofI\_InputCanal\_MS''_0$, $PCofI\_OutCanal\_MS''_0$ і $Aim_i\_MS''_0$. При відповідності $PCofI\_IC\_MS''_0$, $PCofI\_InputCanal\_MS''_0$ та $PCofI\_OutCanal\_MS''_0$ головній меті розробки MS'_0, розробник може вважати що прийняті рішення на даному етапі задовольняють ТЗ на CPPS. Дерево рішень декомпозиції на інфологічному етапі представлене на рисунку 2.11.

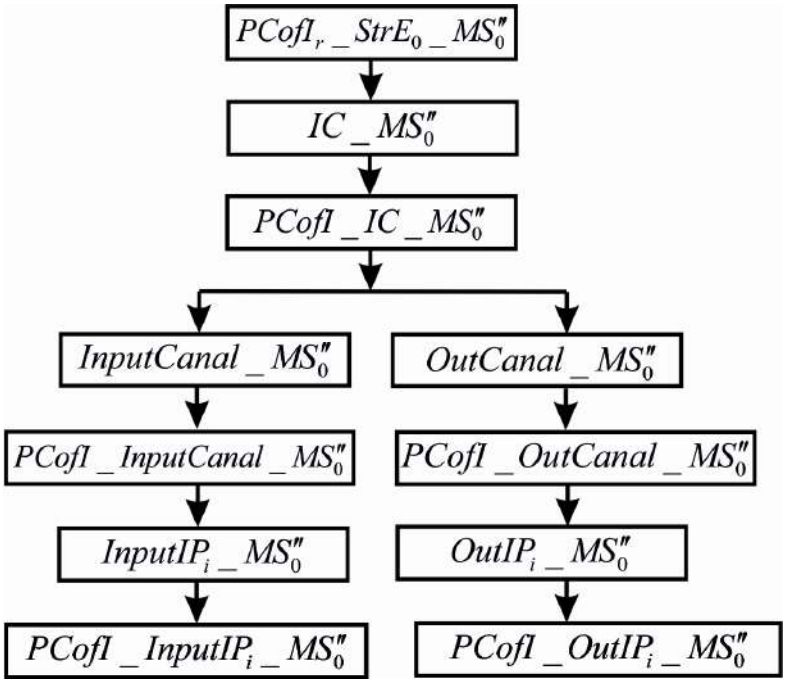


Рисунок 2.11 – Дерево рішень декомпозиції на інфологічному етапі

Ґрунтуючись на результатах попереднього етапу розробник може приступити до розробки інформаційної структури обраного рівня. Для цього, на базі результатів статичного моделювання $PCofI\_InputIP_i\_MS''_0$, $PCofI\_OutIP_i\_MS''_0$, $InputCanal\_MS''_0$, $OutCanal\_MS''_0$, проводитися розробка технічних характеристик фізико-інформаційної моделі даного рівня декомпозиції ($PCofI\_PIM\_MS''_0$). На його базі синтезуються послідовності інформаційних потоків, параметрів і зв'язків. Для перевірки досягнення

головної мети розробки MS'_0 і вимог ТЗ проводиться імітаційне моделювання під навантаженням. Дерево рішень декомпозиції на інфологічному етапі представлене на рис. 2.12.

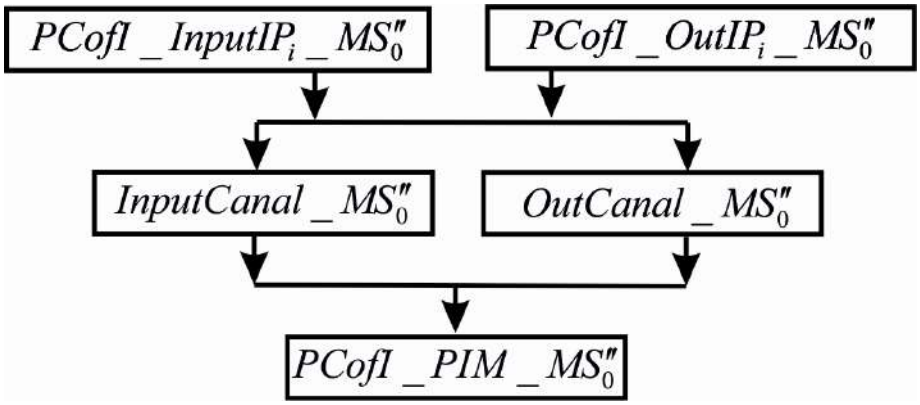


Рисунок 2.12 – Дерево рішень декомпозиції на інформаційному етапі

Базуючись на отриманих результатах розробник може приступити до розробки алгоритмічного етапу даного рівня декомпозиції. Завданням даного етапу є розробка алгоритму функціонування, на базі якого, в подальшому, проводиться розробка кібернетичної складової CPPS.

На базі методу, запропонованого в підрозділі 2.4.3, розробник проводить аналіз $PCofI\_PIM\_MS''_0$ і за (2.4) проводить розробку алгоритму функціонування рівня $AF\_MS''_0$. Для цього розробляються оператори Op_h алгоритму функціонування CPPS на даному рівні декомпозиції, проводиться розрахунок технічних і інформаційних характеристик $PCofI\_PIM\_Op_h\_MS''_0$ системного рівня $AF\_MS''_0$ для кожного $Op_h\_MS''_0$.

Використовуючи отримані результати розробник будує $AF\_MS''_0$ і проводить моделювання під навантаженням, для перевірки правильності прийнятих рішень. Дерево рішень декомпозиції на алгоритмічному етапі представлене на рис. 2.13.

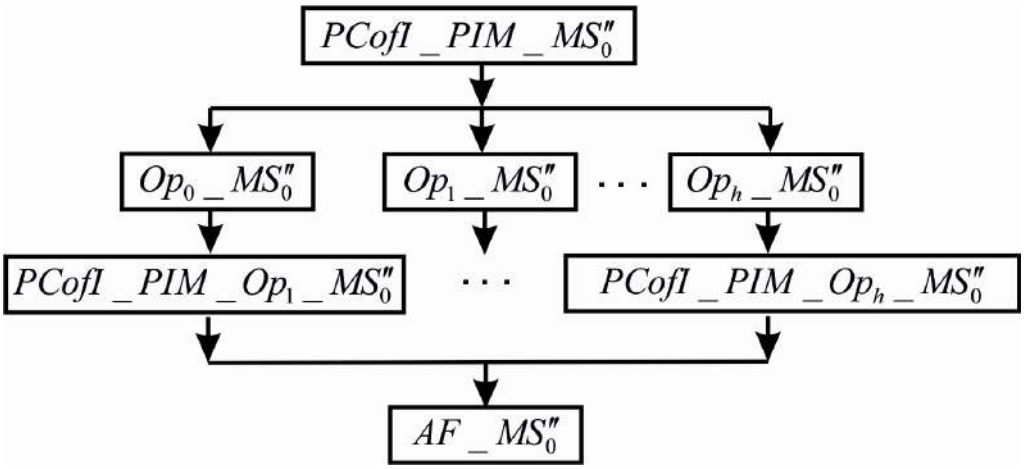


Рисунок 2.13 – Дерево рішень декомпозиції на алгоритмічному етапі

Якщо отриманий результат повністю задовольняють вимогам $Task_j\_MS''_0$ і завданням $Aim_i\_MS''_0$ головної мети CPPS, то можна вважати, що прийняті рішення декомпозиції на інформаційному етапі правильні, отже всі рішення на попередніх етапах процесу керування розробкою CPPS даного рівня декомпозиції є «життєздатними».

Для зручності розуміння всі запропоновані рішення декомпозиції рівня MS''_0 за етапами: стратегічному, функціональному, організаційно-технічному, інфологічному, інформаційному і алгоритмічному, можна уявити у вигляді узагальненого дерева рішень декомпозиції рівня MS''_0, яке представлене на рис. 2.14.

Ґрунтуючись на представленому дереві декомпозиції рівня MS''_0, можна припустити, що використовуючи методи, запропоновані в розділах 2.3–2.4, можна провести декомпозицію всіх рівнів розробки від Sub_S_k до $AEofS_i$ і забезпечити досягнення головної мети розробки CPPS для керування процесами в складних організаційно-технічних виробничих об'єктах.

Варто зауважити, що запропоновані методи декомпозиції дозволяють розробнику не тільки розробити структуру CPPS для керування процесами в складних організаційно-технічних виробничих об'єктах, але і забезпечити вибір необхідних атомарних елементів $AEofS_i$ (фізичної складової), на базі яких можливі рішення елементарних завдань і отримання математичного

опису кожного завдання, або групи, для розробки їх алгоритму функціонування роботи (кібернетичної складової).

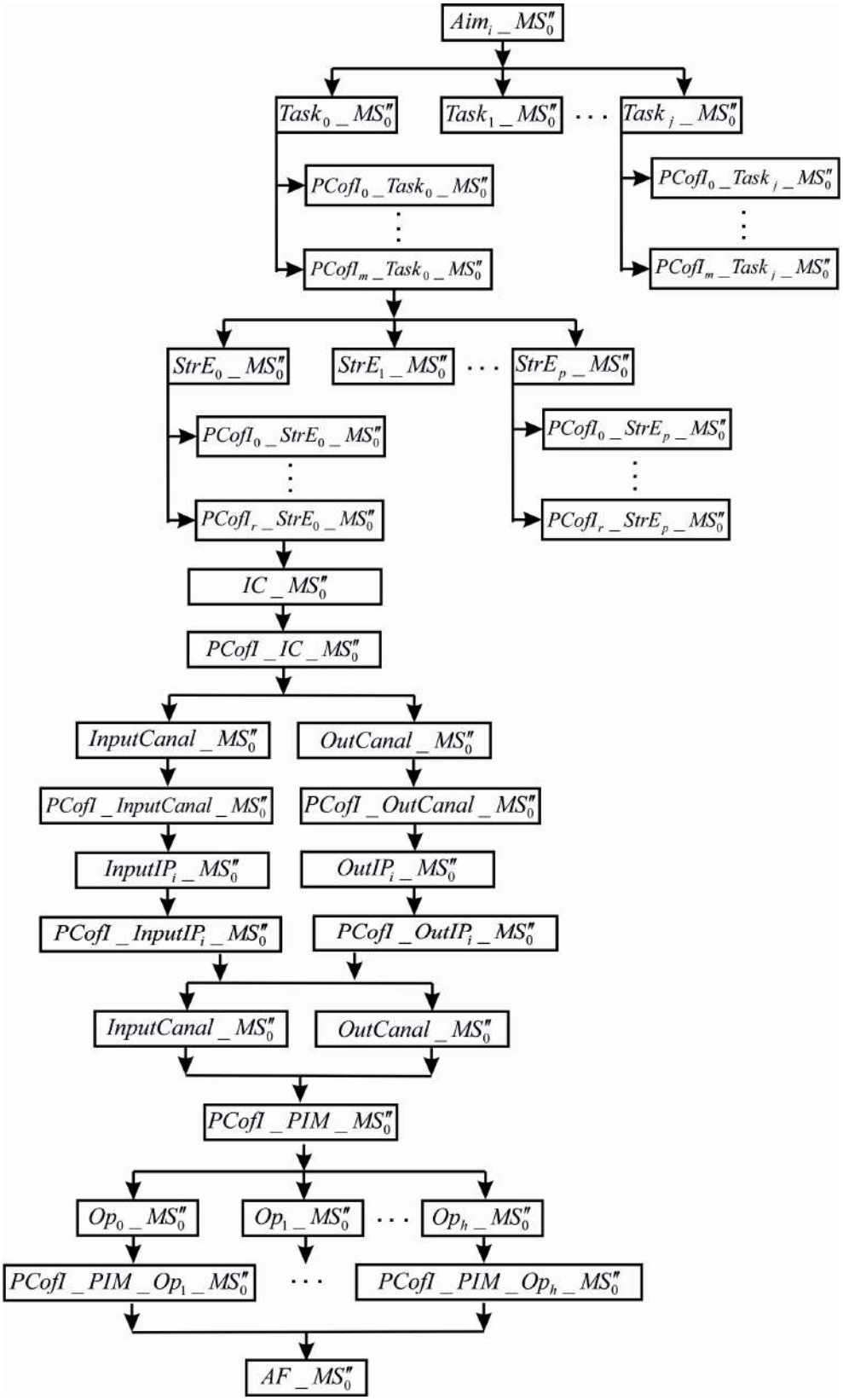


Рисунок 2.14 – Дерево рішень декомпозиції рівня MS''_0

Отримані рішення з розробки CPPS дають можливість провести імітаційне моделювання під навантаженням, а також оцінити тимчасові, якісні та функціональні параметри розроблюваної CPPS.

Висновки до розділу 2

1. В даному розділі розроблена архітектурно-логічна модель керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS, яка об'єднує в собі послідовність взаємопов'язаних етапів за стратегічною, фізичною і кібернетичною складовими. Запропонована п’ятирівнева декомпозиція архітектурно-логічної моделі. При цьому доведено можливість характеризувати 1-4 рівень як моносистему, а 5 рівень як метасистему, що дає можливість реалізувати дерево прийняття рішень.

2. Визначена структура кожного етапу архітектурно-логічної моделі керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS, яка містить логічно взаємопов'язану послідовність функціонального, організаційно-технічного етапів для фізичної складової та інфологічного, інформаційного та алгоритмічного етапів для кібернетичної складової.

3. Обґрунтовано необхідність використання стратегічного етапу на початковому етапі архітектурно-логічної моделі керування процесами в складних організаційно-технічних виробничих об'єктах. Запропонована формалізація всіх рівнів, доведена їх необхідність і достатність.

4. Запропоновані взаємопов'язані методи керування процесами в організаційно-технічних об'єктах, які дають можливість визначити і окреслити мету та завдання на кожному рівні декомпозиції, провести математичний опис елементарних завдань і їх фізичне або імітаційне моделювання досягнення мети рівня і головної мети розробки CPPS. Розроблена послідовність взаємопов'язаних методів процесу управління

прийняття рішень на всіх етапах розробки CPPS, що в сукупності дозволило реалізувати технологію «Digital Twins».

5. На базі архітектурно-логічної моделі запропонована технологія розробки кібер-фізичних виробничих систем, яка дозволяє представити їх структуру у вигляді логічно пов'язаної послідовності алгоритмів функціонування кожного рівня. Отримані результати, у вигляді алгоритмів функціонування, відкривають можливість проведення синтезу елементів алгоритму, що дозволить спростити її структуру, а отже мінімізувати час і складність розробки фізичної і кібернетичної складових, а також дає можливість реалізувати гнучкість процесу керування організаційно-технічним об'єктом.

Список джерел, які використано у даному розділі, наведено у повному списку використаних джерел [21], [190 –196].

3 РОЗРОБКА СИСТЕМНИХ МОДЕЛЕЙ КІБЕР-ФІЗИЧНИХ ВИРОБНИЧИХ СИСТЕМ

3.1 Групування методів розробки CPPS

Ґрунтуючись на твердженнях, що розробка кібер-фізичних систем для керування процесами в складних організаційно-технічних виробничих об'єктах, базується на застосуванні кількох приватних системних методах прийняття рішень, етапи яких запропоновано у 2 розділі даної роботи, можна відмітити, що при аналізі цих рішень існує деяка закономірність використання одного методу на різних рівнях розробки CPPS.

Для спрощення розробки системних моделей CPPS, пропонується провести таке групування даних методів:

– згрупувати методи декомпозиції Aim_i, $Task_j$ структур, алгоритмів функціонування верхнього рівня на елементи нижнього рівня;

– згрупувати методи розробки Op_h_AF, по $Task_j$ і структурам об'єкта *InputCanal*, *OutCanal* по структурам, $AEofS_i$ по $Task_j$, $Task_j$ по Aim_i, застосувавши метод трансформації можна бачити таку закономірність: цілі трансформуються в завдання, завдання в структуру, канали зв'язків, алгоритм функціонування об'єкта;

– згрупувати методи розрахунку *PCofI*, Aim_i, $Task_j$, структур *InputCanal*, *OutCanal*, Op_h;

– згрупувати методи розробки системної, стратегічної, функціональної, інфологічної, інформаційної та алгоритмічної моделей;

– згрупувати аналітичні методи ухвалення рішень чітко за етапами розробки.

Використання даних методів розробником CPPS для різних етапів розробки носить обов'язковий, закономірний характер. Але всі запропоновані методи мають відмінності в застосування і залежать від етапу розробки.

Проведемо аналіз першої групи методів і визначимо головні положення:

– дослідження і аналіз аналогів, прототипів і визначення основних властивостей, критеріїв, ознак – системний аналіз CPPS;

– виявлення складових частин CPPS проводиться за певними ознаками (стратегічною, функціональною, інфологічною і т.д.);

– визначення зв'язків відбувається між складовими частинами CPPS і всередині їх;

– декомпозиція CPPS на складові частини проводиться на базі принципу слабких зв'язків між його частинами.

Логічно, що для кожного CPPS керування процесами в складних організаційно-технічних виробничих об'єктах і його складових частин, зв'язки і властивості між ними різні і будуть верифіковані закономірностями тієї предметної області, до якої відносяться процеси, що відбуваються всередині. Отже, приватні методики визначення складових частин CPPS (вид, зв'язки, властивості, і т.д.) будуть визначені даною предметною областю знань, але при цьому метод декомпозиції для всіх буде один.

Проведемо дослідження групи трансформації, за результатами якого можна бачити можливість ізоморфізму, отже ліквідним буде такий запис:

$$Aim_i \cong Task_j \cong StrE_q \cong IC_v \cong Op_h .$$

Це дає можливість виділення елементів, груп, підсистем і систем та віднесення їх до об'єктів системного дослідження, що не є безпосередньо завданням розробника.

Згруповання методів розрахунку обумовлено їх тісним зв'язком з групою трансформації і її закономірністю для предметної області знань, тому визначають математичні закони, методи і способи розрахунку параметрів. Дана група вивчається фахівцями даної галузі і на базі їх рішень розробляється методика розрахунків для кожного випадку розробки CPPS.

Ґрунтуючись на вищевказаних основних вимогах, для успішної розробки CPPS, розробник повинен маніпулювати системними моделями які:

– описуються в графічній формі;

– мають можливість алгебраїчних перетворень для зручності їх моделювання.

Отже, проведений аналіз методів формалізації, побудови і перетворення структур і алгоритмів показав, що для опису структури та зв'язків, що підходять під вищеперераховані вимоги, можна використати теорію графів, а їх аналіз проводити на базі регулярних схем і мов.

3.2 Метод подання структурних системних моделей CPPS

Визначення 1. Системна модель – це графічне представлення, в якому вузли – об'єкти, відповідні етапам і рівням розробки CPPS, а ребра – канали зв'язків між ними. Базуючись на розроблених методах декомпозиції (2.1)–(2.4) і дереві рішень (рис. 2.1., рис. 2.7), пропонуються наступні уявлення системних моделей. Логічно, що для досягнення головної мети розробки CPPS необхідно досягти всі підцілі на всіх рівнях декомпозиції, тому для спрощення запису визначимо через предикат Ω – умову досягнення всіх цілей нижнього рівня $Aim_j\_MS''_0$, необхідних і достатніх для досягнення цілей рівня $Aim_i\_MS'_0$:

$$Aim_i\_MS'_0 \Rightarrow \underset{j}{\Omega}(Aim_j\_MS''_0), \tag{3.1}$$

де $Aim_i\_MS'_0$ – головна мета розробки CPPS у відповідності до ТЗ;

$Aim_j\_MS''_0$ – цілі на рівні декомпозиції MS''_0.

Ґрунтуючись на (3.1) можна провести декомпозицію цілі $Aim_j\_MS''_0$ на підцілі рівня $Sub\_S_k\_MS''_0$ у вигляді (3.2).

$$Aim_j\_MS''_0 \Rightarrow \underset{m}{\Omega}(Aim_m\_Sub\_S_k\_MS''_0), \tag{3.2}$$

де $Aim_m\_\ Sub\_S_k\_MS''_0$ – цілі на рівні декомпозиції $Sub\_S_k\_MS''_0$.

Проведемо декомпозицію цілі розробки по всьому дереву:

$$Aim_m\_Sub\_S_k\_MS''_0 \Rightarrow \Omega_l(Aim_l\_G\{AEofS\}_p\_MS''_0), \tag{3.3}$$

де $Aim_i\_G\{AEofS\}_p\_MS''_0$ – мета для групи атомарних елементів $G\{AEofS\}_p\_MS''_0$.

$$Aim_l\_G\{AEofS\}_p\_MS''_0 \Rightarrow \Omega_s(Aim_s\_AEofS_t\_MS''_0), \tag{3.4}$$

де $Aim_s\_AEofS_t\_MS''_0$ – мета атомарного елемента $AEofS_t$;

Проводячи аналіз (3.1)–(3.4) можна помітити, що головна мета розробки CPPS, є наслідком досягнення цілей на кожному рівні декомпозиції на кожному етапі розробки, отже можна з упевненістю стверджувати існування «спадковості» (→) цілей за принципом декомпозиції «зверху-вниз»:

$$\begin{aligned} &Aim_i\_MS'_0 \rightarrow Aim_j\_MS''_0 \rightarrow Aim_j\_Sub\_S_k\_MS''_0 \rightarrow \\ &\rightarrow Aim_l\_G\{AEofS\}_p\_MS'' \rightarrow Aim_s\_AEofS_t\_MS''_0 \end{aligned} \tag{3.5}$$

Ґрунтуючись на (3.5) і визначенні 1 можна уявити граф досягнення головної мети розробки CPPS, представлений на рис. 3.1, з урахуванням його багаторівневості, дозволяє враховувати вплив досягнення цілей одна на одну всередині одного рівня декомпозиції.

Виходячи з рис. 3.1 можна сказати, що розробник CPPS отримає набір системних моделей декомпозиції головної мета $Aim_i\_MS'_0$ на всіх рівнях розробки. Розмістивши кожен вузол Aim_i за відповідними $TandTR\_Aim_i$, розробник має можливість провести статистичне і динамічне моделювання під навантаженням на досягнення головної мети розробки CPPS.

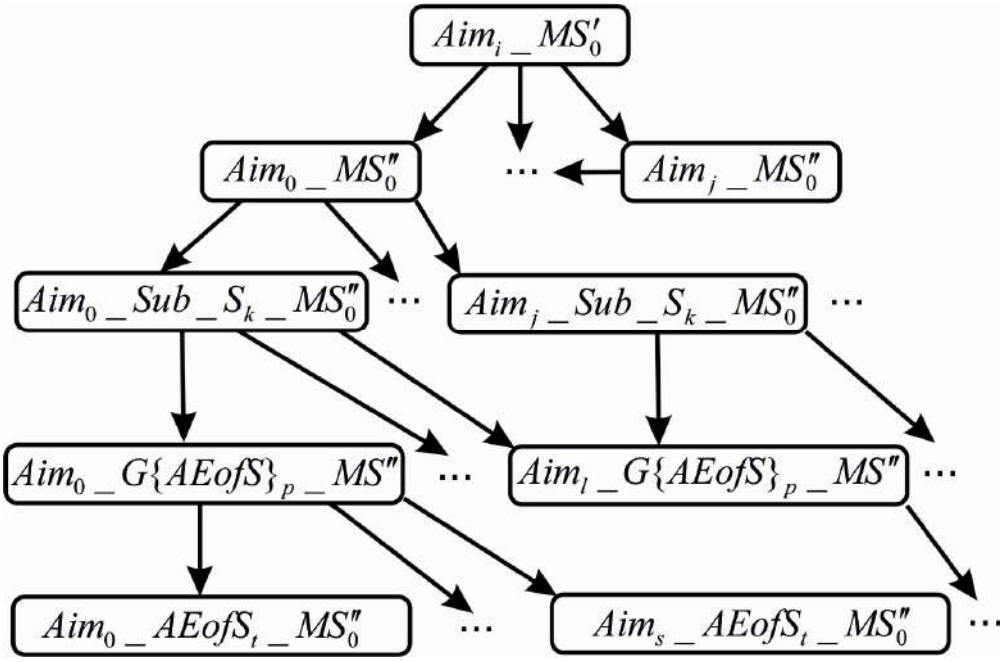


Рисунок 3.1 – Граф досягнення головної мети розробки CPPS

Аналогічно розробник може побудувати граф функціонального рівня. Для цього за (3.6) розробляється комплекс завдань ($Task_j$) для рівня MS'_0.

$$Aim_i\_MS'_0 \Rightarrow \underset{j}{\Omega}(Task_j\_MS''_0), \tag{3.6}$$

де $Aim_i\_MS'_0$ – головна мета розробки CPPS;

$Task_j\_MS''_0$ – завдання рівня MS''_0 для досягнення $Aim_i\_MS'_0$.

Визначимо *Task* вузлами графа на кожному рівні декомпозиції, а ребрами графа – зв'язок (→) між завданнями для досягнення головної $Task_j$ на цьому рівні, тому необхідно провести фіксацію зв'язків між завданнями за допомогою лічильника задач:

$$\begin{aligned} &Task_j \rightarrow Task_{j+1}; Task_j \rightarrow Task_{j+2}; \\ &Task_{j+1} \rightarrow Task_{j+3}; Task_{j+n-1} \rightarrow Task_{j+n} \end{aligned} \tag{3.7}$$

Запропонована «спадковість» дозволяє отримати підграфову модель представлення $Task_j$, приклад якої наведено на рис. 3.2.

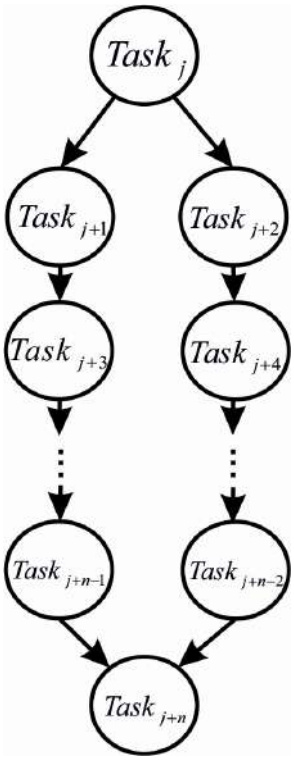


Рисунок 3.2 – Підграфова модель представлення $Task_j$

Базуючись на підграфовій моделі представлення $Task_j$, отримуємо можливість провести декомпозицію наступного підрівня $Sub\_S_k\_MS''_0$. Логічно, що для даного підрівня тотожним буде запис:

$$Task_j\_MS''_0 \Rightarrow \underset{m}{\Omega}(Task_m\_Sub\_S_k\_MS''_0), \tag{3.8}$$

де $Task_m\_Sub\_S_k\_MS''_0$ – завдання на функціональному етапі рівня Sub_S_k.

Підставивши підграф (3.2) в математичне уявлення (3.8) можна отримати граф декомпозиції (рис. 3.3.) головного завдання рівня MS'_0 до рівня Sub_S_k.

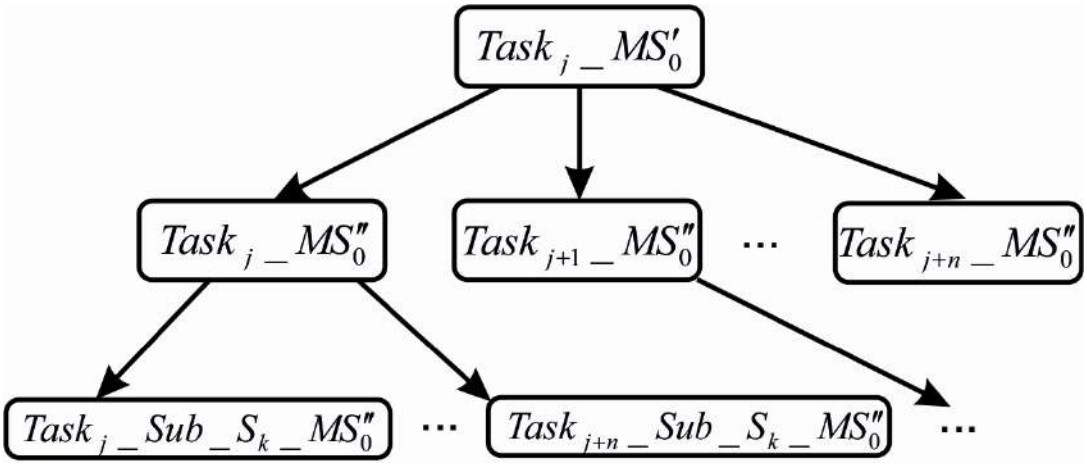


Рисунок 3.3 – Граф декомпозиції функціонального етапу до Sub_S_k рівня

Аналогічно (3.6) та (3.8) можна вивести залежності для всіх рівнів декомпозиції на функціональному етапі:

– для $Task_m\_Sub\_S_k\_MS''_0$:

$$Task_m\_Sub\_S_k\_MS''_0 \Rightarrow \underset{l}{\Omega}(Task_l\_G\{AEofS\}_p\_MS''_0). \qquad (3.9)$$

– для $Task_l\_G\{AEofS\}_p\_MS''_0$:

$$Task_l\_G\{AEofS\}_p\_MS''_0 \Rightarrow \underset{s}{\Omega}(Task_s\_AEofS_t\_MS''_0). \qquad (3.10)$$

Аналогічно розробляються системні функціональні моделі на всіх рівнях декомпозиції, де ребра – зв'язки між ними.

Метод побудови системної організаційно-технічної моделі буде аналогічним функціональній моделі і будується у вигляді графа, в якому вузлом виступатиме $StrE_q$, а ребрами – «спадковість» зв'язків у вигляді:

$$\begin{array}{l} StrE_q \rightarrow StrE_{q+1}; StrE_q \rightarrow StrE_{q+2}; \\ StrE_{q+1} \rightarrow StrE_{q+3}; StrE_{q+n-1} \rightarrow SrtE_{q+n} \end{array}. \qquad (3.11)$$

Ґрунтуючись на цьому можна уявити системну модель організаційно-технічного етапу у вигляді графа, представленого на рис. 3.4.

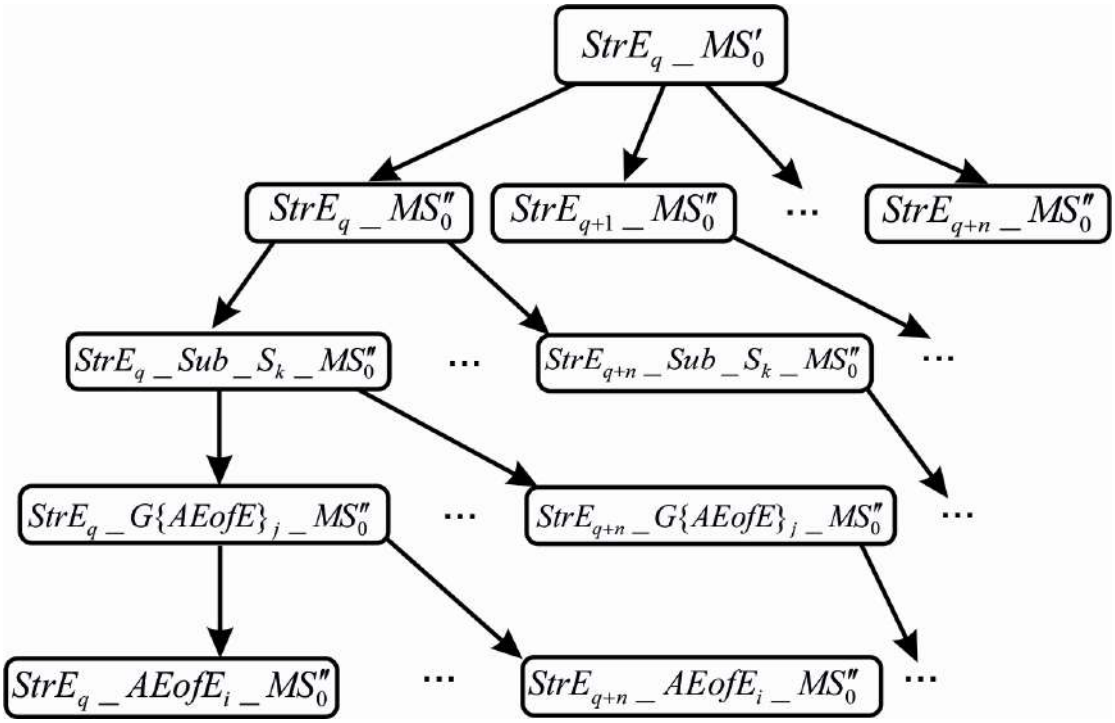


Рисунок 3.4 – Граф системної моделі організаційно-технічного етапу

Отже, на базі запропонованого методу, можна побудувати комплекс системних моделей організаційно-технічного етапу для аналізу досліджень прийнятих рішень на будь-якому рівні декомпозиції.

Для опису системної інфологічної моделі приймемо IC_v у вигляді вузла графа (підрозділ 2.5), а в якості ребра графа виступатимуть *InputCanal* і *OutCanal* зв'язків. Ґрунтуючись на розробленому методі побудови системної моделі Aim_i, опишемо системну інфологічну модель рівня MS''_0:

$$I^{MS}\_MS_0' \Rightarrow \Omega_v(IC_v\_MS_0'', \\ InputCanal\_IC_v\_MS_0'', OutCanal\_IC_v\_MS_0''), \quad (3.12)$$

де $I^{MS}\_MS_0'$ – інфологічна модель системи рівня MS''_0;

$IC_v\_MS_0''$ – інформаційний перетворювач на рівні MS''_0;

$InputCanal\_IC_v\_MS_0''$ – вхідні канали на рівні MS''_0;

$OutCanal\_IC_v\_MS_0''$ – вихідні канали на рівні MS''_0.

Графова системна модель інфологічного етапу, на рівні декомпозиції MS'_0, представлена на рисунку 3.5.

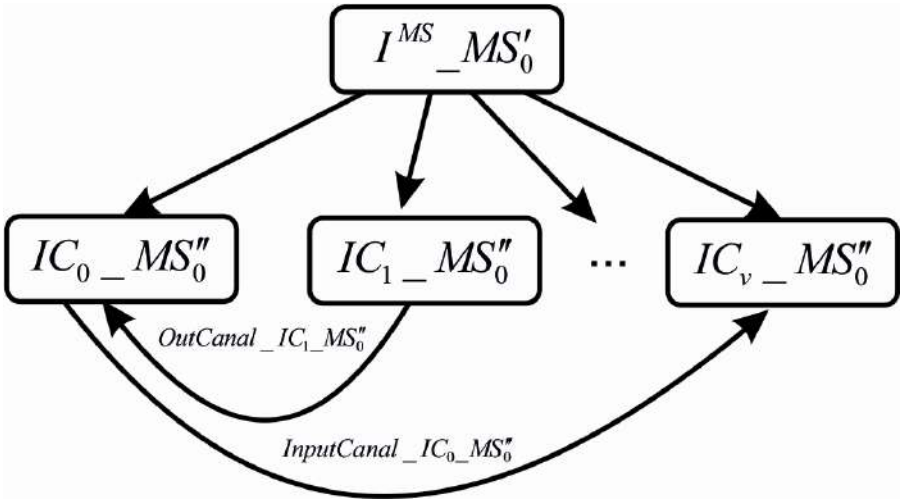


Рисунок 3.5 – Графова системна модель інфологічного етапу на рівні декомпозиції MS'_0

Аналогічно (3.12) можна уявити системну модель інфологічного етапу за наступними підсистемними рівнями:

– системна модель на рівні $I^{MS}\_MS''_0$:

$$I^{MS}\_MS''_0 \Rightarrow \underset{v}{\Omega}(IC_v\_Sub\_S_k, InputCanal\_IC_v\_Sub\_S_k, OutCanal\_IC_v\_Sub\_S_k). \quad (3.13)$$

– системна модель на рівні $I^{MS}\_Sub\_S_k$:

$$I^{MS}\_Sub\_S_k \Rightarrow \underset{v}{\Omega}(IC_v\_G\{AEofS\}_p, InputCanal\_IC_v\_G\{AEofS\}_p, OutCanal\_IC_v\_G\{AEofS\}_p). \quad (3.14)$$

– системна модель на рівні $I^{MS}\_G\{AEofS\}_j$:

$$I^{MS}\_G\{AEofS\}_p \Rightarrow \Omega_v(IC_v\_AEofS_t, \\ InputCanal\_IC_v\_AEofS_t, OutCanal\_IC_v\_AEofS_t) \quad (3.15)$$

– системна модель на рівні $I^{MS}\_AEofS_i$

$$I^{MS}\_AEofS_t \Rightarrow \Omega_{p,q}(AEofS_t(InputIP_p\_AEofS_t, OutIP_q\_AEofS_t)). \quad (3.16)$$

Таким чином, на базі розроблених системних моделей (3.12)–(3.16), можна отримати набір інфологічних системних моделей на будь-якому рівні представлення «верхнього» рівня через «нижній», для аналізу правильності вибору інформаційного перетворювача, в залежності від каналів зв'язків.

За результатами, отриманими на інфологічному етапі, можна приступити до реалізації системних моделей інформаційного етапу розробки CPPS. На даному етапі основним завданням є розробка системних технічних характеристик фізико-інформаційної моделі (*PCofI_PIM*) на всіх рівнях декомпозиції CPPS. Базуючись на запропонованих методах, в підрозділі 2.5, можна представити системну модель інформаційного етапу за наступними підсистемними рівнями:

– системна модель рівня $PCofI\_PIM\_MS_0''$:

$$PCofI\_PIM\_MS_0'' \Rightarrow \Omega_k(I^{MS}\_Sub\_S_k\_MS_0'', PCofI\_PIM\_Sub\_S_k). \quad (3.17)$$

– системна модель рівня $PCofI\_PIM\_Sub\_S_k$:

$$PCofI\_PIM\_Sub\_S_k \Rightarrow \Omega_p(I^{MS}\_G\{AEofE\}_p\_MS_0'', \\ PCofI\_PIM\_G\{AEofE\}_p) \quad (3.18)$$

– системна модель рівня $PCofI\_PIM\_G\{AEofE\}_p$:

$$PCofI\_PIM\_G\{AEofE\}_p \Rightarrow$$
$$\Rightarrow \underset{t}{\Omega}(I^{MS}\_AEofS_t\_MS''_0, PCofI\_PIM\_AEofS_t). \quad (3.19)$$

– системна модель рівня $PCofI\_PIM\_AEofS_t$:

$$PCofI\_PIM\_AEofS_t \Rightarrow$$
$$\Rightarrow \underset{p,q}{\Omega}(PCofI\_InputIP_p\_AEofS_t, PCofI\_OutIP_q\_AEofS_t). \quad (3.20)$$

Для опису системної інформаційної моделі приймемо *PCofI_PIM* у вигляді вузла даного рівня, а ребрами виступатимуть тактико-технічні характеристики інформаційних зв'язків (*PCofI_InputIP*$_p$, *PCofI_OutIP*$_q$). Приклад графової інформаційної системної моделі для (3.19) представлений на рис. 3.6. При цьому потрібно дотримуватися умови існування бінарних співвідношень:

$$PCofI\_PIM\_AEofS_0 \cong PCofI\_PIM\_AEofS_t. \quad (3.21)$$

Останнім етапом розробки CPPS, у відповідності до запропонованої архітектурно-логічної моделі керування процесами в складних організаційно-технічних виробничих об'єктах (рис. 2.1), є розробка алгоритму функціонування (*AF*) на кожному рівні декомпозиції CPPS.

Для його побудови пропонується використовувати принцип граф-схем із-за зручності їх подання. Отже пропонується наступний метод їх побудови:

– проводиться аналіз *Aim*$_i$, *Task*$_j$, *InputCanal*, *OutCanal* системного рівня розробки;

– відповідно кожній *Task*$_j$ обираємо *OP*$_h$ граф-схему алгоритму, на базі запропонованого методу (підрозділ 2.5). Позначимо, що першим завданням рівня *Task*$_0$*_MS*$'_0$, буде *OP*$_0$, отже для наступного рівня декомпозиції

$Task_0\_MS'_0$ є $Task_1\_MS''_0$, а йому буде відповідати оператор OP_1 і за аналогією до $Task_j\_MS''_0$ оператором буде OP_h;

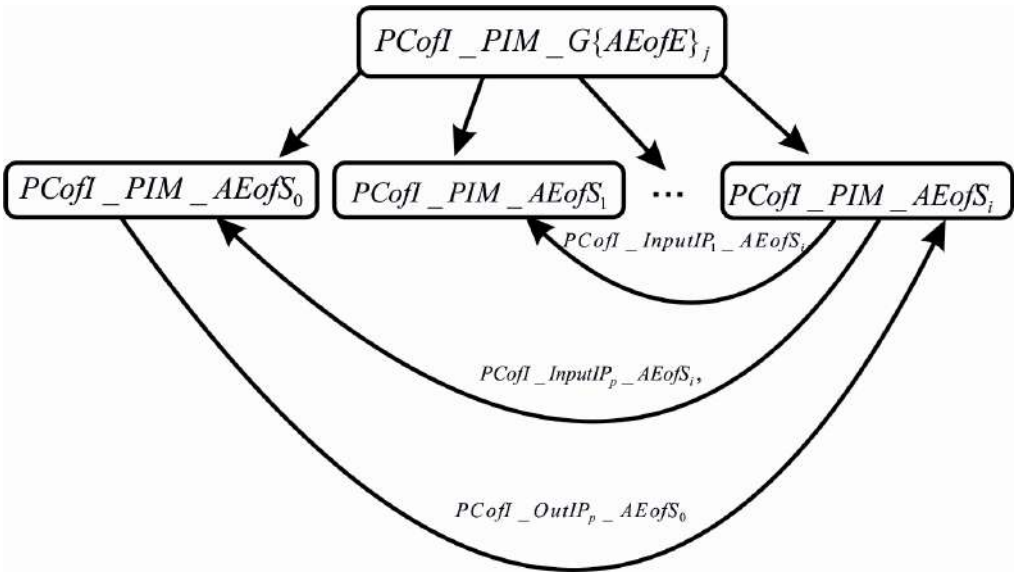


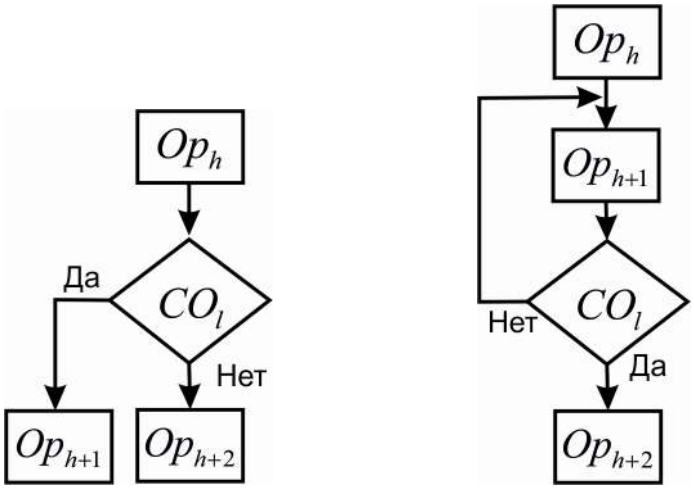
Рисунок 3.6 – Графова системна модель інформаційного етапу на рівні декомпозиції $PCofI\_PIM\_G\{AEofE\}_j$

– проводиться аналіз послідовності виконання $Task_j\_MS''_0$ для даного рівня, де досягається $Aim_j\_MS''_0$, або є необхідним для досягнення головної мети розробки CPPS;

– проводиться розстановка OP_h за логічною послідовністю для виконання $Task_j\_MS''_0$. Вводяться поняття умовного і безумовного переходу від $Task_j\_MS''_0$ до $Task_{j+1}\_MS''_0$, якщо перехід без умови то OP_h з'єднується →, якщо OP_h має зв'язок з іншими операторами OP_{h+1}, то необхідно вести поняття умовного оператора (CO_l), який працює за аналогією з блоком «умови» з базової теорії побудови алгоритму. На базі даного припущення AF може враховувати не тільки «лінійний вид», а й реалізувати «диз'юнктивний перехід», «цикли» і «умови переходу» для досягнення $Task_j$. Приклад варіантів побудови з CO_l представлений на рис. 3.7;

– маючи данний функціонал розробник може розробити AF кожному рівні системного представлення і об'єднати їх в $AF\_MS'_0$, який досягає головну мету розробки CPPS. На базі розробленого $AF\_MS'_0$ проводиться

перевірка за допомогою імітаційного моделювання під навантаженням, яка покаже правильність прийнятих рішень, як на всіх етапах так і за рівнями процесів розробки CPPS.



а) б)

а) представлення «диз'юнктивного переходу»;

б) представлення «циклу»

Рисунок 3.7 – Приклад варіантів побудови з CO_l

3.3 Формалізація системних моделей

Для формалізації системних моделей, розроблених в підрозділі 3.2 даного дослідження, було запропоновано використовувати математичний апарат регулярних схем алгоритму і алгоритмічних алгебр [197]. Обґрунтуванням цього вибору є забезпечення доступності формалізації представлення системних моделей, що дає можливість реалізувати результати формалізації, за допомогою мов високого рівня програмування, розробку систем керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS [198]. Ґрунтуючись на теорії апарату регулярних схем і алгоритмічних алгебр, введемо такі позначення:

OA_q – алгебра операторів, елементами алгебри операторів є OP_h, а також для зручності подання введемо додаткові операції, що не є операторами перетворення інформації: H – тотожний оператор, Ø– порожній оператор;

OA_r – алгебра умов включає в себе всі логічні умови CO_l, можуть набувати наступних значень *true, false* або $CO_l^x \quad x = [true, false, b, c, y, \ldots, r]$. Отже можна уявити OA_q у вигляді набору алгебр:

$$OA_q = (Op_h, CO_l, H, \varnothing, true, false, x). \tag{3.22}$$

Аналізуючи OP_h можна помітити, що реалізація кібер-складової CPPS заснована на візуальних подієвих моделях, побудованих на візуальних об'єктах інтерфейсів користувача (розділ 4), які об'єднують в собі цілі або різні частини алгоритмів та мають в собі розгалуження, цикли, лінійні ділянки, які доцільно замінити функціоналами (представити алгоритм більш укрупненої форми), а окремі частини алгоритму уявити у вигляді підалгоритму.

Для зручності маніпулювання OA_q, в рамках алгоритмічних алгебр, необхідно визначити основні типи операцій:

Визначення 1. Множення операторів – чітко послідовне виконання операторів в порядку їх черги.

$$OA = OA_i \cdot \underset{CO_l}{(} OA_k \vee OA_n) \cdot OA_m, \tag{3.23}$$

де $OA_i = Op_i, \ldots, Op_j$;

$OA_k = Op_k \underset{CO_l}{(} Op_l \vee Op_m)$, де CO_l – логічні умови;

$OA_n = Op_n \{Op_p\} Op_t$;

$OA_m = Op_m, \ldots, Op_s$.

Визначення 2. Додавання операторів – це умовне розгалуження простих, або вкладених одна в одну операцій ($\widetilde{O}p_h$).

$$OA_q = \widetilde{O}p_1 \cdot \widetilde{O}p_2 \cdot,\ldots,\cdot \widetilde{O}p_h . \tag{3.24}$$

$$OA_{q-1} = (\underset{CO_1}{} \widetilde{O}p_1 \vee (\underset{CO_2}{} \widetilde{O}p_2 \vee (\underset{CO_3}{} \widetilde{O}p_3 \vee \ldots \vee (\underset{CO_l}{} \widetilde{O}p_h \vee e))) . \tag{3.25}$$

На прикладі (3.25) пропонуються наступні розуміння: $(\underset{CO_l}{} \widetilde{O}p_h \vee e)$ результати складання простих операцій повинні задовольняти умови CO_l, після цього виконується складання до умови CO_3, і т.д. Але варто врахувати, що OP_h може бути складним компонентом алгоритму, тому пропонується вести поняття «процес складання».

Визначення 3. Процес складання в системній моделі – це дотримання умовного розгалуження і з'єднання шляхів алгоритму, в залежності від CO_l. Синтаксис запису представлений у вигляді (3.26):

$$OA_q = (\underset{CO_l}{} Op_{h-1} \vee Op_h)^{CO_l} . \tag{3.26}$$

Приклад представлення процесу складання в системні моделі можна представити таким чином:

$$OA_q = (\underset{CO_1}{} Op_1 \vee Op_2 \cdot (\underset{CO_3}{} Op_3 \vee Op_4)^{CO_1} \vee Op_5)^{CO_3} . \tag{3.27}$$

За допомогою (3.27) в системних моделях можна описати ітерацію «вперед» після перевірки умов CO_l, при цьому необхідно характеризувати поширену ітерацію «назад», в якій повернення здійснюється до OP_h, а не до умови CO_l (3.30), що вимагає вести нове визначення.

Визначення 4. Ітераційний процес складання – це правила запису і читання послідовних і концентрично вкладених циклів. Ґрунтуючись на даному визначенні представимо існуючі типи циклів:

- проста послідовність циклів:

$$CY = CY_1 \cdot CY_2 \cdot \ldots CY_n, \tag{3.28}$$

де $\mathrm{CY_n} = \{Op_h \}_{CO_l}$;

- концентричне вкладених циклів:

$$CY_n = \{\{\{\ldots\{Op_1 \}_{CO_1} Op_2 \}_{CO_2} Op_3 \}_{CO_3} \ldots Op_h \}_{CO_l}. \tag{3.29}$$

- окремий випадок циклу CY_n з поверненням до оператора OP_h при виконанні умови CO_l:

$$CY_n = \{Op_{h-2} \ldots Op_{h-1} \}_{CO_l} Op_h. \tag{3.30}$$

Ґрунтуючись на визначеннях 1–4 можна подати такі описи ітеративних шляхів проектованого алгоритму:

– виконання інтерактивного шляху: за умовою $CO_{l-1} = false$, до закриття фігурної дужки і повторення, потім повернення до перевірки $CO_l = true$, тоді дія алгоритму переносяться за дужку, що закривається до наступного за ним оператору.

$$CY_{n-3} = \{_{CO_{l-1}} Op_1 \ldots Op_{h-1} \}^{CO_l} Op_h. \tag{3.31}$$

– виконання інтерактивного шляху за умовою $CO_{l\text{-}1} = false$, повернення до скобки, що відкривається і повторення. За умовою $CO_l = true$ дія переноситься за дужку, що закривається.

$$CY_{n-m} = \overset{CO_{l-1}}{\{} Op_1 \ldots Op_{h-1} \underset{CO_l}{\}} Op_h . \quad (3.32)$$

Умова CO_l, представлена знизу дужок, визначає перевірку його, в залежності від його параметра (*false, true*) і в залежності від заданого параметра відбувається умовний перехід. Умова $CO_{l\text{-}1}$, представлена зверху дужок, показує те місце куди треба повернутися при параметрі *false* і продовжити дію алгоритму. Приклад даного виду опису алгоритму представлений в (3.33).

$$OA_q = \overset{CO_{l-1}}{\{} Op_{h-5} \underset{CO_{l-3}}{(} Op_{h-4} \vee Op_{h-3} \overset{CO_{l-3}}{)} \overset{CO_l}{\{} \underset{CO_{l-2}}{(} Op_{h-2} \underset{CO_{l-1}}{\}} \vee Op_{h-1} \overset{CO_{l-2}}{)} \underset{CO_l}{\}} Op_h . \quad (3.33)$$

Запропонований опис алгоритмів дозволяє уявити клас послідовних алгоритмів будь-якої складності перетинання циклів. Але дослідження сучасних алгоритмів CPPS показує, що необхідно розробити математичну формалізацію паралельних шляхів його виконання.

Визначення 5: Кон'юнкція алгоритму – це безумовне розгалуження з виконання декількох паралельних шляхів алгоритму.

$$OA_q = [Op_{h-2} \wedge Op_{H-1} \wedge Op_h]. \quad (3.34)$$

Введемо припущення, якщо дії операторів алгоритму, які укладені в квадратні дужки, починаються паралельно, переносяться за дужки. Після виконання алгоритму, за допомогою одного з шляхів, здійснюється перехід до виконання оператора, що стоїть за дужками. Для коректності цього запису

необхідно дотримати рівність всіх шляхів (3.35), тому визначимо через P – довжину шляху (кількість операторів) на даній гілці алгоритму.

$$POp_{h-2} = POp_{H-1} = POp_h. \tag{3.35}$$

Для виконання умови (3.35) при виникненні $POp_{h-2} \neq POp_{H-1}$ необхідно до меншої довжині алгоритму додати число n операторів.

Для визначення розгалуження алгоритму пропонується наступний метод запису: CO_l^x вказується внизу відкритої квадратної дужки, що означає перевірку умови і початок паралельної роботи алгоритму при *x*, а зверху над дужкою, що закриває вказується умови виходу CO_{l-1}^x з алгоритму.

Варто зауважити, що при виконанні паралельних алгоритмів необхідно врахувати параметр *x* при CO_l, який повинен визначатися розмірністю умов і завжди повинен бути визначений для кожної конкретної умови $x = [true, false, b, c, y, \ldots, r]$, виконання умов *b,c, r*, які задовольняють вимогам умов виходу CO_{l-1}, тоді можна вважати що паралельний алгоритм виконав свою функцію. Приклад запису представлений в (3.36).

$$OA_q = \underset{CO_1^x}{[} Op_{h-4} \overset{b}{\wedge} \ldots \overset{c}{\wedge} Op_{H-1} \overset{r}{\wedge} Op_h \overset{CO_{l-1}^x}{]}. \tag{3.36}$$

Аналогічно можливо існування CO_l^x при $x = [true, false, b, c, y, \ldots, r]$ для операції ітераційного процесу складання алгоритмів. Для зручності запису системної моделі пропонується наступні правила синтаксису:

– умова перевірки CO_l записується знизу закритої фігурної дужки;

– визначимо наступний запис у вигляді визначення верхніх індексів для фігурних дужок $CO_l^{true, false, b, c, y, \ldots, r}$ і розставимо в тих місцях, куди повертається дія алгоритму, в залежності від умови значень $x = [true, false, b, c, y, \ldots, r]$

$$OA_q = \overset{CO^b_{l-1}}{\{} Op_{h-4} \overset{CO^{true}_{l-2}}{\{} Op_{h-3} \{Op_{h-2} \cdot \overset{CO^y_{l-3}}{Op_{h-1}} \}_{CO_l} . \quad (3.37)$$

На базі розробленої мови формалізації системних моделей, проводиться алгебраїчний опис всіх операторів і умов їх взаємодії у вигляді алгоритму функціонування (*AF*) на всіх етапах і рівнях декомпозиції CPPS.

3.4 Метод синтезу алгоритмів функціонування CPPS

Формалізація системних моделей дає можливість розробити алгоритм функціонування певного етапу, на обраному рівні розробки CPPS, для керування процесами в складних організаційно-технічних виробничих об'єктах, але при цьому не вирішує загальної задачі їх взаємодії як загального цілого для вирішення $Aim_i\_MS'_0$. Отже алгоритми функціонування необхідно порівняти, комплексувати з частних рішень в загальний, декомпозувати із загального в частні, провести мінімізацію і здійснити алгебраїчні дії. Ідеальним рішенням є абстрактний синтез алгоритмів функціонування для отримання ефекту «одного пристрою або рішення». Тому виникає задача синтезу приватних алгоритмів функціонування в загальний *AF*.

Ґрунтуючись на вищесказаному пропонуються наступні методи синтезу:

Визначення 6. Об'єднання алгоритмів функціонування – якщо набір OP_h для виконання частних алгоритмів є загальним і CO_l теж загальний, отже їх можна об'єднати за CO_l.

$$AF = \bigcup_{OC_l} (AF_r \vee AF_m). \quad (3.38)$$

за умови, що $CO_l = (true, false)$

$$AF_r + AF_m = \underset{OC_l}{(} AF_r \vee AF_m) = \begin{cases} AF_r \text{ при } CO_l = true \\ AF_m \text{ при } CO_l = false \end{cases}. \quad (3.39)$$

Ґрунтуючись на (3.38)–(3.39) можна застосувати систему аксіом тотожних перетворень алгоритмів, що дасть можливість спростити логічну структуру реалізованого алгоритму.

Визначення 7. Декомпозиція алгоритмів функціонування – це розчленування алгоритму на прості алгоритми без втрати тотожності умов його роботи. Приклад декомпозиції алгоритму функціонування (3.40) представлений в (3.41)–(3.42).

$$AF_i = \underset{OC_{l-2}}{(} AF_j \vee \underset{OC_{l-1}}{(} AF_m \vee AF_n \overset{OC_{l-1}}{)} \overset{OC_{l-2}}{)} \cdot \underset{OC_l}{(} AF_p \vee AF_g \overset{OC_l}{)}. \quad (3.40)$$

– укрупнений приклад декомпозиції:

$$AF_i = \begin{cases} AF_1 = AF_j \underset{OC_l}{(} AF_p \vee AF_g) \text{ при } OC_{l\text{-}2} = true \\ AF_2 = \underset{OC_{l-1}}{(} AF_m \vee AF_n) \underset{OC_l}{(} AF_p \vee AF_g) \text{ при } OC_{l\text{-}2} = false \end{cases} \quad (3.41)$$

– деталізований приклад декомпозиції:

$$AF_i = \begin{cases} AF_1 = \begin{cases} AF_1' = AF_j \cdot AF_p \text{ при } (OC_{l\text{-}1} \wedge OC_l) = true \\ AF_2'' = \underset{OC_{l-1}}{(} AF_m \vee AF_n) AF_g \text{ при } (OC_{l\text{-}1} \wedge OC_l) = false \end{cases} \\ AF_2 = \begin{cases} AF_j \underset{OC_l}{(} AF_p \vee AF_g) \text{ при } (OC_{l\text{-}2} \wedge OC_{l\text{-}1}) = true \\ AF_n \underset{OC_l}{(} AF_p \vee AF_g) \text{ при } (OC_{l\text{-}2} \wedge OC_{l\text{-}1}) = false \end{cases} \end{cases}. \quad (3.42)$$

Приклад декомпозиції (3.42) доводить можливість декомпозиції складного алгоритму на прості, а на базі простих можна проводити перетворення з метою його мінімізації.

В ході аналізу існуючих математичних апаратів і алгебр маніпуляцій з алгоритмами, для вирішення питань об'єднання (склеювання) різних алгоритмів, пропонується модифікувати апарат регулярних схем системних моделей. Для цього введемо такі поняття:

– шлях – це фрагмент AF_i, який містить певну послідовність OP_h, шляхом алгоритму буде $Op_{h-1} \to Op_h$. Звідси, шлях алгоритму є простим, якщо не має розгалужень і складним, коли містить ітераційний процес (визначення 4) та кон'юнкцію (визначення 5) і т.д.

– довжина шляху – часова характеристика алгоритму, яка служить для доказу тотожності алгоритму при структурній мінімізації за однаковими шляхами. Позначимо $\vec{A}F_i$ – частковий алгоритм або шлях алгоритму.

$$\vec{A}F_i = \vec{A}F_1 \cdot \vec{A}F_2 \cdot \vec{A}F_3 \cdot \ldots \cdot \vec{A}F_{i-1}. \qquad (3.43)$$

Для об'єднання вихідних алгоритмів, необхідно провести його декомпозицію, відповідно до визначення 7 і визначити прості шляхи. Проводиться аналіз OP_h і CO_l по кожному $\to$ вихідних AF_i, з метою визначення однакових. Виходячи з цього опису приймемо такі припущення щодо еквівалентності алгоритмів:

Аксіома 1. Припустимо, що два будь-яких алгоритми $\vec{A}F_i$ і $\vec{A}F_{i-1}$ будуть тотожно еквівалентні, якщо вони складаються з однакових →, довжин шляху, OP_h і CO_l в → будуть рівнозначні, CO_l – ідентичні. Приклад реалізації аксіоми 1: якщо дано три будь-яких алгоритми (3.44)–(3.46):

$$\vec{A}F_1 = Op_1 \cdot Op_2 \underset{OC_1}{(} Op_3 \vee Op_4 \overset{OC_1}{)} \overset{OC_2}{\{} Op_5 \cdot Op_6 \underset{OC_2}{\}} \vec{A}F_{i-1}. \qquad (3.44)$$

$$\vec{A}F_2 = \underset{OC_1}{(}\ Op_3 \vee Op_4\ \overset{OC_1}{)}\ Op_1 \cdot Op_2\ \overset{OC_2}{\{}\ Op_5 \cdot Op_6\ \underset{OC_2}{\}}\ \vec{A}F_{i-1}. \quad (3.45)$$

$$\vec{A}F_3 = \vec{A}F_{i-1} \cdot Op_1 \cdot Op_2\ \underset{OC_1}{(}\ Op_3 \vee Op_4\ \overset{OC_1}{)}\ \overset{OC_2}{\{}\ Op_5 \cdot Op_6\ \underset{OC_2}{\}}. \quad (3.46)$$

Тоді можна бачити, що алгоритми $\vec{A}F_1, \vec{A}F_2$, і $\vec{A}F_3$ мають однакові оператори (OP_1, OP_2, OP_3, OP_4, OP_5, OP_6) і умови (OC_1, OC_2), а отже і однакові →. Для отримання об'єднаного алгоритму і спрощення маніпуляції з даними ведемо такі скорочення:

$$Op_1 \cdot Op_2 = W. \quad (3.47)$$

$$\underset{OC_1}{(}\ Op_3 \vee Op_4\ \overset{OC_1}{)} = M. \quad (3.48)$$

$$\overset{OC_2}{\{}\ Op_5 \cdot Op_6\ \underset{OC_2}{\}} = K. \quad (3.49)$$

$$\vec{A}F_{i-1} = Z. \quad (3.50)$$

Підставим прийняті скорочення з (3.47)–(3.50) в (3.44)–(3.46), отримаємо наступну вид запису алгоритмів (3.51)–(3.53):

$$\vec{A}F_1 = WMKZ. \quad (3.51)$$

$$\vec{A}F_2 = MWKZ. \quad (3.52)$$

$$\vec{A}F_3 = ZWMK. \quad (3.53)$$

Визначимо додаткові умови існування алгоритмів $\vec{A}F_1, \vec{A}F_2$, і $\vec{A}F_3$ через системи:

$$\hat{S}_1 = \begin{cases} \vec{A}F_1 \text{ при } s_1 = true \\ \vec{A}F_1 \ \vec{A}F_2 \text{ при } s_1 = false \end{cases}. \quad (3.54)$$

$$\hat{S}_2 = \begin{cases} \vec{A}F_2 \text{ при } s_2 = true \\ \vec{A}F_3 \text{ при } s_2 = false \end{cases}. \quad (3.55)$$

На першому кроці синтезу алгоритмів $\vec{A}F_1, \vec{A}F_2$, і $\vec{A}F_3$ необхідно визначити однакові шляхи за критерієм найменшої повторюваності і зробити запис відповідно до умов існування $\vec{A}F_1, \vec{A}F_2$, і $\vec{A}F_3$. В результаті маніпуляцій можна отримати багато варіантів синтезу алгоритмів в один, і в залежності від вимог, які необхідно досягти обирається найбільш оптимальний. Запис (3.56) наводить приклад синтезу алгоритму за критерієм мінімізації кількості операторів, використовуючи наведені скорочення (3.51)–(3.53) з урахуванням додаткових умов (3.54)–(3.55).

$$\vec{A}F_{1-3} = (\underset{s_2}{} M \vee Z) \cdot W \cdot (\underset{s_1 \vee \bar{s}_2}{} M \vee e) \cdot K \cdot (\underset{s_1 \vee s_2}{} Z \vee e). \quad (3.56)$$

Проведемо підстановку в (3.56) прийнятих в (3.47)–(3.50) скорочень, отримаємо синтезований алгоритм $\vec{A}F_{1-3}$, у вигляді набору операторів і умов.

$$\begin{array}{l} \vec{A}F_{1-3} = (\underset{s_2}{} (\underset{OC_1}{} Op_3 \vee Op_4) \vee \vec{A}F_{i-1})^{s_2} \cdot Op_1 \cdot Op_2 \ (\underset{s_1 \vee \bar{s}_2}{} (\underset{OC_1}{} Op_3 \vee Op_4) \vee e)^{s_1 \vee \bar{s}_2} \\ \{Op_5 \cdot Op_6\}_{OC_2} \ (\underset{s_1 \vee s_2}{} \vec{A}F_{l-1} \vee e) \end{array}. \quad (3.57)$$

Аналізуючи результат синтезу об'єднаного алгоритму (3.57) і вихідних алгоритмів (3.44)–(3.46), можна побачити що в алгоритмах $\vec{A}F_1, \vec{A}F_2$ і $\vec{A}F_3$ є

21 оператор і 6 умов, при заданому критерії мінімізації операторів, було отримано 10 операторів при збереженні кількості умов [199]. Тому можна з упевненістю стверджувати, що кількість операторів при цьому синтезі, скоротилося в 2 рази, при цьому алгоритм виконує всі початково задані умови. Використовуючи запропоновані в підрозділі 3.4 методи можна маніпулювати з операторами алгоритмів і проводити синтез для мінімізації їх структури і отримання еквівалентних алгоритмів на всіх етапах і рівнях розробки CPPS, в залежності від вимог ТЗ.

Висновки до розділу 3

1. В ході аналізу результатів, отриманих у 2 розділі, було відзначено, що існує певна закономірність використання одного методу прийняття рішень на різних рівнях розробки CPPS. Тому для спрощення розробки системних моделей запропоновано провести групування методів на 5 підгруп, в ході якого було виявлено закономірності трансформацій і доведене існування ізоморфізму. Це дало можливість дати визначення системним моделям і на їх базі розробити метод представлення процесу управління розробкою CPPS на всіх рівнях декомпозиції (3.1)–(3.21). Архітектурно-логічні моделі розроблені на базі теорії графів, а їх формалізація за допомогою математичного апарату регулярних схем алгоритму і алгоритмічних алгебр. Це дозволило дати визначення операторам, умовам (3.22), на базі якого розроблені основні типи операцій, що дає можливість проводити маніпуляції з ними (3.23)–(3.37). Ґрунтуючись на запропонованих операціях можна провести алгебраїчний опис всіх операторів і умов їх взаємодії у вигляді алгоритму функціонування (*AF*) на всіх етапах і рівнях процесу управління розробкою CPPS.

2. Формалізація системних моделей і алгебраїчного опису всіх операторів і умов, які представлені в підрозділах 3.1–3.3, дало можливість розробити алгоритми функціонування на певному етапі на обраному рівні

розробки CPPS, але при цьому не вирішує загальної задачі взаємодії їх як загального цілого для вирішення головної мети розробки CPPS. Тому алгоритми функціонування необхідно порівнювати, комплексувати з частних рішень в загальні, декомпозувати із загального в частні, проводити мінімізацію і здійснювати алгебраїчні дії.

Таким чином була виконана задача розробки методів синтезу (визначення 6 та 7) алгоритмів функціонування кожного рівня і етапів розробки CPPS в єдиному алгоритмі функціонування, який забезпечує досягнення головної мети розробки CPPS.

3. Виходячи із запропонованого методу синтезу алгоритмів функціонування, були проведені теоретичні дослідження прикладу синтезу трьох алгоритмів (3.44)–(3.46), які показали, що на початковому етапі був 21 оператор і 6 умов, при заданих умовах мінімізації операторів. В результаті апробації розробленого методу синтезу було отримано 10 операторів, при збереженні кількості умов, що дало можливість мінімізувати загальний алгоритм функціонування CPPS в 1,5–2 рази, при цьому синтезований алгоритм виконує всі початкові умови. Підвівши підсумки можна сказати, що розроблений метод синтезу дає можливість автоматизувати процес розробки узагальненого алгоритму функціонування.

4. Розроблений метод синтезу дає можливість автоматизувати процес розробки узагальненого алгоритму функціонування CPPS для керування процесами в складних організаційно-технічних виробничих об'єктах.

Список джерел, які використано у даному розділі, наведено у повному списку використаних джерел [197–199].

4 МОДЕЛІ ТА МЕТОДИ РОЗРОБКИ КІБЕРНЕТИЧНОЇ СКЛАДОВОЇ ДЛЯ КЕРУВАННЯ ПРОЦЕСАМИ В ОРГАНІЗАЦІЙНО-ТЕХНІЧНИХ ВИРОБНИЧИХ ОБ'ЄКТАХ

4.1 Аналіз вимог, що пред'являються до Smart Factory

Розробка CPPS керування процесами в складних організаційно-технічних виробничих об'єктах, в рамках технології Industry 4.0, є складним завданням синтезу фізичних і програмних рішень, які в сукупності повинні забезпечити безперервний процес функціонування, моніторингу та аналізу даних, в контексті хмарного і віддаленого виробництва, з використанням глобальних інформаційних мереж (iFactory).

Запропоновані у другому розділі дисертації методи керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS, дозволяють уявити CPPS як цільову систему, яка декомпозується за етапами і рівнями розробки, що дає можливість прийняти комплекс рішень і реалізувати їх у вигляді формалізованого підходу для оцінки правильності прийнятих рішень на кожному етапі і рівні розробки.

Розроблені в третьому розділі системні моделі та їх формалізація дозволяють розробити алгоритми функціонування організаційно-технічних виробничих об'єктів, на базі CPPS, як окремо за рівнями і етапам розробки, так і в єдиному алгоритмі функціонування CPPS. Але варто зауважити, що найскладнішою задачею є розробка кібернетичної складової CPPS.

Досліджуючи рішення компаній промислової автоматизації Advantech, можна помітити, що принцип Industry 4.0 – це використання технологій M2M (машина до машини), що вимагає постійного забезпечення інформацією про стани системи управління виробничими процесами для аналізу і швидкості прийняття виробничих і управлінських рішень. Таким чином досягається інтеграція в єдину корпоративну мережу iFactory, яка забезпечує синтез

функції систем виробничого процесу (c-MES) [200]. Основні функції c-MES:

– PM (Process Management) – керування процесами виробництва;

– DPU (Dispatching Production Units) – диспетчеризація виробництва;

– QM (Quality Management) – управління якістю;

– DCA (Data Collection / Acquisition) – збір і зберігання даних;

– PA (Performance Analysis) – аналіз ефективності;

– RAS (Resource Allocation and Status) – контроль стану і розподіл ресурсів;

– LUM (Labor / User Management) – управління людськими ресурсами;

– PTG (Product Tracking & Genealogy) – відстеження і генеалогія продукції.

Як можна бачити з функцій, c-MES – дуже складна колобарація взаємопов'язаних програмних рішень в єдиний цілий інформаційний простір підприємства.

Варто врахувати ще один важливий факт того, що розробка CPPS є одиничним рішенням, яке залежить від спрямованості підприємства та його технологічного забезпечення, що ставить перед розробником складне завдання проектування і розробки єдиного інформаційного середовища. Звідси виникає дилема: з одного боку існують методики розробки ПП та їх комерційні аналоги для вирішення деяких функцій c-MES, з іншого – розробка інформаційного простору в розроблюємій CPPS повинна охоплювати і об'єднувати всі етапи виробництва в єдине ціле, в рамках специфіки (обладнання, ПЛК, датчики, і т.д.) і завдань (типи і види технологічних процесів, види виробництв, методи аналізу та керування, і т.д.) роботи iFactory. Виходячи з вищевказаного можна сказати, що не існує певних рекомендацій, методів та моделей розробки кібернетичної складової складних організаційно-технічних виробничих об'єктах, в рамках технології Industry 4.0.

Ґрунтуючись на проведеному аналізі можна зробити висновок, що існуючі аспекти та рекомендації, представлені в сучасних міжнародних

стандартах, не дозволяють використовувати їх в рамках комплексного підходу для вирішення завдання розробки кібернетичної складової і в основному мають часткові пропозиції і рекомендації щодо систематизації розробки прикладних ПП загального призначення. Виходячи з цього, для реалізації єдиного підходу до розробки кібернетичної складової складних організаційно-технічних виробничих об'єктах, в даному дослідженні, пропонується удосконалити модель життєвого циклу (ЖЦ) процесу керування розробкою кібернетичної складової «Jump», яка представлена на рис. 4.1.

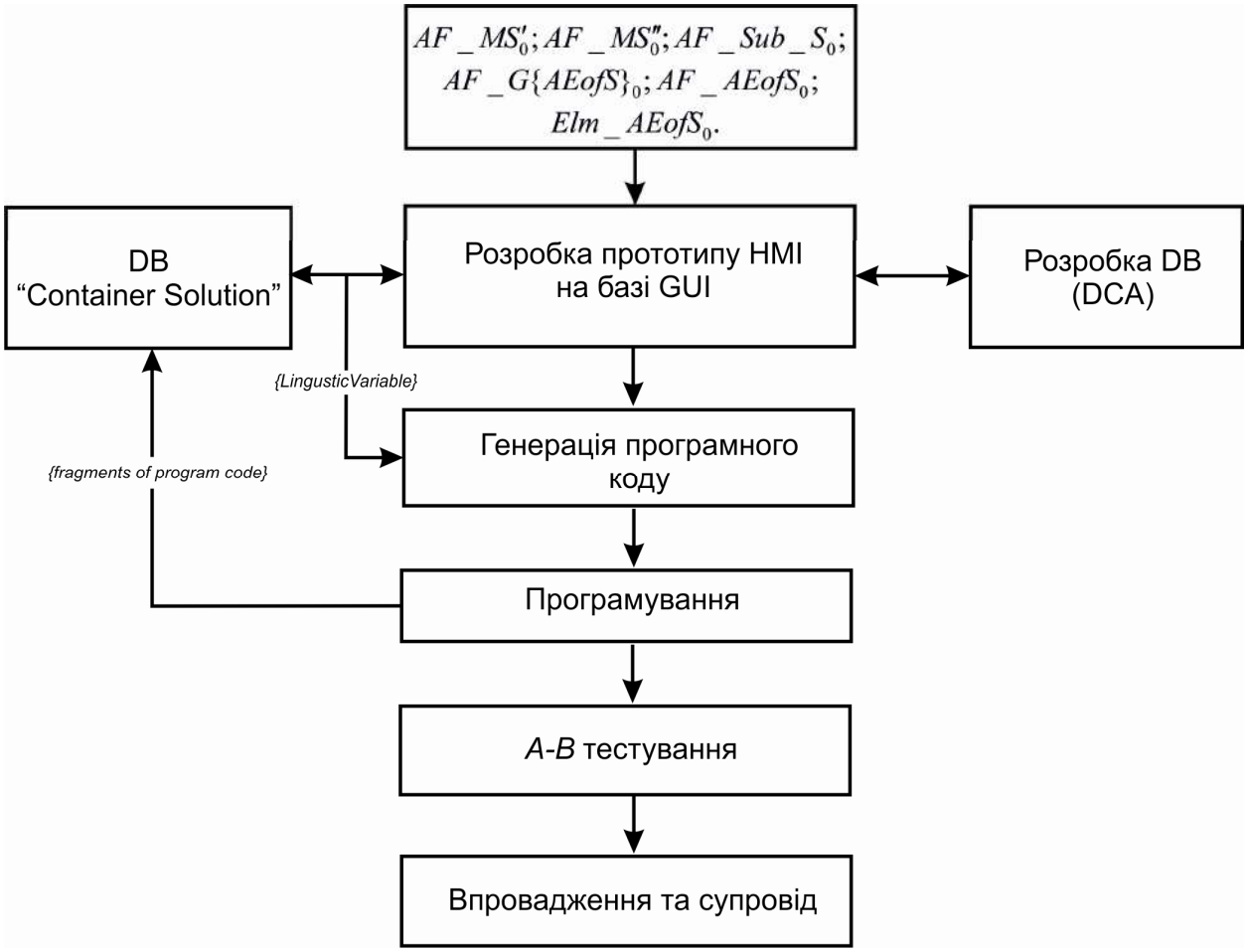


Рисунок 4.1 – Удосконалена модель ЖЦ процесу керування розробкою кібернетичної складової «Jump»

Переваги даної моделі ЖЦ полягають в тому, що вона складається з основних етапів:

– аналізу алгоритму функціонування складних організаційно-технічних виробничих об'єктах (загальний алгоритм функціонування CPPS, за етапами і рівнями декомпозиції);

– розробки прототипу інтерфейсу користувача для візуалізації функції DPU, PM, QM і т.д., залежно від вимог ТЗ на CPPS;

– генерація програмного коду прототипу інтерфейсу, який враховує оператори і події при виконанні заданих умов, визначених алгоритмом функціонування CPPS на кожному рівні і етапі декомпозиції;

– тестування і впровадження в єдину корпоративну мережу iFactory.

Запропонована модель ЖЦ процесу керування розробкою кібернетичної складової «Jump» включає в себе синтез існуючих концепцій і підходів до розробки прикладних ПП, модифікованих під запропоновані моделі і методи, які представлені в другому і третьому розділі дисертації.

Розглянемо детально кожен етап ЖЦ процесу керування розробкою кібернетичної складової «Jump», як сукупність взаємопов'язаних підсистем, що утворюють цілісну послідовність вирішення завдання автоматизації розробки кібернетичної складової CPPS.

Запропонована модель ЖЦ заснована на моделях і методах розробки CPPS за наступними принципами: єдності та ієрархії; відкритого ступеня взаємодії із зовнішнім середовищем; властивості самоорганізації при зміні специфіки завдань; вимог до мов високого рівня програмування і середовища розробки, в залежності від вимог що пред'являється до неї.

Для опису всіх зв'язків в підсистемах і всередині системи запропоновано використовувати методи структуризації, на базі теоретико-множинних уявлень і інформаційного підходу до формалізації процесів та зв'язків, що протікають всередині моделі і на кожному етапі. Даний підхід дозволить вирішити такі важливі завдання:

– забезпечить представлення моделі ЖЦ на такому рівні абстракції, на якому вона стає виразною і інтуїтивно зрозумілою аналітикам, розробникам CPPS, Software Product Manager, програмістам і кінцевим користувачам;

– забезпечить специфікацію CPPS, її структуру і поведінкову організацію, в залежності від заданого в ТЗ середовища розробки та специфіки виробництва.

Результати аналізу і вимог, що пред'являються ТЗ і алгоритмом функціонування CPPS на всіх етапах і рівнях декомпозиції, для досягнення головної мети розробки CPPS, обумовлюють необхідність розробки нових або вдосконалення існуючих моделей і методів. Для досягнення мети досліджень, в даній роботі, необхідно досягти максимально можливу автоматизацію процесу керування розробкою кібернетичної складової CPPS у вигляді адитивного кібер-дизайну, за рахунок наступних вимог, що висуваються до моделі:

– модель ЖЦ «Jump» повинна володіти необхідною і достатньою спільністю, що б її засоби надавали можливість забезпечити прозоре, адекватне, комплексне уявлення даних і зв'язків для всіх учасників проекту з виконанням умов єдиної структури і цілісності;

– розбіжності, між пропонованої моделлю ЖЦ «Jump» і об'єктно-орієнтованими підходами, на яких реалізуються необхідні функції c-MES, ERP в єдиному інформаційному середовищі Smart Factory, повинні бути адаптивні до динамічних змін предметної області CPPS, в залежності від ТЗ, а алгоритм функціонування бути мінімальним але достатнім [201,202].

Отже, склад і структура набору формальних об'єктів, створюваних в моделі ЖЦ «Jump», на базі принципу об'єктно-орієнтованої побудови, повинні бути максимально близькими, що б при відображенні концептуальної моделі CPPS можна скористатися відомими правилами перетворення структур, побудови, ієрархії без втрат цілісності;

– можливість комплексного використання розроблених алгоритмів функціонування складних організаційно-технічних виробничих об'єктів на кожному етапу та рівні декомпозиції, як загального функціонального алгоритму функціонування CPPS;

– реалізація «часткової автоматизації» за рахунок генерації програмного коду події, адитивного кібер-дизайну інтерфейсу користувача, на базі функціональних алгоритмів і головної мети розробки;

– можливість використання природних мов опису подій і об'єктів, заданих в функціональному графі CPPS, для розробки адитивного кібер-дизайну інтерфейсу користувача (HMI) на базі графічних елементів (GUI) об'єктно-орієнтованого підходу до програмування.

Для забезпечення сформульованих вище вимог виділимо чотири основні етапи:

– виявлення деякої множини латентно-семантичних понять (концепцій) (LSA), які будуть використовуватися для опису моделі;

– визначення набору формальних об'єктів, які можуть використовуватися для подання обраних понять;

– визначення формальних правил підтримки цілісності даних в моделі ЖЦ «Jump»;

– визначення формальних операторів взаємодії моделі ЖЦ «Jump» з «реальним світом», між етапами, як підсистемами даної моделі, а також взаємодії всередині кожної підсистеми до атомарного рівня декомпозиції графу функціонування CPPS керування процесами в складних організаційно-технічних виробничих об'єктах [203].

Інформація про розглянуту предметну область («реальний світ») представляється через сприйняття розробником або аналітиком, і складається з безлічі взаємозалежних і взаємопов'язаних факторів, що орієнтуються на розгляд моделі ЖЦ «Jump» з позиції закономірності системного цілого і взаємодії його складових сутностей (етапів).

Ґрунтуючись на концепції багаторівневої ієрархії суті моделі ЖЦ «Jump», процеси і явища доречно розглядати як множину дрібніших ознак (підмножин) опису. При цьому етапи логічно розглядати як елементи більш високих класів узагальнень.

Виходячи з запропонованої моделі, ЖЦ «Jump» повинний

забезпечувати адекватне правильне сприйняття. Отже, доцільно використовувати принцип, за яким таке уявлення спочатку вибудовує для себе аналітик або розробник природною мовою.

Ґрунтуючись на визначенні, що всі процеси «реального часу» протікають в просторі і в часі, то модель ЖЦ «Jump» являє собою динамічну систему, яка складається з певних послідовностей станів, де в кожен момент часу визначається її множина кінцевих станів.

Кожна множина кінцевих станів описується елементарними факторами природною мовою в термінах об'єкта, зв'язків і приналежностях об'єкту, значеннях параметрів і його властивостей, під час виконання тієї чи іншої події, що задаються за допомогою розроблених моделей і методів представлення алгоритму функціонування процесами, описаних в 2-3 розділах дисертаційної роботи.

Опис моделі ЖЦ «Jump» можна уявити з точки зору доцільності у термінах предметної області, розширивши і синтезувавши їх набір. В контексті розробленої моделі ЖЦ «Jump» запропоновані наступні поняття:

– об'єкт, пов'язаний з різними частинами моделюємого ПП або модулями для адитивного кібер-дизайну;

– властивість (опис характеристик) об'єкта, який дозволяє співвіднести безліч об'єктів (підкласів) і їх властивостей в один загальний клас;

– подія – реакція, яка відбувається на об'єкті (певного підкласу), кожна подія описується його властивостями і визначається моментом часу його настання або виклику;

– значення характеристик об'єкта і події – необхідні ознаки (параметри, що відносяться до певних класів і їх підкласів) уявлення об'єкта (опис його візуалізації), або реакції на реалізацію події, у вигляді певних необхідних дій, яке представлену як «контейнер рішень»;

– елемент – частина об'єкта (підклас), який формується як залежна частина об'єкта і визначає необхідність логічної взаємодії всередині об'єкта або групи об'єктів, в контексті певного і необхідного рішення.

При цьому, варто зауважити, що основним в удосконаленій моделі ЖЦ «Jump», є обов'язкова умова існування «об'єкта» адитивного кібер-дизайну (форми, GUI елемента) і події, які можуть належати як об'єкту або класу, так і підкласам які входять в них.

В удосконаленій моделі ЖЦ «Jump» знайшли застосування певні підходи, рішення і методології, які вже апробовані: методологія RUP (Rational Unified Process) – розроблена компанією Rational Software; методологія Microsoft Solution Framework – запропонована компанією Microsoft, заснована на практичному досвіді; методологія Scrum – управління проектами, яка заснована на принципах Time-менеджменту; гнучка методологія Agile-розробки (швидке проектування і розробка ПП і ПМ, де головне завдання – працюючий продукт, а не його документація; методологія візуального об'єктно-орієнтованого програмування (абстрагування, поліморфізм, інкапсуляція), семантичної мережевої, онтологічної, інфологічної логіки предикатів з розширеною підтримкою в часі.

Грунтуючись на тому, що парадигма методології візуального об'єктно-орієнтованого програмування концептуальної та логічної моделі близькі, можна спростити перехід від концептуальної моделі до логічної. Також на реалізацію моделі ЖЦ «Jump» вплинули: семантичні моделі, що дають можливість опису і представлення складних ситуацій; концепція ієрархії типів, що дозволяє обґрунтувати і систематизувати уявлення класів, метакласів; концепція ролі понять і часу, що дозволяє реалізувати висунуті вище вимоги, які пред'являються до моделі ЖЦ «Jump» розробки адитивного кібер-дизайну керування процесами в складних організаційно-технічних виробничих об'єктах.

4.2 Формалізація моделі ЖЦ «Jump» для розробки адитивного кібер-дизайну

Відповідно певних вимог і понять, модель ЖЦ «Jump» розробки адитивного кібер-дизайну ($\aleph$) було проведенно ряд досліджень [204] – [211], результати якого представлені в додатку В. Це дозволило представити і формалізувати $\aleph$, як наступний кортеж, на базі четвертого виду представлення системи і на основі класичної теорії системного аналізу:

$$\aleph = \langle P(\Im, \Re, H), K, L \rangle, \tag{4.1}$$

де $P(\Im, \Re, H)$ – сукупність правил і структурування даних про об'єкт моделювання, яка визначається AF_MS' та ТЗ для розробки адитивного кібер-дизайну;

P – об'єкт моделювання;

$\Im$ – множина базових понять (концепцій уявлення) об'єкта моделювання;

$\Re$ – множина відносин між базовими поняттями моделі;

H – множина функцій описів і трактування базових понять і відносин;

K – множина вимог обмеження цілісності;

L – множина представлення моделювання даними.

Для подальшого опису сукупності правил і структурування даних P, необхідно розробити словник опису і трактування базових понять H, який складений для множин базових понять $\Im$. Уявімо множину $\Im$ як набір базових понять в такому вигляді:

$$\begin{aligned} \Im = \{ & Form, ParameterForm, ValueForm, \ EventForm, \\ & LinguisticVariable, ElementForm, ParameterElement, . \\ & ValueElement, EventElement, ContainerSolutions\} \end{aligned} \tag{4.2}$$

Ці поняття є основними елементами уявлення деякого формального об'єкта моделювання P. При розробці опису адитивного кібер-дизайну кожне поняття з (4.2) буде містити конкретні імена (назви), які будуть запозичуватись зі словників і визначень того чи іншого середовища розробки, або неформально, шляхом узгодження базових понять між розробником і адитивного кібер-дизайну для керування процесами в складних організаційно-технічних виробничих об'єктах.

Звідси $\Re$ – безпосередня множина відносин між однойменними множинами, співвідношеннями з даними поняттями і їх елементами, як формальними об'єктами, що дає можливість представляти властивість модельованих елементів адитивного кібер-дизайну, їх взаємодію і зв'язки, за умови, що модель має кінцеву множину таких зв'язків і взаємодій та визначаються алгоритмом функціонування і цілям (Aim_i) для кожного етапу так і в цілому для CPPS.

Наведемо опис основних базових понять з (4.2), зумовивши неформальне визначення та призначення, в рамках формалізації об'єкта моделювання, на базі удосконаленої моделі ЖЦ «Jump»:

Form (Windows Form) – деяка виділена і унікально ідентифікована частина предметної області. Її призначення – опис і подання візуальної структури адитивного кібер-дизайну у вигляді основних блоків;

ParameterForm – сукупність типів і способів опису властивостей предметної області, виділених і згрупованих за деякими ознаками, а також ідентифікованих за ім'ям. Призначення – опис параметрів, необхідних і достатніх для відображення і моделювання візуального представлення *Form*;

ValueForm – значення, яке присвоюються типу і способу опису властивостей предметної області. Призначення – привласнення конкретного значення (цілочисельного, лінгвістичного, булевого) типу або способу опису параметрів, в залежності від функціональних ознак і допустимих рішень;

EventForm – подія або група подій (дія), які можуть відбуватися (вже

відбулися, або відбудуться) з предметною областю, в певний момент або інтервал часу. Ідентифікується часом (необхідністю) і об'єктом, до якого належить подія. З одним об'єктом, в один момент часу, може відбуватися тільки одна подія, яка попередньо ініціалізується користувачем. Необхідні для опису призначення допустимого набору подій (умов CO_l) (підрозділ 2.3), які задаються в алгоритмі функціонування для кожного етапу і рівня, в залежності від вимог до алгоритмів функціонування CPPS;

LinqusticVariable («Лінгвістична змінна») – іменований (природною мовою системи) логічний опис дій при виникненні подій. Такі описи можуть бути згруповані за рядом ознак. Призначення – привласнення класу події або одиничній події лінгвістичної інтуїтивно-зрозумілої користувачеві моделі змінної опису реакцій при виникненні тієї чи іншої події;

ElementForm – елемент або група елементів GUI (підклас об'єкту) для візуального представлення взаємодії користувача і моделюємого функціоналу адитивного кібер-дизайну. Містять необхідні ознаки для реалізації інтерфейсу користувача, або управління і взаємодії з інформаційними потоками і даними, в залежності від вимог до атомарних елементів ($AEofS_i$) і алгоритмів їх функціонування при рішенні елементарних завдань до рівня MS'_0. Призначення – графічне відображення елементів або групи елементів, які можуть мати деревоподібну структуру візуального представлення і використовуватися як для реалізації інтуїтивно зрозумілого інтерфейсу, так і, безпосередньо, для взаємодії з користувачем CPPS;

ParameterElement – типи і способи опису властивостей елементів, як поодиноких, так і згрупованих за деякими ознаками і ідентифікованих ім'ям. Призначення – опис параметрів, необхідних і достатніх для подання та моделювання візуального представлення елемента, в рамках єдиного інформаційного об'єкта;

ValueElement – значення, що привласнюється типу і способу опису властивостей GUI елемента. Призначення – привласнення конкретного значення (цілочисельного, лінгвістичного, булевого) типу, або способу опису

параметрів, в залежності від функціональних ознак і реалізації візуального інтуїтивно-зрозумілого інтерфейсу кожної частини предметної області у відповідності з алгоритмом функціонування. Для деяких *ParameterForm* і *ParameterElement* при їх функціональному призначенні і поіменній ідентифікації, значення можуть бути однаковими, в залежності від вимог CPPS;

EventElement – подія, група подій або умова (CO_l), які можуть відбуватися (вже відбулися або відбудуться) з елементом GUI, який виконує певну функцію в певний момент або інтервал часу. Ідентифікується часом (необхідністю) та елементом, до якого належить подія. З одним об'єктом в один момент часу може відбуватися тільки одна подія, що ініціалізується користувачем. Призначення – одна з основних властивостей елемента, яке обмежено: функціональними можливостями цього GUI елемента, областю застосування (з точки зору необхідності використання в даній моделі адитивного кібер-дизайну CPPS), необхідністю і роллю в загальній концепції застосування в інтерфейсі користувача для відпрацювання інформаційних потоків;

ContainerSolutions («Контейнер рішень») – іменований опис реакцій, при виникненні події або групи подій, в певний момент часу на елемент (групу елементів), або предметну область. Є жорстко структурованим, в залежності від мови високого рівня програмування і середовища розробки, який необхідний для досягнення мети розробки. Застосування – часткове або повне рішення виконання необхідних дій з даними (інформаційними потоками) необхідних для досягнення мети розробки, за умови досягнення головної мети розробки адитивного кібер-дизайну або на певних рівнях декомпозиції;

$\Re$ – множина відносин між базовими поняттями моделі, що дозволяють формально представляти класифікацію відносин, які визначають тип взаємодії між об'єктами предметної області та їх елементами. Визначимо в даній системі такі види відносин: «об'єкт-об'єкт», «елемент-об'єкт»,

«елемент-елемент», «об'єкт-елемент», за допомогою яких можна зробити класифікацію елементів предметної області. При цьому утворюються класи подій, встановлюються правила відносин між усіма учасниками процесу розробки адитивного кібер-дизайну, з використанням *LinqusticVariable* і *ContainerSolutions*, як невід'ємної частини множини $\Re$ і існуючих в будь-якому вигляді відносин визначених вище.

Виходячи з методів візуального об'єктно-орієнтованого програмування, для розроблюваної методу необхідно розглянути поняття багаторівневої абстракції існування двох видів підмножин на базі ЖЦ «Jump».

Припустимо, як аксіому, що будь-який P володіє одним (обов'язковим) або набором *Form*. Для зручності побудови моделі розділимо цю множину на дві підмножини: «*master*» і «*slave*». «*Master*» будемо називати $Form_n^{master}$, яка, в розробленому нижче методі нумерується через $n=1$ та є головною в цій ієрархії, а «підлеглою» буде $Form_n^{slave}$ (відповідно $Form_{n+1}^{slave}$, де $n=\overline{1,i}$). Виходячи з цього, за умови що $n\neq 0?$ можна представити у вигляді такого запису:

$$Form = Form_1^{master} \cup Form_n^{slave}. \tag{4.3}$$

при $n=0$ запис (4.3) прийме такий вигляд:

$$Form = Form_1^{master}. \tag{4.4}$$

Тобто, адитивний кібер-дизайн, що розробляється (моделюється), буде містити єдиний об'єкт взаємодії з користувачем і в даному випадку це буде $Form_1^{master}$.

З (4.3) представимо адитивний кібер-дизайн, як множину об'єктів візуальних $Form_n^{slave}$ різного рівня ієрархії, що підпорядковуються $Form_1^{master}$. Тому тотожним буде такий запис:

$$Form_1^{master} = \bigcup_{n=2} Form_n^{slavr}. \quad (4.5)$$

Відповідно запису (4.5) для $Form_1^{master}$ існує множина $Form_n^{slave}$ n-го рівня ієрархії, що мають унікальні імена:

$$Form_1^{master} = \{Form_1^{slave}, \ldots, Form_{n-1}^{slave}, Form_n^{slave}\}. \quad (4.6)$$

Вони взаємодіють між собою і з $Form_1^{master}$ через події, що належать множинам *EventForm* або *EventElement* з використанням *ContainerSolutions*.

Ведемо наступні ознаки відносин типу взаємодії між базовими поняттями такими як: «включає», «є елемент», «є параметр», «є подія», «є значення», «є ім'я», «є рішення» [212]. Нижче наведені приклади формалізації математичного опису відносин для кожної ознаки:

– «включає»

$$[(Form_1^{slave}, Form_2^{salve}, \ldots, Form_n^{slave}) \in Form_1^{master}] \in P, \quad (4.7)$$

або за умови (4.4) запис матиме вигляд:

$$(Form_1^{master}) \in P, \quad (4.8)$$

де P – візуальна модель адитивного кібер-дизайну.

Приймемо, що змодельований адитивний кібер-дизайн «включає» в себе множину *Form*, які мають ієрархію взаємозв'язків відповідно до (4.7)

– «є елемент»

$$\exists(ElementForm_1^1, ElementForm_2^1, \ldots, ElementForm_t^1) \in Form_1^{master}, \quad (4.9)$$

тоді можливий наступний запис, коли $(ElementForm_t^1) \in Form_1^{master}$ при $t = \overline{1,i}$, містить у собі ще й елементи графічного інтерфейсу (GUI):

$$\begin{array}{l}[(ElementForm_1'^{1(t)}, ElementForm_2'^{1(t)}, \ldots, ElementForm_q'^{1(t)}) \in \\ \in ElementForm_t^1] \in Form_1^{master}\end{array}. \quad (4.10)$$

Слід розуміти, що $ElementForm_1'^{1(t)}$, $ElementForm_2'^{1(t)}, \ldots, ElementForm_q'^{1(t)}$ при $(t) = \overline{1,i}$ і $q = \overline{1,i}$, «є елементами» підкласу, класу $ElementForm_t^1$, що входять в нього (для зручності реалізації групування за деякою умовою: логічною, інформаційною, змістовною).

– «є параметр» для $Form_1^{master}$ і $Form_n^{slave}$:

$$\begin{array}{l}((parameter_1, \ldots, parametr_p) \in ParameterForm_z^1) \in Form_1^{master} \equiv \\ \equiv ((parameter_1, \ldots, parameter_p) \in ParameterForm_z^1) \in Form_2^{slave}\end{array} \quad (4.11)$$

За умови, що використовується одне середовище розробки адитивного кібер-дизайну:

$$((parameter_1, \ldots, parametr_p) \in ParameterElement_h^1) \in ElementForm_t^1. \quad (4.12)$$

Множини *ParameterForm* і *ParameterElement* «є параметрами» ($parameter_p$ при $p = \overline{1,i}$), які описують його візуальні характеристики (розміри, відображення, колір), в залежності від його призначення. Для *Form* можна уявити у вигляді запису (4.11), для опису $ElementForm_t^1$ (4.12).

– «є подія» для $Form_n^{master}$ і $Form_n^{slave}$:

$$\begin{array}{l}((event_1, \ldots event_e) \in EventForm_c^1) \in Form_1^{master} \equiv \\ \equiv ((event_1, \ldots event_e) \in EventForm_c^2) \in Form_2^{slave}\end{array} \quad (4.13)$$

З (4.13) видно, що $event_e \in EventForm_c^1 \equiv event_e \in EventForm_c^2$ при $e = \overline{1,i}$ і $c = \overline{1,i}$, мають однаковий набір певних подій, який притаманний усім *Form*, в рамках одного адитивного кібер-дизайну.

$$((event_1, \ldots event_e) \in EventForm_c^1) \in Form_1^{master}. \quad (4.14)$$

Визначаємо належність того чи іншого $event_e$, як невід'ємної частини множини $EventForm_c^1$ і $ElementForm_t$ як «є подія».

– «є значення»

$$\exists(ValueForm_1, ValueForm_2, \ldots ValueForm_u) \in parameter_p,\ u = \overline{1,i}. \quad (4.15)$$

За умови, що $parameter_p \in ParameterForm_z$ з (4.11).

$$\exists(ValueElements_1, ValueElements_2, \ldots ValueElements_y) \in parameter_p. \quad (4.16)$$

За умови, що $parameter_p \in ParameterElement$ і $y = \overline{1,i}$ з (4.12).

– «є значення» (цілочисельне, лінгвістичне, булеве), яке задається обмеженням $parameter_p$, в залежності від середовища розробки і призначення того чи іншого параметра, який входить до множини $ParameterForm_z$ і $ParameterElement_h$.

– «є ім'я»

$$\exists! LingusticVariable_w \in LingusticVariable \forall event_e \in EventForm_c. \quad (4.17)$$

$$\exists! LingusticVariable_w \in LingusticVariable \forall event_e \in EventElement_r. \quad (4.18)$$

при $w=\overline{1,i}$ і $r=\overline{1,i}$

За умови, що хоч одна «Логічна змінна», яка описує вимоги, «є ім'я» необхідне для виконання події, яке належить множині $EventForm_c$ або $EventElement_r$.

– «є рішення»

$$\exists! ContainerSolutions_d \in ContainerSloution = \\ LingusticVariable_w \in LingusticVariable \; ; \; d=\overline{1,i}. \quad (4.19)$$

Розуміється так: існує хоч один «Контейнер рішень», який «є рішення» (процедуру, операцію з програмним кодом) для даної «Логічної змінної». Ґрунтуючись на даному допущенні, уявімо для наочності множини існуючих в моделі ЖЦ «Jump» основних базових понять і їх взаємодію у вигляді орієнтованого графа $L=(D,O)$. Де вершину графа D варто розуміти як умовне позначення базових понять, а O – тип відносин (взаємодії) між ними. Подання основних базових понять для моделі ЖЦ «Jump», з (4.2), представлено на рисунку 4.2.

Н – безліч функцій опису і трактування базових понять і відносин $\Re$, що визначає правила подання і опису структури даних предметної області моделювання. Тому набір формальних об'єктів (математичну структуру представлення об'єкту і опис його властивостей і подій) можна представити у вигляді математичної структури, яка дозволяє реалізувати в рамках об'єктно-орієнтованої моделі, достатньої для моделювання даних, в рамках певної предметної області. Розглянемо функції інтерпретації, як вид відносин між множиною понять.

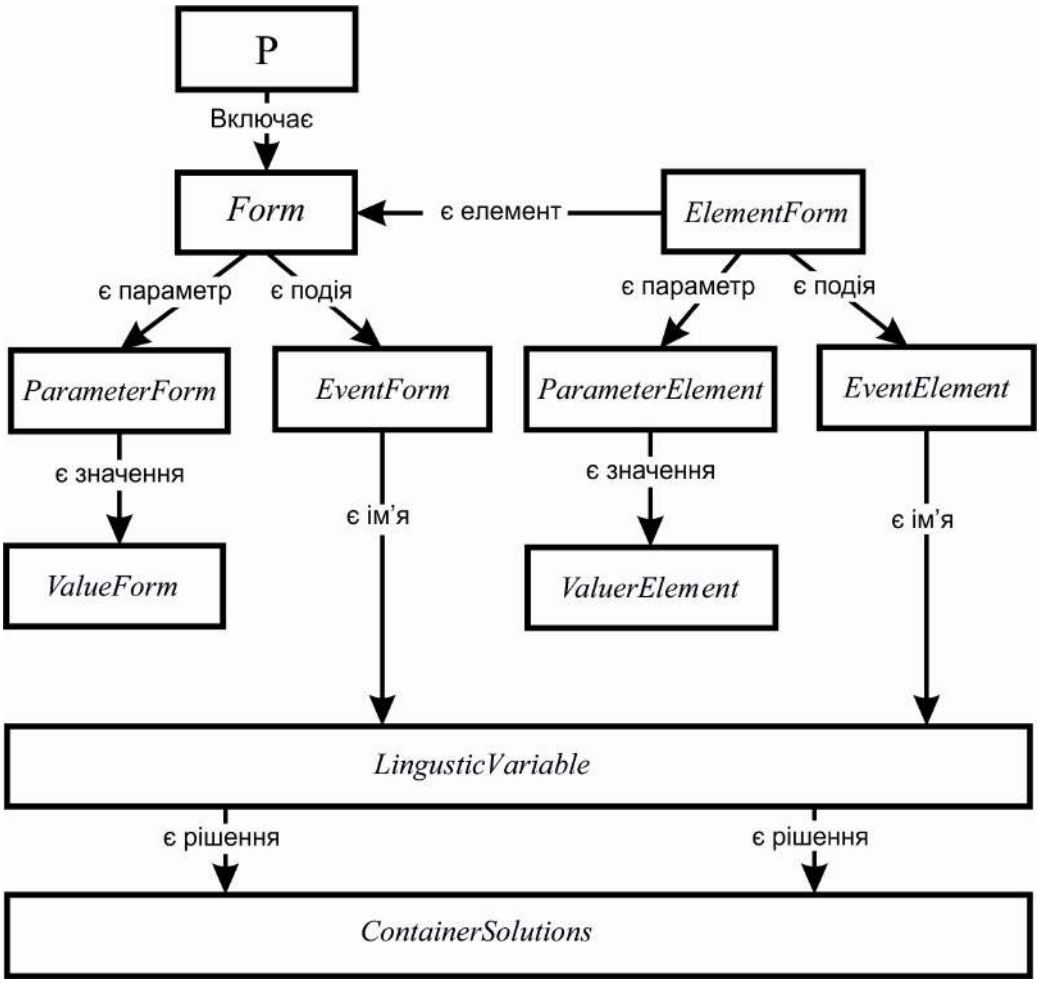


Рисунок 4.2 – Подання основних базових понять і відносин між ними

У запропонованій моделі ЖЦ розробки CPPS «Jump» визначимо такі поняття:

– «є значення»:

$$\begin{aligned} & f : Form_n^{master} \times ParameterForm_z \to ValueForm_u \Leftrightarrow \\ & \Leftrightarrow Form_n^{master} \times ParameterForm_z \xrightarrow{f} ValueForm_u \end{aligned}, \tag{4.20}$$

де для будь-якої впорядкованої пари $(Form_n^{master}, parameter_p)$ із $Form_n^{master} \times ParameterForm_z$ існує не більше одного елемента $value_v$ з $ValueForm_u$, який є $(Form_n^{master}, parameter_p, value_v) \in f$. На базі (4.20) наступний запис можна представити в такому вигляді:

$$value_v = f(Form_n^{master}, parameter_p), \quad (4.21)$$

де $Form_n^{master} \in Form$; $parameter_p \in ParameterForm_z^{master}$; $value_v \in ValueForm_u^{master}$.

Аналогічно можна описати «є значення» для $ElementForm_t^{master}$:

$$value_v = f(ElementForm_t^{master}, parameter_p), \quad (4.22)$$

де $ElementForm_t^{master} \in Form_n^{master}$; $parameter_p \in ParameterElement_h$; $value_v \in ValueElement_y$.

– «є подія» ґрунтується на структурі запису (4.20).

Представимо дане поняття у вигляді (4.23) для $Form_n^{master}$ і (4.24) відповідно для $ElementForm_t^{master}$:

$$name_m = f(Form_n^{master}, event_e), \quad (4.23)$$

де $Form_n^{master} \in Form$; $event_e \in EventForm_t$; $name_m \in ValueElement_y$.

$$name_m = f(ElementForm_t, event_e), \quad (4.24)$$

де $ElementForm_{t-1} \in ElementFormt_t$, $event_e \in EventrElement_r$; $name_m \in ValueElement_y$.

– «має рішення» для $ElementForm_t$ виразом (4.25), а для $EventForm_c$, відповідно (4.26).

$$cod_l = f(event_e, name_m), \quad (4.25)$$

де $event_e \in EventElement_r$; $name_m \in LingusticVarable_w$; $cod_l \in ContainerSolutions_d$

$$cod_l = f(event_e, name_m), \tag{4.26}$$

де $event_e \in EventForm_c$; $name_m \in LingusticVarable_w$; $cod_l \in ContainerSolutions_d$

K – множина вимог щодо обмеження цілісності. В даному контексті – це деяка логічна умова, що обмежує семантику обраного об'єкта або елемента, для уникнення втрати відносин всередині адитивного кібер-дизайну. В рамках даного дослідження виділимо два типи обмежень цілісності: явні (накладаються семантикою) і неявні (накладаються для підтримки і досягнення головної цілі моделювання адитивного кібер-дизайну самою моделлю ЖЦ «Jump»). Систематизуємо і подамо опис обмежень за такими типами.

Явні обмеження накладаються щодо списку параметрів $ParameterForm_z$ об'єкта $(Form), ParameterElement_h$ – елемента, як підкласу об'єкта ($ElementForm_t$) і подій для $EventForm_c$, $EventelEment_r$ (за визначенням безлічі допустимих значень для відповідних подань):

$$\begin{gathered}
ParmeterForm_z^1 = \{parameter_1^z, parameter_2^z, ..., parameter_p^z\};\\
ParmeterElement_h^1 = \{parameter_1^h, parameter_2^h, ..., parameter_p^h\};\\
EventForm_c^1 = \{event_1^c, event_2^c, ..., event_e^c\};\\
EventElement_r^1 = \{event_1^r, even_2^r, ..., event_e^r\};\\
ParameterForm_z = dom(parameter_{p_\text{спис}}^z);\\
ParameterElement_h = dom(parameter_{p_\text{спис}}^h);\\
EventForm_c^1 = dom(event_{e_\text{спис}}^c);\\
EventElement_r^1 = dom(event_{e_спис}^r);
\end{gathered} \tag{4.27}$$

де $dom(parameter_{p\_\text{спис}}^z)$, $dom(parameter_{p\_\text{спис}}^h)$, $dom(event_{e\_\text{спис}}^c)$, $dom(event_{e\_спис}^r)$ – домени відповідних характеристик (значень), що належать до

перерахованих (облікових) типів, які обираються із заздалегідь сформованого списку.

Прикладом можуть виступати деякі параметри $ParameterForm_z$, наприклад, параметр відображення $Form_n^{master}$, який може приймати значення $true$ або $false$, або параметр $Align$, якому притаманний $dom(parameter^z_{p\_спис})$, $dom(parameter^h_{p\_спис})$, що може приймати значення фіксоване середовищем розробки ($Top, Down, Center$ і т.д.). Ґрунтуючись на (4.27) конкретні значення зазначених характеристик можна обрати з представлених в (4.28) для відповідних доменів:

$$\begin{aligned} &parameter^1_p \in dom(parameter^z_{p\_спис});\\ &parameter^1_p \in ParameterForm_z;\\ &ParameterForm_z \in Form^{master}_1;\\ &parameter^1_p \in dom(parameter^h_{p\_спис});\\ &parameter^1_p \in ParameterElement_h; \end{aligned} \quad . \qquad (4.28)$$

Аналогічно вибору певного параметра опису (4.28) можна описати вибір події з $dom(event^c_{e\_спис})$ для $Form^{master}$ і $Form^{slaver}$, а також подій візуальних елементів $dom(event^r_{e\_спис})$:

$$\begin{aligned} &event^1_e \in dom(event^c_{e\_спис});\\ &event^1_e \in EventForm_t;\\ &EventForm_t \in Form^{master}_1;\\ &Form^{master}_1 \in P; \end{aligned} \quad . \qquad (4.29)$$

Ґрунтуючись на (4.28)–(4.29) визначимо параметри опису $Form^{master}_1$ і подій, які належать даній формі, а параметри візуальних елементів і їх властивості можна уявити у вигляді dom-списку, який обмежується

середовищем розробки і є фіксованим.

$$\begin{aligned} &event_e^1 \in dom(event_{e\_cnuc}^r); \\ &event_e^1 \in EventElement_1^r; \\ &EventElement_1^r \in CD_x^1; \\ &CD_1^x \in CF_f^1; \\ &CF_1^f \in Form_1^{master}; \\ &Form_1^{master} \in P; \end{aligned} \quad . \tag{4.30}$$

Неявні обмеження, що накладаються на модель формалізації опису P, в даному дослідженні, будуть введені на використання значень фізичних величин ($value_v$), які описують параметри $ParameterForm_z$ і $ParameterElement_h$. Розроблена модель опису адитивного кібер-дизайну дозволяє уявити обмеження по існуванню, які полягають в тому, що для існування в даній моделі параметра (4.28) необхідно і достатньо, що б він був і мав деяке значення $value_v$.

Аналогічно для (4.29) і (4.30) існування $event_e^1$ необхідно, що б вони мали $LibgusticVarable_w$, який посилається на певний і єдиний $ContainerSolution_d$, за (4.25)–(4.26). Наприклад, у пропонованій моделі без конкретного заданого значення ($value_v$), певний параметр $parameter_p^1 \in ParameterForm$ і $parameter_p^1 \in ParameterElement_h$ не буде використовуватися, або буде заданим за замовчуванням.

В якості обмежень за допустимими операціями в моделі, що розроблюється пропонуються:

– операція створення елементів множин, однойменних базовим поняттям моделі $\Im$, за (4.1), і їх зв'язків між собою, у відповідності з $\Re$ і H;

– операція зміни зазначених елементів множин, однойменних базовим поняттям;

– операція видалення зазначених елементів множини, однойменних

базовим поняттям, а, отже, залежних від $\Re$ і H, відповідно, елементів інших множин;

– операції вибірки, у відповідності із зазначеними умовами існування елементів, як частини множини однойменних базових понять запропонованої моделі.

Обмеження за унікальною ідентифікацією елементів множин, співвіднесених з однойменними базовими поняттями запропонованої моделі, досягається унікальністю іменування:

– для елементів множин *Form*, *ParameterForm*, *EventForm*, *ElementForm*, *ParameterElement*, *EventElement* – унікальністю відповідних в ієрархії імен, в рамах даної предметної області;

– для значень параметрів множин *ValueForm*, *ValueElement* – унікальністю імен, в рамках розглянутої предметної області;

– для елементів множин *LinqusticVariable* – унікальністю імен конкретних рішень, які посилаються на *ContainerSolutions*, що містить однозначне рішення для множин *EventForm* і *EventElement*.

Посилальна цілісність передбачає обов'язкову наявність об'єкта, на який посилається інший об'єкт і забезпечується завдяки запропонованим типам взаємодії між елементами множини.

L – мова представлення даних, яка може бути представлена теоретико-множинною моделлю, що не дозволяє непідготовленому фахівцю правильно і повно описати всі необхідні дані для адекватного представлення моделі і вхідної інформації для неї.

Тому була розроблена спеціальна мова, що поєднує в собі простоту синтаксису і опису даних. Побудована вона на базі близької до природної мови, яка дає зрозумілість і інтуїтивність використання CPPS, а також представлення даних моделюємої предметної області для всіх учасників. Більш детально розроблена мова моделювання і опис її синтаксису представлені в розділі 5.

На даному етапі позначимо такий умовний поділ метаопису мови

моделювання даних в розроблюваній моделі як: дані та метадані. Під мета даними, або інтенсіоналом предметної області, розуміється сукупність конкретних множин.

4.3 Метод синтезу візуальних компонентів і елементів структури адитивного кібер-дизайну

Для реалізації автоматизації розробки адитивного кібер-дизайну для керування процесами в складних організаційно-технічних виробничих об'єктах, необхідно розробити метод синтезу візуальних компонентів, на базі алгоритму функціонування, який буде забезпечувати повноту опису вхідних даних, логічний взаємозв'язок між ними і властивостями основних параметрів, притаманних елементам об'єктно-орієнтованих мов високого рівня програмування і інтерфейсу користувача.

Визначимо P – як порядкову множину призначених для користувача $Form_n$, які виконують умову:

$$\exists Form_n \in P \quad \text{при } n = \overline{1, i}, \tag{4.31}$$

де $i \le 40$ – максимальна кількість візуальних, призначених для користувача, форм інтерфейсу, з яких складається адитивний кібер-дизайн, що взаємодіє з головною (основною) формою ($Form_1^{master}$).

Ґрунтуючись на аналізі конструкторів інтерфейсу об'єктно-орієнтованих мов програмування, $Form_n$ можна представити у вигляді підмножин $Form_n\ Propertied\ and\ evens$ і $Components\ on\ Form_n\ structure$, що містять необхідну і достатню кількість даних для даного дослідження.

Таким чином, визначимо об'єкт $Form_n$, де $n = \overline{1,i}$, як множину:

$$Form_n \in \{Form_n Propertird\ and\ evens,\ Components\ on\ Form_n\ structure\}, \tag{4.32}$$

за умови, що $Form_n Propertird\ and\ evens$ і $Components\ on\ Form_n\ structure \neq \varnothing$.

Теорема 1. Про існування $Form_n$ тільки за умов $Form_n Propertird\ and\ evens$ і $Components\ on\ Form_n\ structure \neq \varnothing$.

Доказ: Припустимо $Components\ on\ Form_n\ structure = \varnothing$, що на $Form_n$, як робочому вікні інтерфейсу, не будуть присутні елементи візуалізації роботи з даними (GUI елементи: Button, Grid, і т.д.), а отже інтерфейс розроблюваного адитивного кібер-дизайну не має функціонального призначення і його розробка не є доцільною і це логічно. Виходячи з цього $Components\ on\ Form_n\ structure \neq \varnothing$. Цей вислів ліквідний так само для $Form_n Propertird\ and\ evens$ і дозволить уникнути парадоксу. Теорема доведена.

Визначимо $Form_n Propertird\ and\ evens$, як множину параметрів $Main\ propertied$ і значень $Properties\ parameters$, притаманних кожній $Form_n$, як невід'ємної частини $Form_n Propertird\ and\ evens$. Уточнімо, що кожному параметру множини $Main\ propertied$ належить хоч одна підмножина значень $Properties\ parameters$, але при цьому деякі значення підмножини можуть бути $Properties\ parameters = 0$. У результаті можна записати:

$$\begin{array}{l} \exists Properties\ parameters_s \in Properties\ parameters: \\ : Properties\ parameters_s \neq 0 \end{array}. \qquad (4.33)$$

Наведемо підмножини $Main\ propertied$ і $Properties\ parameters$, як впорядковану жорстко фіксовану послідовність параметрів $mp_x \in Main\ propertied$, де mp_x – параметри, які є невід'ємною частиною підмножини $Main\ propertied \in Form_n Propertied\ and\ evens$. Відповідно, множину значень $pp_a \in Properties\ parameters_s$, яка в свою чергу задовольняє

умові $Properties\ parameters_s \in Form_n Propertied\ and\ evens$. Введемо множину Z з елементами $(z_1, z_2, \ldots, z_k)$, як множину можливих варіацій значень (розмір $Form_n$ в піх, розташування «Top», «Left», зарезервовані слова «alNone – немає вирівнювання» і логічних «True» «False» і т.д.), що залежать від синтаксису опису $Form_n$ для певної мови програмування і середовища розробки.

Тому визначимо $\exists z_k \in Z : pp_q \neq 0$. В окремих випадках можливо $z_k = 0$, за умови, що $z_k \in Z \subseteq pp_q \in Properties\ parameters \neq \varnothing$. Також невід'ємною частиною множини є підмножина $Main\ events$, яка представляє набір параметрів $Form_n Propertird\ and\ evens$, таких як: «Crate», «Close», «Onclick» і т.д. Тому для їх опису введемо me_h, $(me_1, me_2, \cdots, me_h) \in Main\ events$, як всілякі допустимі дії над множиною $Form_n$. Множина параметрів обмежується правилами, заданими в тому чи іншому середовищі розробки. Обґрунтуємо існування me_h, як елементу безлічі $Main\ events$, наступним правилом існування:

$$Main\ events = \{me_h \in Main\ events : Form_n Propertied\ and\ event \neq \varnothing\}. \quad (4.34)$$

Кожному елементу me_h властива множина $Events\ on\ action\ whith\ Form_n$, яка складається з нескінченної кількості елементів ea_z – можливий набір імен *LinqusticVariable*, які посилаються на *ContainerSolutions* з певним конкретним «шаблоном», у вигляді програмного коду ($code_l$). Цей код містить в собі всі допустимі процедури/функції, які можуть виконуватися елементом me_h для кожної мови об'єктно-орієнтованого програмування. Наведемо це як:

$$Events\ on\ action\ whith\ Form_n \in \{ea_1, ea_2, \ldots, ea_z\}. \quad (4.35)$$

На базі висунутих припущень представимо множину $Form_n Propertird\ and\ evens$ як:

$$\{(\exists pp_a \cap mp_x) \in Main\ properties : Main\ properties \neq \varnothing) \& \\ \&(\exists ea_z \cap me_h \in Main\ properties \neq \varnothing)\} \subseteq Form_n Propertied\ and\ evens . \quad (4.36)$$

Вираз (4.36) не є достатнім і повним для вирішення завдання, поставленого в даному дослідженні, так як не враховує візуальні компоненти і компоненти роботи з відображення даних. Внаслідок цього, ведемо множина $Components\ on\ Form_n\ structure \in Form_n$ за умови:

$$(Components\ on\ Form_n\ structure \backslash Form_n\ Propertied\ and\ evens),$$

як множина елементів опису візуальних компонентів інтерфейсу, необхідних і достатніх для реалізації керування даними. $Components\ description_f \in Components\ on\ Form_n\ structure$, де $f = \overline{1,i}$ (кількість візуальних компонентів, які присутні на $Form_n$ і виконують певні функції). У свою чергу, для опису візуальних компонентів, ведемо підмножини $Preporties\ components_f$ і $Component\ events\ on\ action_f$, як невід'ємні частини множини $Components\ description_f$ (кількість різноманітних візуальних форм за допомогою яких можна працювати з інформацією).

Тоді будь-який візуальний елемент можна описати в наступному вигляді:

$$\{PPreportis\ components_f, Component\ events\ on\ action_f\} \in \\ \in Components\ description_f . \quad (4.37)$$

Для опису властивостей множин $Preporties\ components_f$ і

$Component\ events\ on\ action_f$ ведемо параметр $pc_m \in Preporties\ components_f$, де $m = \overline{1,i}$ кількість параметрів, які описують властивості об'єкта. Значення pc_m може приймати значення з множини Z, описаного вище. На відміну від $Main\ properties_t$, який єдиний для $Form_n$, кількість $Components\ description_f \in Components\ on\ Form_n\ structure$ повинна бути $f \geq 1$, де $f = \overline{1,i}$. В іншому випадку, при невиконанні цієї умови спрацьовує теорема 1.

Нескінченна різноманітність візуальних компонентів, які можуть використовуватися при формуванні множини $Components\ on\ Form_n\ structure$ вимагає накладення обмежень на $Preporties\ components_f$:

$$pc_1 \in Preporties\ components_1 \neq pc_m \in Preporties\ components_f,$$

при цьому $pc_1 = pc_m$ так само ґрунтується на тому, що безліч $Components\ description_f$ може містити однакові назви параметрів pc_m в множині $Preporties\ components_f$, але при цьому різні значення з безлічі Z, або навпаки.

Наведемо множину $Component\ events\ on\ action_f$, як впорядкування набір подій ce_w, які повинні виконувати обов'язкову умову:

$$ce_w \in Componenet\ events\ on\ action_f : Components\ description_f \neq 0. \quad (4.38)$$

Визначимо ce_w, як множину значень (набір «контейнерів» з програмним кодом), які можна застосувати для параметру pc_m, тоді логічним буде вислів:

$$\exists ce_w \in \forall pc_m : Components\ description_f \neq 0, \quad (4.39)$$

в іншому випадку це доводить неналежність $Components\ description_i \notin Components\ on\ Form_n\ structure$, тобто воно відсутнє як візуальний елемент в $Form_n$.

Для подальших досліджень і обґрунтування обраних рішень, в даному методі, для зручності математичних записів введемо скорочення:

$Form_n Propertid\ and\ evens$ – будемо надалі розуміти як $Form_n PE$;

$Main\ propertied$ – MP_n, де n – номер $Form_n$, котрій він належить, а параметри відповідні mp_x^n, де n – номер форми якій належить параметр, x – номер параметру в множині MP_n.

$Properties\ parameters$ – PP_n – множина значень, які може приймати mp_x^n у вигляді певних pp_a^n.

Варто врахувати, що кожному $mp_x^n \to pp_a^n$, при безлічі варіацій, які може приймати pp_a^n зберігається в множині Z, як описано вище. Допустимі дії, які може виконувати $Form_n$ у вигляді множини $Main\ events_n$ визначимо як ME_n, де n – номер форми. Множина ME_n може володіти унікальним набором параметрів, які позначимо як me_h^n, де n – ідентифікатор приналежності до ME_n, а h – номер параметра. Кожному ME_n відповідає множина $Events\ on\ action\ whith\ Form$ (EA_n), яка в свою чергу містить безліч значень ea_z^n, де n – ідентифікатор приналежності до EA_n, а z – номер параметра.

При цьому варто врахувати, що $me_h^n \to ea_z^n$, за умови, що n=n значення ea_z^n, для кожного параметра me_n^n, можуть обиратися з набору «контейнерів». $Components\ on\ Form_i\ stucture$ позначимо як CF^n, де n – номер $Form_n$, якій належить множина візуальних компонентів, які описані у вигляді $Components\ description_f$ (CD_x^n), де n показує приналежність до тієї чи іншої $Form_n$, а x – номер компонента $Form_n$, в структурі множини CF^n.

Уявімо структуру множини CD_x^n, як упорядковану безліч параметрів опису візуальних компонентів, для роботи з даними $Preporties\ components_f$, яку в подальшому позначимо через PC_y^{x}, де *x* показує приналежність до CD_x^n, а *n* – номер компонента в даній структурі. Уявімо PC_y^{x} як впорядкований набір параметрів pc_m^y, де $y = y$ номер приналежності до PC_y^{x}, а m – як порядковий номер параметрів масиву PC_y^{x}. $Component\ events\ on\ action_i$ позначимо як CE_z^x, де x – номер приналежності до певного CD_x^n, а z – порядковий номер. Наведемо CE_z^x як безліч подій, які може обробляти візуальний компонент ce_w^z, де $z = z$ з нумерації CE_z^x.

Множині подій ce_w^z може належати «контейнер рішень». Звернемо увагу, що множина CD_x^n може використовуватися безліч разів, при цьому візуальні форми можуть виконувати різні функції і описуватися різними параметрами, але можуть мати одне і те ж програмне рішення з «контейнера рішень».

Подамо адитивний кібер-дизайн кібернетичної складової (P) [213], що розробляється, у вигляді математичного опису структур $Form_n$ і зв'язків між ними, як представлено на рисунку 4.3.

Для визначення зв'язків введемо Ξ – як умову взаємодії між множинами $Form_n$. Надалі будемо розглядати такий запис $Form_1^{master} \xrightarrow{\Xi} Form_n^{slave}$, як взаємодію (передача даних, виклик і т.д.) $Form_1^{master}$ через подію me_h^n, або ce_w^n подію, які належать будь-якому GUI елементу, що належить $Form_1^{master}$ на $Form_n^{slave}$.

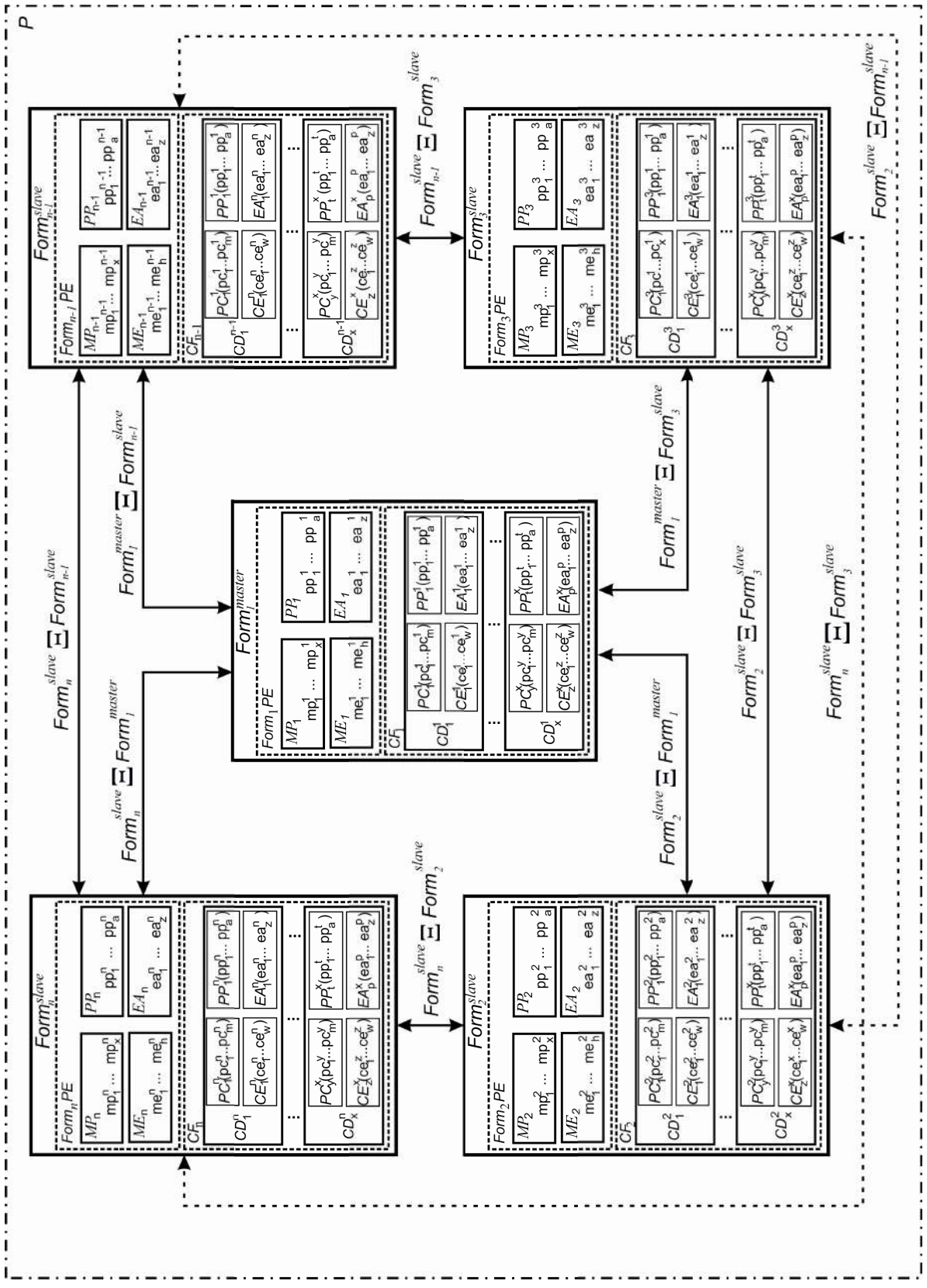


Рисунок 4.3 – Структурне подання $Form_n$ та зв'язків між ними в рамках P

Для математичного опису зв'язку $Form_1^{master}$ (в (4.40)–(4.41) – $Form_1$) через подію ea_a^n з $Form_n^{slave}$ (в (4.40)–(4.41) – $Form_i$) використовується ce_w^z і запис приймає частний вид:

$$Form_1^{master} \xrightarrow{Form_1 PE_1(ea_2^1 \in EA_1 : me_4^1 \in ME_1)\Xi\{(ce_w^z \in CE_z^x) \in CD_x^n\} \in CF_n} Form_n^{slave}. \qquad (4.40)$$

Наведена взаємодія одностороння, тобто подія me_4^1 належить $Form_1^{master}$ через «лінгвістичну змінну» ea_2^1, яка виконує якусь дію, з події ce_w^z графічного елементу CD_x^n, який належить CF_n на $Form_n^{slave}$, при цьому не повертає $Form_1^{master}$ ніяких результатів.

Надалі, для взаємодії між $Form_1^{master}$ і $Form_n^{slave}$, визначимо такий запис:

$$Form_1^{master} \xleftarrow{Form_1(парметри)} \Xi \xleftarrow[Form_n(параметри)]{} Form_n^{slave}, \qquad (4.41)$$

де під «параметрами» будемо розглядати, які елементи $Form_1 PE$ і CD_x^1 взаємодіють з елементами $Form_n PE$ або CD_x^n в обох напрямках. Кількість взаємозв'язків між $Form_1^{master}$ і $Form_n^{slave}$ теоретично не мають обмежень, але в рамках даних досліджень визначимо, що кількість $Form_n \le 40$ і її елементи ce_w^z не можуть посилатися самі на себе за допомогою «Контейнера рішень».

Визначимо детально підмножини, які входять в множину $Form_n$ і доведемо їх існування і основні властивості які вони описують.

Нехай P – адитивний кібер-дизайн, що розробляється, отже множина $Form_n \subseteq P$, якщо кожен елемент множини $Form_n$ є множиною P (рис. 4.2).

Теорема 2. $Form_n = P$ тоді і тільки тоді, коли одночасно $Form_n \subseteq P$ і $P \subseteq Form_n$, то $Form_n = P \Leftrightarrow Form_n \subseteq P$ і $P \subseteq Form_n$. Позначимо умову $P(Form_n)$: не існує ні одного елемента $Form_n$, який задовольняв би умові

теореми 2. Наприклад $P(Form_n)=\{Form_n \neq P\}$. У контексті даної умови можна сказати, що всі елементи $(Form_n PE, CF_n) \in Form_n \neq P$. Виходячи з цього, для даної умови, множину P, яка не містить в собі жодного загального елемента $(Form_n PE, CF_n) \in Form_n$, позначімо як порожню множину $\varnothing$. В даному дослідженні це означатиме, що множина $Form_n$ не має ніяких $\xrightarrow{\Xi}$ в множині P і дана множина вважається надмірною і, отже, $Form_n = \varnothing$.

Для доказу виконання умов об'єднання і перетину запропонованих множин, що входять в P, необхідно щоб вони задовольняли наступним властивостям:

- комутативність 2 і $2'$;
- асоціативність 3 і $3'$;
- дистрибутивність 4 і $4'$.

Теорема 3. Нехай $Form_n$, $Form_n PE$ і CF_n довільна множина властивостей і параметрів, які входять в множину P. Тоді для них справедливі такі рівності:

1. $Form_n \cup Form_n = Form_n$;
2. $Form_n \cup Form_n PE = Form_n PE \cup Form_n$;
3. $(Form_n \cup Form_n PE) \cup CF_n = Form_n \cup (Form_n PE \cup CF_n)$;
4. $(Form_n \cup Form_n PE) \cap CF_n = (Form_n \cap CF_n) \cup (Form_n PE \cap CF_n)$; (4.42)

$1'$. $Form_n \cap Form_n = Form_n$;

$2'$. $Form_n \cap Form_n PE = Form_n PE \cap Form_n$;

$3'$. $(Form_n \cap Form_n PE) \cap CF_n = Form_n \cap (Form_n PE \cap CF_n)$;

$4'$. $(Form_n \cap Form_n PE) \cup CF_n = (Form_n \cup CF_n) \cap (Form_n PE \cup CF_n)$.

Доказ. Кожне твердження цих властивостей випливає з визначення операцій над множинами і теореми 2. Для даних тверджень необхідно і достатньо довести 4-у властивість. Решта властивості операцій доводяться

аналогічно. Для спрощення математичних записів позначимо через Ψ ліву і Θ праву частина рівності 4. Покажемо, що обидві умови теореми 2 виконуються. Для цього спочатку доведемо, що $\Psi \subseteq \Theta$. Для цього візьмемо будь-який довільний елемент $\psi \in \Psi$. Тоді і тільки тоді ψ одночасно належить $Form_n \cup Form_nPE$ і CF_n, з даної умови $\psi \in Form_n \cup Form_nPE$ випливає, що відповідно $\psi \in Form_n$ або $\psi \in Form_nPE$. Виходячи з цього якщо $\psi \in Form_n$, то $\psi \in Form_n \cap CF_n$. Якщо $\psi \in Form_nPE$, то $\psi \in Form_nPE \cap CF_n$. Отже, в будь-якому випадку $\psi \in Form_n \cap CF_n$. Значить $\psi \in \Theta$.

Доведемо, що виконується зворотне включення $\Theta \subseteq \Psi$. Візьмемо довільний $\theta \in \Theta$, тоді можемо записати:

$$\theta \in (Form_n \cap CF_n) \cup (Form_nPE \cap CF_n) \Rightarrow \theta \in Form_i \cap CF_n,$$

чи

$$\theta \in Form_nPE \cap CF_n.$$

Якщо $\theta \in Form_n \cap CF_n \Rightarrow \theta \in Form_n$ і $\theta \in CF_z \Rightarrow \theta \in Form_i \cup Form_iPE$ та $\theta \in CF_n \Rightarrow \theta \in (Form_n \cup Form_nPE) \cap CF_n = \Psi$. Виходячи з цього якщо $\theta \in Form_nPE \cap CF_n \Rightarrow \theta \in Form_nPE$ і $\theta \in CF_n \Rightarrow \theta \in Form_n \cup Form_nPE$ та $\theta \in CF_n \Rightarrow \theta \in (Form_n \cup Form_nPE) \cap CF_n = \Psi$. Звідси $\theta \in \Psi$. З теореми 2 $\Psi = \Theta$. Решта властивостей операцій над запропонованими множинами перевіряються аналогічно.

Визначимо доповнення множини P через позначення $\overline{P}$, як різниці між універсальним множиною I і множиною P, звідси $\overline{P} = \{ Form_nPE : Form_nPE \in I$ і $Form_nPE \notin P \}$.

Теорема 4. Доведемо властивості різниці множин $Form_nPE$ і CF_n, за умови, що $\{Form_nPE, CF_n\} \in Form_n$, тоді вони повинні виконувати наступні властивості:

$$1.\ Form_n PE \cup \varnothing = Form_n PE;$$
$$2.\ Form_n PE \cup I = I; \qquad (4.43)$$
$$3.\ \overline{Form_n PE \cup CF_n} = \overline{Form_n PE} \cap \overline{CF_n};$$

$$1'.\ Form_n PE \cap \varnothing = \varnothing;$$
$$2'.\ Form_n PE \cap \mathrm{I} = P; \qquad (4.44)$$
$$3'.\ \overline{Form_n PE \cap CF_n} = \overline{Form_n PE} \cup \overline{CF_n}.$$

Доказ. Спочатку необхідно довести, що 3, 3′ виконують умови. Позначимо через Λ ліву і через Δ праву частину рівності. Покажемо, що $\Lambda \subseteq \Delta$ і $\Delta \subseteq \Lambda$.

Для будь-якого $\lambda \in \Lambda$ виконується $\lambda \notin Form_n PE \cup CF_z \Rightarrow \lambda \notin Form_n PE$, $\lambda \notin CF_n \Rightarrow \lambda \in \overline{Form_n PE}$ і $\lambda \in \overline{CF_n} \Rightarrow \lambda \in \Delta$. Отже, $\Lambda \subseteq \Delta$.

Для будь-якого $\upsilon \in \Delta$ виконується $\vartheta \in \overline{Form_n PE}$, $\lambda \in \overline{CF_n} \Rightarrow \lambda \notin \overline{Form_n PE}$ і $\lambda \notin CF_n \Rightarrow \lambda \notin Form_n PE \cup CF_n \Rightarrow \lambda \in Form_n PE$. Отже, $\Delta \subseteq \Lambda$.

Використовуючи розроблену структуру P, представлену на рис. 4.3, на базі проведених вище доказів, зробимо узагальнений математичний опис $Form_n$ у вигляді:

$$\begin{aligned} Form_n \in \{ & Form_n PE \subseteq ((mp_1^n,...,mp_x^n) \in MP_n \cup (pp_1^n,...,pp_a^n) \in PP_n) \wedge \\ & \wedge CF_n \subseteq [((pc_1^y,...,pc_m^y) \in PC_y^1 \cap (ce_1^z,...,ce_w^z) \in CE_z^1) \in CD_1^n], \\ & \ldots,[((pc_1^y,...,pc_m^y) \in PC_y^x \cap (ce_1^z,...,ce_w^z) \in CE_z^x) \in CD_x^n]\} \end{aligned} \qquad . (4.45)$$

Можна помітити, що запис (4.45) не описує всі можливі параметри, які притаманні $Form_n$, а має на увазі їх присутність у вигляді множини наборів параметрів $mp_1^n,...,mp_x^n$ і множини значень $pp_1^n,...,pp_a^n$, які можуть приймати

параметри. Залежно від середовища розробки та мови високого рівня програмування, набір параметрів може змінюватися. Але, в більшості випадків, значення, які вони можуть приймати, описуються або в десятковій системі числення і вимірюються в pixel (в яких задаються розміри графічного вікна, його місце розташування і розташування елементів всередині нього та їх розміри), або за допомогою логічних значень «True», «False» (які, описують чи відображається елемент, чи ховається за певних подій), а також за допомогою зарезервованих лінгвістичних слів, що дають можливість спростити виконання дій з візуалізації і відображення інтерфейсу користувача для зручності роботи («Align», «BorderStyle», «Caption» і. т.д.).

Отже, кожному параметру mp_x^k може відповідати одне або кілька значень pp_a^n (залежно від способу завдання значень). Виходячи з цього, в даному методі, буде запропоновано два окремих випадки відповідності.

Окремий випадок 1. Визначимо відповідність ζ між множинами MP_n і PP_n, як довільну підмножину їх добутків $MP_n \times PP_n$, тобто $\zeta \subseteq MP_n \times PP_n$. Зауважимо, що ця відповідність складається з упорядкованих пар. Кожна пара $(mp_x^n, pp_a^n) \in \zeta$ показує, що параметру $mp_x^n \in MP_n$ відповідає значення $pp_a^n \in PP_n$ при відповідності ζ. Приклад представлення відповідності для першого окремого випадку між параметром і значенням наведено на рис. 4.4.

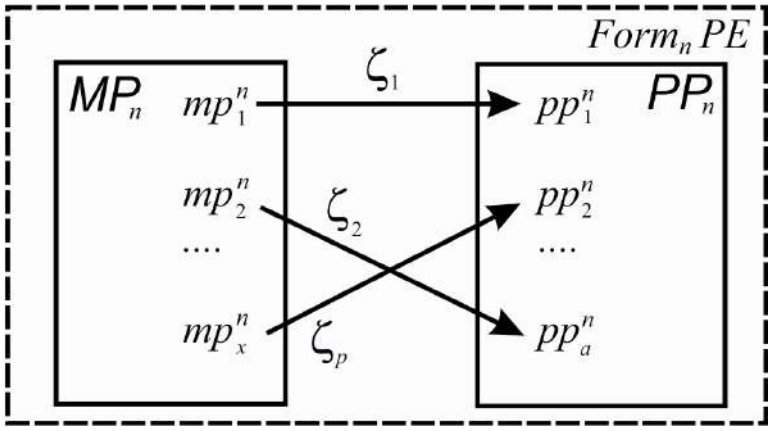


Рисунок 4.4 – Відповідність першого окремого випадку (коли параметру mp_x^n належить тільки одне значення pp_a^n)

Окремий випадок 2. Областю визначення відповідності ζ_1 назвемо множиною $Dom\zeta = \{mp_x^n \in MP_n : де\ існує\ значення\ pp_a^n,$ що $(mp_x^n, pp_a^n) \in \zeta\}$. Отже, дане визначення можна записати рекурсивно, множиною значень відповідності ζ, яке називає множину $\mathrm{Im}\zeta = \{pp_a^n \in PP_n : існує\ параметр\ mp_x^n \in MP_n,\ що\ (mp_x^n, pp_a^n) \in \zeta\}$.

На рис. 4.5 представлена відповідність для другого окремого випадку.

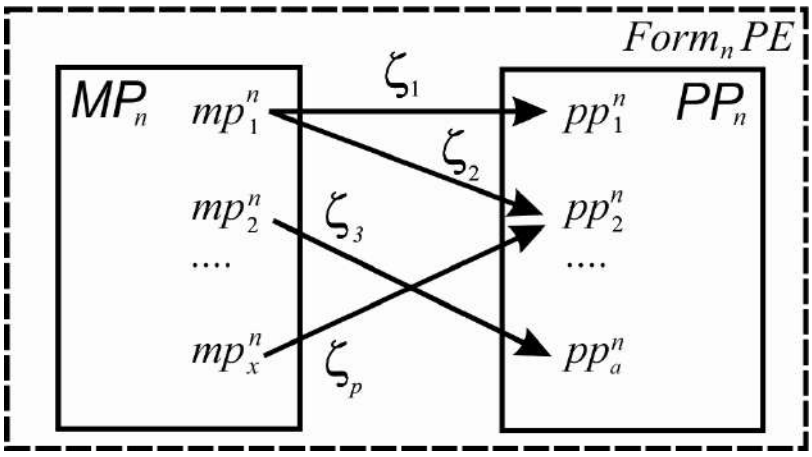


Рисунок 4.5 – Відповідність другого окремого випадку (коли параметру mp_x^n може належати множина значень pp_a^n)

Надалі позначимо відповідність між множиною параметрів $(mp_1^n,...,mp_x^n) \in MP_n$ і множиною значень $(pp_1^n,..,pp_a^n) \in PP_n$ через такий запис:

– для першого окремого випадку

$$mp_1^n \xrightarrow{\zeta} pp_1^n \text{ чи } \zeta : (mp_1^n) \in MP_n \to (pp_1^n) \in PP_n . \quad (4.46)$$

– для другого окремого випадку

$$mp_1^n \xrightarrow{\zeta} (pp_1^n, pp_2^n, pp_{a-1}^n) \quad (4.47)$$

чи

$$\zeta : (mp_1^n) \in MP_n \to (pp_1^n, pp_2^n, pp_{a-1}^n) \in PP_n . \quad (4.48)$$

Визначимо відповідність ζ, що виконує такі вимоги:

– певність, якщо $Dom\zeta = MP_n$;

– сюр'єктивність, якщо $\mathrm{Im}\zeta = PP_n$;

– однозначність, якщо кожному $mp_x^n \in Dom\zeta$ відповідає єдиний параметр pp_a^n з PP_n, тобто з $(mp_x^n, pp_a^n) \in \zeta$ і $(mp_x^n, pp_a^n) \in \zeta \Rightarrow pp_a^n = pp_1^n$;

– ін'єкційність, якщо різним параметрам $Dom\zeta$ відповідають різні параметри з PP_t, тобто $(mp_x^n, pp_a^n) \in \zeta$ і $(mp_x^n, pp_a^n) \in \zeta \Rightarrow mp_x^n = mp_1^n$. При виконанні цих вимог відображення матиме хоч одну відповідність (визначимо його частковою бієкцією).

Дане рішення може бути застосованим і достатнім для визначення множини значень $(pp_1^n,...,pp_a^n)$, які можуть приймати параметри $((mp_1^n...,mp_x^n) \in MP_n)$ для завдання параметрів візуалізації $Form_n$, тобто всі необхідні параметри, які дають можливість створити «порожній» шаблон Windows Form ("вікна"), що не враховує подій і реакцій при взаємодії з $Form_n$ (стандартні вбудовані події «Close», «Create» і реакції у вигляді «контейнера рішень»), в якому розробником адитивного кібер-дизайну реалізовані варіанти взаємодії через GUI з даними, параметри яких надходять з фізичної складової, або вирішують завдання функціоналу HMI на кібернетичному рівні, відповідно до вимог ТЗ і головної мети CPPS. Ґрунтуючись на вищесказаному, відповідності 1 і 2 окремих випадків знаходження $mp_1^n \xrightarrow{\zeta} pp_1^n$ і $mp_1^n \xrightarrow{\zeta} (pp_1^n, pp_2^n, pp_{a-1}^n)$, для опису $Form_n$ не підходять.

Для їх вирішення на рис. 4.3 визначені множини $(ME_n, EA_n) \in Form_n PE$, як невід'ємні частини множини $Form_n$. Розглядаючи запропоновану концепцію реалізації процесу розробки адитивного кібер-дизайну, визначимо призначення кожної множини:

$(me_1^n,...,me_h^n) \in ME_n$ – множина параметрів подій, які відповідають

множині $Form_n PE$, де me_h^n описується лінгвістичної константою («Close», «Create», і.т.д) і може розширюватися, в залежності від середовища розробки, в рамках однієї мови об'єктно-орієнтованого програмування.

Множина $(ea_1^n,...,ea_z^n) \in EA_n$ введена для завдання природного лінгвістичного опису події, наприклад, ea_z^n можна описати простими словами або виразами: «Закрити форму», «Закрити форму зі збереженням даних», «Закрити вікно з діалоговим вікном» і.т.д. Опис структури команд на природній мові представлено в розділі 5. Дане введення необхідно за твердженням $(mp_1^n ...,mp_x^n) \in MP_n \notin (me_1^n,...,me_h^n) \in ME_n$, внаслідок того, що елементи $pp_a^n \neq ea_z^n$ відповідають за способом представлення значень опису.

Тоді виникає необхідність існування множини $(z_1^o, z_2^o,...,z_q^o) \in Z_o$, де Z_o – множина можливих фрагментів рішень у вигляді програмного коду виконання певної функції і процедури, яка містить «шаблон», що залежить від мови програмування і середовища розробки. Надалі, в рамках даних досліджень, визначимо його як «*Container Solution*», який визначається шляхом завдання унікального «*Lingustic Variable*» ea_z^n. Наведемо приклад взаємодії множини $(me_1^n,...,me_h^n) \in ME_n$ за допомогою «*Lingustic Variable*» $(ea_1^n,...,ea_z^n) \in EA_n$ і множини $(z_1^o, z_2^o,...,z_q^o) \in Z_o$ «*Container Solution*» для середовища розробки Red Studio X3 (рисунок 4.6).

Визначимо існування зворотної відповідності $\zeta \subseteq ME_n \times EA_n$ до $\zeta' = \{(ea_2^n, me_1^n) : (me_h^n, ea_z^n) \in \zeta\}$. Зауважимо, що $\zeta' \subseteq EA_n \times ME_n$, тому ζ' – це відповідність між EA_n і ME_n. На рисунку 4.7 показана зворотна відповідність ζ.

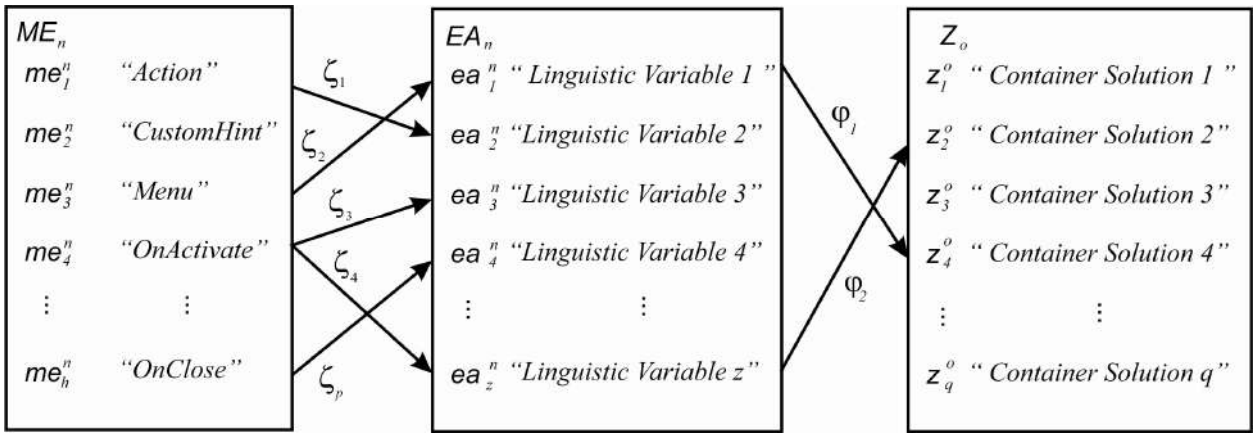


Рисунок 4.6 – Приклад взаємодії множин $(me_1^n,...,me_h^n) \in ME_n$ за допомогою «*Lingustic Variable*» $(ea_1^n,...,ea_z^n) \in EA_n$ і множини $(z_1^o, z_2^o,...,z_q^o) \in Z_o$ «*Container Solution*»

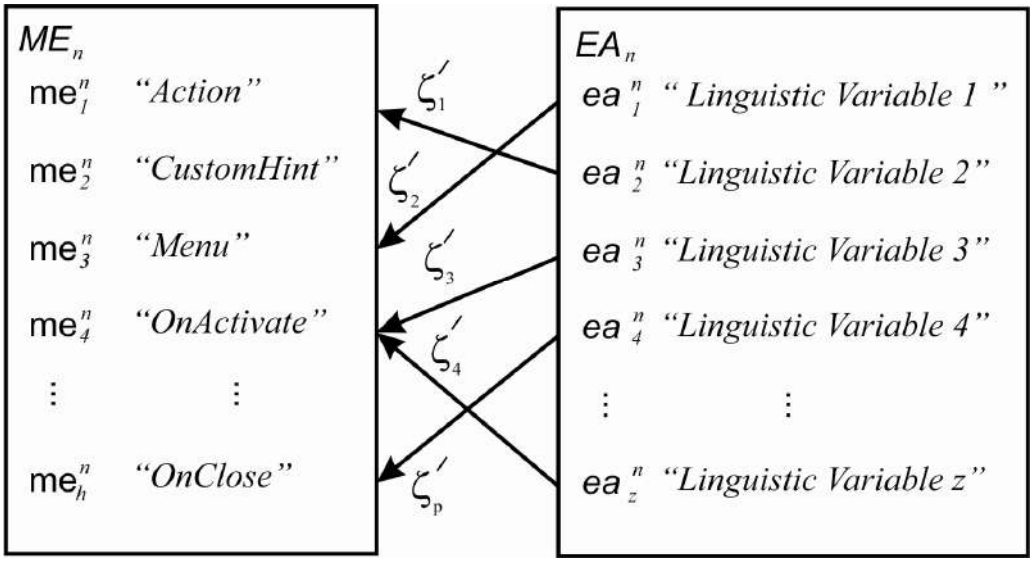


Рисунок 4.7 – Зворотна відповідність ζ

В даному випадку для рис. 4.6 математичний запис буде такий:

$$\zeta' = \{(ea_1^n, me_3^n),(ea_2^n, me_1^n),(ea_3^n, me_4^n),(ea_4^n, me_h^n),(ea_z^n, me_4^n)\}. \quad (4.49)$$

Для доказу існування такого рішення необхідно визначити композиційну відповідність. Отже, з рис. 4.6 композиційною відповідністю

$\zeta \subseteq ME_n \times EA_n$ і $\phi \subseteq EA_n \times Z_o$ назвемо таку відповідність $\chi \subseteq ME_n \times Z_o$, що $\chi = \{(me_h^n, z_q^o) : існує\ елемент\ ea_z^n \in EA_n,\ що\ (me_h^n, ea_z^n) \in \zeta\ \ і\ \ (ea_z^n, z_q^o) \in \varphi\}$. Надалі, в даному дослідженні, композиційну відповідність будемо записувати як $\chi = \zeta \circ \varphi$. Тоді композиційна відповідність для рис. 4.6 , є безліччю, яку представимо у вигляді системи:

$$\begin{cases} \zeta_2 \circ \varphi_1 = \{(me_{h-1}^n, z_4^0)\} \\ \zeta_4 \circ \varphi_2 = \{(me_h^n, z_2^o)\} \end{cases}. \tag{4.50}$$

Теорема 5. Про існування часткової бієкції ζ' і $\zeta \circ \varphi$. Якщо $\zeta \subseteq ME_n \times EA_n$ і $\varphi \subseteq EA_n \times Z_o$ – мають властивості часткової бієкції, то повинні відповідати таким властивостям:

1. ζ' є частковою біекцією між EA_n і ME_n;

2. $\zeta \circ \varphi$ є частковою біекцією між ME_n і Z_o.

Доказ:

1. Так як ζ є частково безумовно, то ζ' сюр'єктивне, а отже ζ сюр'єктивне, то ζ', яке всюди визначено хоч одним зв'язком. Дві властивості, що залишаються перевіряються аналогічно.

2. Оскільки ζ і φ всюди частково визначені, то їх композиція $\zeta \circ \varphi$ буде визначена хоча б в одному параметрі me_h^n безлічі ME_n. Так як ζ діє на ea_z^n *(«Lingustic Variable»)* множину EA_n і φ – на всі можливі z_q^o *«Container Solution»* безлічі Z_o, то $\zeta \circ \varphi$ є сюр'єктивною відповідністю. Однозначність і ін'єктивність доводиться аналогічно.

Розширимо математичну множину $CF_n \in Form_n$ (рис. 4.2) і визначимо його як набір підмножин, представлених на рис. 4.8.

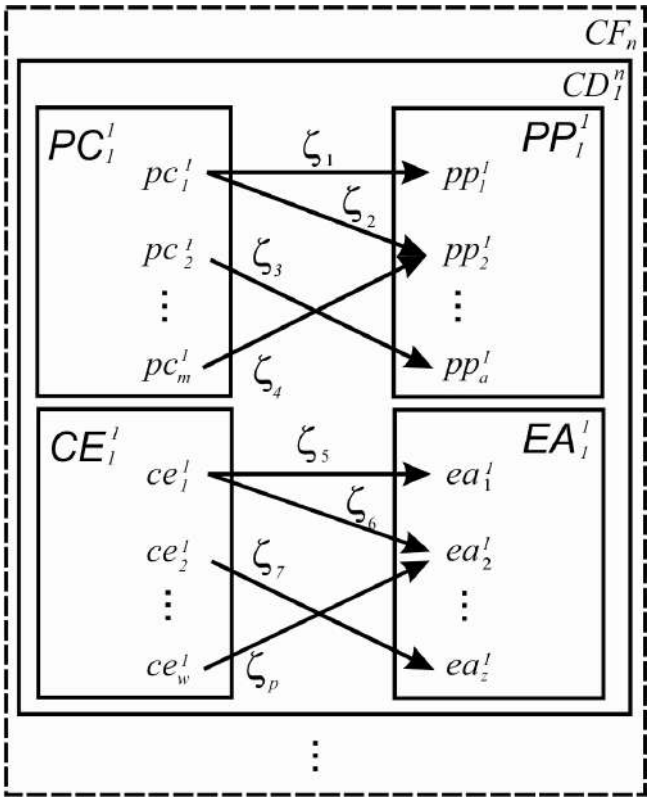


Рисунок 4.8 – Подання множини CF_n для опису властивостей елементів управління роботи з даними

Визначимо основні властивості множини CF_n, як $(CD_1^n,...,CD_x^n) \in CF_n$, яка містить в собі певну кількість візуальних графічних елементів HMI, та є необхідною і достатньою для реалізації всіх функцій і операцій досягнення головної мети розробки адитивного кібер-дизайну. Під візуальними компонентами будемо розуміти GUI елементи інтерфейсу («Grid», «Table», «Menu», «Button», і т.д.). Запис CD_1^n означає наступне: n – номер $CF_n \in Form_n$, якому належить візуальний компонент, а 1 – його нумерація в множині CF_n.

Це пояснюється тим, що множина опису візуальних компонентів $CD_1^n,...,CD_x^n$, які формують HMI взаємодії управління з даними (які, в свою чергу, можуть мати різне призначення, а, отже, різний набір параметрів як для графічного уявлення, так і для подій, які вони можуть обробляти). Зауважимо, що один елемент CD_x^n може бути використаний

безліч разів, але при цьому мати різні призначення відповідно до логіки, закладеної розробником. Тому визначимо, що $CD_x^n \in (PC_y^x, PP_t^x, CE_z^x, EA_p^x)$ мають властивості, доведені в теоремі 1-4 і так як:

$$CF_n \in \{CD_1^n \in (PC_y^1, PP_t^1, CE_z^1, EA_p^1), ..., CD_x^n \in (PC_y^x, PP_t^x, CE_z^x, EA_p^x)\},$$

є невід'ємною частиною $Form_n \in P$. Тому для спрощення реалізації даного методу і ґрунтуючись на особливостях завдання основних геометричних параметрів опису візуалізації елементів форм, приймемо що множина $CF_n \in (PP_t^n, EA_p^n) = Form_n PE(PP_t^n, EA_p^n)$ містить всі допустимі параметри для опису візуальних властивостей PP_t^n так і «лінгвістичних імен» EA_p^n, як форми ($Form_n PE$), так і компонентів CF_n, що дозволить в майбутньому використовувати одну базу знань. Використання даного рішення дозволяє, на базі доведених двох окремих випадків (рис. 4.4 - 4.5), реалізувати зв'язок між елементами як:

$$\begin{aligned} \varepsilon = \{&(pc_1^n, pp_1^n), (pc_1^n, pp_2^n), (pc_2^n, pp_a^n), (pc_m^n, pp_2^n), \\ &(ce_1^n, ea_1^n), (ce_1^n, ea_2^n), (ce_2^n, ea_z^n), (ce_w^n, ea_z^n)\} \end{aligned}, \qquad (4.51)$$

за умов, що всі властивості ζ, які доведені вище, зберігаються для ε. В наслідок цієї взаємодії множини $(ce_1^n, ..., ce_w^n) \in CE_n$ за допомогою лінгвістичного «імені» $(ea_1^n, ..., ea_z^n) \in EA_n$ і безлічі $(z_1^o, z_2^o, ..., z_q^o) \in Z_o$ «Container Solution». Останнє буде таке ж, як представлено на рис. 4.6 і збережо всі властивості і визначення для CF_n.

Спираючись на запропоновані рішення, в рамках даних досліджень, визначимо наступну форму запису для множини $(Form_n PE, CF_n) \in Form_n$ і призначення кожної з її підмножин:

– математичний опис множини $Form_n PE$:

$$Form_n PE \in \underbrace{((mp_1^n, mp_2^n, ..., mp_x^n) \in MP_n}_{\text{Множина параметрів}} \xrightarrow{\zeta_p} \underbrace{((pp_1^n, pp_2^n, ..., pp_a^n) \in PP_n)}_{\text{Множина значень}} \wedge$$

$$\wedge \underbrace{((me_1^n, me_2^n, ..., me_h^n) \in ME_n}_{\text{Множина подій}} \xrightarrow{\zeta_p} \underbrace{((ea_1^n, ea_2^n, ..., ea_z^n) \in EA_n)}_{\text{Множина "лінгвістичних імен"}} \xrightarrow{\varphi_e} \quad (4.52)$$

$$\xrightarrow{\varphi_e} \underbrace{((z_1^o, z_2^o, ..., z_q^o) \in Z_o)}_{\text{Множина "контейнерів рішень"}}.$$

– математичний опис множини CF_n:

$$CF_n \in (\underbrace{CD_x^n}_{\textit{елемент}} \in \underbrace{[((pc_1^x, pc_2^x, ..., pc_m^x) \in PC_y^x)}_{\text{Множина параметрів елемента}} \xrightarrow{\varepsilon_v} \underbrace{((pp_1^x, pp_2^x, ..., pp_a^x) \in PP_t^x}_{\text{Множина значень}} \wedge$$

$$\wedge \underbrace{((ce_1^z, ce_2^z, ..., ce_w^z) \in CE_z^x)}_{\text{Множина подій}} \xrightarrow{\varepsilon_v} \underbrace{((ea_1^p, ea_2^p, .., ea_z^p) \in EA_p^x)}_{\text{Множина "лінгвістичних імен"}} \xrightarrow{\varphi_e} (4.53)$$

$$\xrightarrow{\varphi_e} \underbrace{((z_1^o, z_2^o, ..., z_q^o) \in Z_o)}_{\text{Множина "контейнерів рішень"}}]).$$

Виходячи з цього $Form_n$ можна представити вигляді 4.54.

Варто врахувати, що кібернетична складова складається з безлічі Windows Form, кожна з яких містить елементи GUI, сукупність яких реалізують HMI та дозволяють контролювати і маніпулювати великими обсягами виробничих параметрів. Отже, для зручності візуального подання інформації в сучасних CPPS, необхідно використовувати групування даних за загальними логічними ознаками. Для цього застосовується багатовіконний інтерфейс, тобто, набір множини $Form_n$. Такий підхід вимагає розробки графічного методу подання та опису зв'язків між формами. Ґрунтуючись на (4.39) і (4.40) необхідно розробити графічне представлення $Form_n$ її видів і типів зв'язків між ними.

$$
\begin{aligned}
&Form_n^{master} \in \{Form_n PE \in ((mp_1^n, mp_2^n, ..., mp_x^n) \in MP_n) \xrightarrow{\xi} (pp_1^n, pp_2^n, ..., pp_a^n) \in PP_n)) \wedge ((me_1^n, me_2^n, ..., me_h^n) \in ME_n) \xrightarrow{\xi} \\
&\xrightarrow{\xi} ((me_1^n, me_2^n, ..., me_h^n) \in ME_n) \xrightarrow{\xi} ((ea_1^n, ea_2^n, ..., ea_z^n) \in EA_n) \xrightarrow{\varphi} ((z_1^o, z_2^o, ..., z_q^o) \in Z_o) \wedge \\
&\wedge CF_n \in (CD_x^n \in [((pc_1^y, pc_2^y, ..., pc_m^y) \in PC_y^x) \xrightarrow{\varepsilon} ((pp_1^t, pp_2^t, ..., pp_a^t) \in PP_t^x) \wedge ((ce_1^z, ce_2^z, ..., ce_w^z) \in CE_z^x) \xrightarrow{\varepsilon} \\
&\xrightarrow{\varepsilon} ((ea_1^p, ea_2^p, ..., ea_z^p) \in EA_p^x) \xrightarrow{\phi} ((z_1^o, z_2^o, ..., z_q^o) \in Z_o)]) \wedge (CD_{x+1}^n \in [((pc_1^y, pc_2^y, ..., pc_m^y) \in PC_y^{x+1}) \xrightarrow{\varepsilon} \\
&\xrightarrow{\varepsilon} ((pp_1^t, pp_2^t, ..., pp_a^t) \in PP_t^{x+1}) \wedge ((ce_1^z, ce_2^z, ..., ce_w^z) \in CE_z^{x+1}) \xrightarrow{\varepsilon} ((z_1^o, z_2^o, ..., z_q^o) \in Z_o)]) \wedge \\
&\wedge (CD_{x+2}^n \in [((pc_1^y, pc_2^y, ..., pc_m^y) \in PC_y^{x+2}) \xrightarrow{\varepsilon} ((pp_1^t, pp_2^t, ..., pp_a^t) \in PP_t^{x+2}) \wedge ((ce_1^z, ce_2^z, ..., ce_w^z) \in CE_z^{x+2}) \xrightarrow{\varepsilon} \\
&\xrightarrow{\varepsilon} ((ea_1^p, ea_2^p, ..., ea_z^p) \in EA_p^{x+2}) \xrightarrow{\varphi} ((z_1^o, z_2^o, ..., z_q^o) \in Z_o)]) \wedge (CD_{x+3}^n \in [((pc_1^y, pc_2^y, ..., pc_m^y) \in PC_y^{x+3}) \xrightarrow{\varepsilon} \\
&\xrightarrow{\varepsilon} ((pp_1^t, pp_2^t, ..., pp_a^t) \in PP_t^{x+3}) \wedge ((ce_1^z, ce_2^z, ..., ce_w^z) \in CE_z^{x+3}) \xrightarrow{\varepsilon} ((ea_1^p, ea_2^p, ..., ea_z^p) \in EA_p^{x+3}) \xrightarrow{\varphi} \\
&((z_1^o, z_2^o, ..., z_q^o) \in Z_o)]) \wedge ... \\
&\qquad\qquad \vdots \\
&\qquad\qquad \vdots \\
&\qquad\qquad \vdots \\
&... \wedge CF_n \in (CD_x^{n+1} \in [((pc_1^y, pc_2^y, ..., pc_m^y) \in PC_y^x) \xrightarrow{\varepsilon_{v-1}} ((pp_1^t, pp_2^t, ..., pp_a^t) \in PP_t^x) \wedge \wedge ((ce_1^z, ce_2^z, ..., ce_w^z) \in CE_z^x) \xrightarrow{\varepsilon_v} \\
&\xrightarrow{\varepsilon_v} ((ea_1^p, ea_2^p, ..., ea_z^p) \in EA_p^x) \xrightarrow{\varphi_e} ((z_1^o, z_2^o, ..., z_q^o) \in Z_o)])\}
\end{aligned}
\tag{4.54}
$$

В ході дослідження, для вирішення даного завдання, були проаналізовані наступні методи [214]:

– структурний аналіз Гейне-Сарсона;

– структурний аналіз проектування Йодана Де Марко;

– системи Джексона;

– структурні системи Варнье-Орра;

– аналіз і проектування СРВ Уорда-Меллора і Хатлі;

– інформаційне моделювання Мартіна;

– інформація сигнально-кодової конструкції (СКК), методологія Константайна.

Ґрунтуючись на специфіці завдань дослідження і складності візуального представлення інформацій та її кількість, запропоновано використовувати методологію СКК, яка представляється в блоковому вигляді, де кожен блок є модулем, тобто модель відносин ієрархій між формами кіберскладової CPPS. При цьому циклічні і умовні виклики модулів моделюються спеціальними вузлами, тому потоки повинні проходити через ці вузли. Міжмодульні зв'язки за даними і управлінню також моделюються спеціальними вузлами, які прив'язані до потоків (тобто до викликів модулів, які, в даному дослідженні, позначено як $Form_n$ і є невід'ємною частиною P). Базовими елементами опису є наступні типи модулів:

– модуль уявлення фрагменту зв'язку і його локалізацію на діаграмі;

– підсистема раніше визначених модулів, деталізованих за допомогою діаграм або математичного опису, що можуть повторюватися будь-яку кількість разів;

– бібліотека визначається поза проектом даної системи;

– область даних використовується для визначення модулів, які не містять області глобальних і розподілених даних.

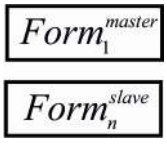
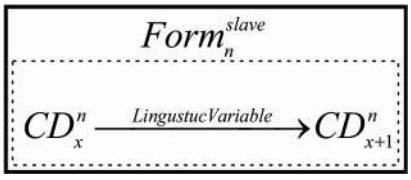
Обрана методологія СКК використовує власні елементи побудови та подання структурної карти, відповідно до стандартів IBM, ISO і ANSI. Для

даної моделі розроблені правила зв'язності і зчеплення модулів, що дає можливість спростити уявлення взаємодії між формами.

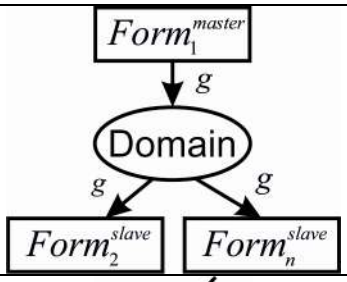
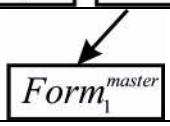
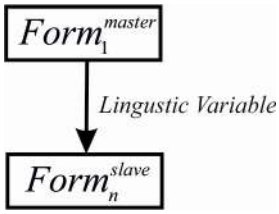
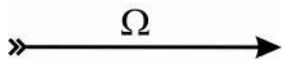
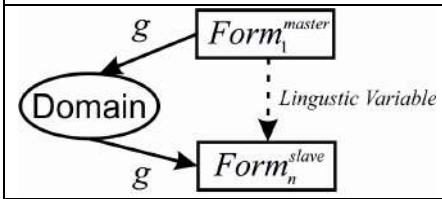
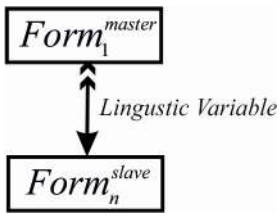
Виходячи з вищесказаного, в рамках даних досліджень, проведемо адаптацію методології Константайна. Це дозволить спростити розробку адитивного кібер-дизайну і дасть можливість розробнику представити візуальний опис і графічне представлення взаємодії між $Form_n$, на базі отриманих алгоритмів функціонування.

У таблиці 4.1 представлена модифікована модель графічного представлення методології Константайна візуалізації зв'язків між Windows Forms ghb розробки кібернетичної складової CPPS.

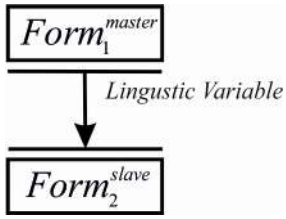
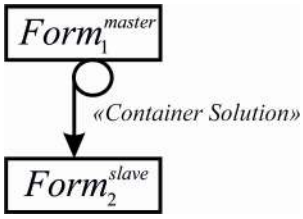
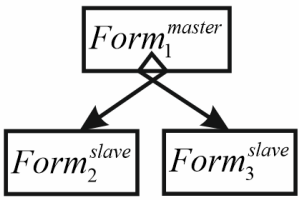
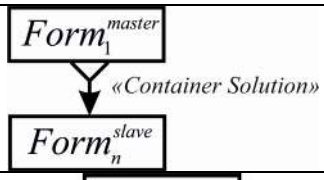
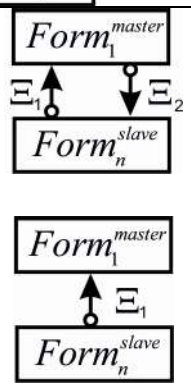
Таблиця 4.1 – Модифікована модель графічного представлення методології Константайна

Графічне представлення	Опис
1	2
	$Form_1^{master} \in [(Form_1 PE \in (MP_1 \in (mp_1^1,...,mp_x^1) \wedge$ $\wedge PP_1 \in (pp_1^1,...,pp_a^1) \wedge ME_1 \in (me_1^1,...,me_h^1)) \wedge$ $\wedge (CF_1 \in (CD_x^1 \in (PC_y^x (pc_1^y,...,pc_m^y) \wedge$ $\wedge PP_t^x \in (pp_1^t,...,pp_a^t) \wedge CE_z^x (ce_1^z,...,ce_w^z),...,)))]$
	$EA_n \in (ea_1^n,...,ea_z^n) \xrightarrow{\varphi} Z_o \in (z_1^o,...,z_q^o)$, як певний («Lingustic Variable») ea_z^n, якому належить множини або єдиний «Container Solution» $((z_1^o) \vee (z_2^o, z_3^o,...,z_q^o))$, що задовольняє умові φ
	$CF_n \in (CD_x^n \in (PC_y^x (pc_1^y,...,pc_m^y) \wedge$ $\wedge PP_t^x \in (pp_1^t,...,pp_a^t) \wedge CE_z^x (ce_1^z,...,ce_w^z)) \xrightarrow{\varphi_n}$ $\xrightarrow{\varphi_n} (CD_{x+1}^n \in (PC_y^{x+1} (pc_1^y,...,pc_m^y) \wedge PP_t^{x+1} \in$ $\in (pp_1^t,...,pp_a^t) \wedge CE_z^{x+1} (ce_1^z,...,ce_w^z)))$ при взаємодії $CD_x^n \to CD_{x+1}^n$, в рамках однієї з $Form_n^{master}$, за умов φ

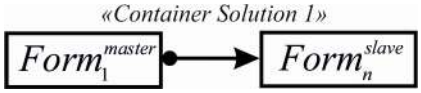
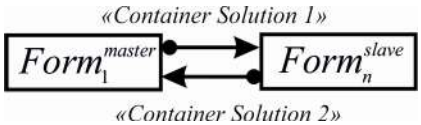
Продовження табл. 4.1

1	2
	Набір глобальних змінних, які передають необхідні дані між $Form_1^{master} \xrightarrow{g} Form_2^{slave}$ за умови, що $P \in (Form_1^{master}, Form_2^{slave})$ є необхідними для подальшої роботи g.
	Ініціалізація $Form_1^{master}$, за умови, що ніякі дані на неї не передаються
	Посилання на елемент $CD_x^n \in (CE_z^x \in (ce_1^n,...,ce_w^n))$, де ce_w^n – подія, яка є необхідною для вирішення поставленого завдання, на базі «Lingustic Variable» $ea_z^p \in EA_p^x$, за допомогою z_q^o «Container Solution»
	Зв'язок за даними (змінним) Ξ, типи даних представлені в параметричній моделі
	Зв'язок за управлінням Ψ візуальними елементами, подіями CD_x^n, через «Container Solution»
	Потік – виклик (Ω) $Form_1^{master}$ в $Form_2^{slave}$. Використовується для графічного позначення виконання асинхронних і синхронних операцій, що дозволяє виконувати тривалі операції і паралельно з цим працювати з інтерфейсом $Form_1^{master}$.
	Паралельний виклик – робота з асинхронними операціями, що дозволяє при тривалих розрахунках взаємодіяти з $Form_1^{master}$, а результат виконання буде виведений в $Form_2^{slave}$.
	Послідовний виклик – робота з графічним інтерфейсом $Form_1^{master}$ блокується для користувача, поки тривала операція не буде виконана, результат виводитися в $Form_2^{slave}$. Після завершення роботи користувача з $Form_2^{slave}$, за результатами якого вона буде закрита, користувач може продовжити роботу з $Form_1^{master}$.

Продовження табл. 4.1

1	2
	$Form_1^{master}$ викликає $Form_2^{slave}$ як співпрограму, яка використовує принцип багатопотокового коду, при цьому не вимагаючи наявності декількох потоків, а може виконуватися всередині одного потоку. В рамках даних досліджень, при розробці CPPS, співпрограми корисні для реалізації методів кінцевих автоматів, моделей акторів і генераторів (ітераторів).
	Цикл – призначений для організації багаторазового виконання набору інструкцій, поки не будуть виконані умови. «Container Solution», що містить елемент циклу з умовою, що належить $Form_1 PE$ або елементу CD_x^1, які по завершенню ініціалізують передачу результату в $Form_2^{slave}$.
	Розгалуження «Container Solution» забезпечує виконання певних наборів команд (команди), за умови істинності деякого логічного виразу. Може існувати з однією гілкою, з двома гілками, з декількома умовами. При виконанні «Container Solution», який належить $Form_1^{master}$, в залежності від виконання умов істинності може ініціалізувати передачу результату в $Form_2^{slave}$ або $Form_3^{slave}$.
	Одноразове виконання «Container Solution», яка належить $Form_1^{master}$ в чітко заданій послідовності програмного коду викликає $Form_n^{slave}$.
	Графічне уявлення взаємодії (4.40): - двостороннє $\xrightarrow{Form_1^{mater}(значення)} \Xi \xleftarrow{Form_n^{slave}(значення)}$ - одностороннє $Form_1^{master}(значення)\Xi \xleftarrow{Form_n^{slave}(значення)}$ через глобальну змінну Ξ, в рамках одного P, яка визначена в будь-якій $Form_1^{master},...,Form_n^{slave}$, або належить CD_x^n.

Продовження табл. 4.1

1	2
 	Зв'язок по управлінню (4.39) - односторонній через подію, що належить $Form_1^{master}$ на $Form_n^{slave}$ $Form_1^{master}[FP_1(ea_z^p \in EA_p^1 : me_h^d \in (ME_d^1)] \xrightarrow{\Psi_i^1}$ $\xrightarrow{\Psi_i^1} Form_n^{slave}[CF_z^n\{(ce_w^z \in CE_z^n : pc_l^f \in (PE_f^n)) \in CF_z^n\}]$ - двосторонній $Form_1^{master}[FP_1(ea_z^p \in EA_p^1 : me_h^d \in (ME_d^1)] \xrightarrow{\Psi_1^1} \Xi$ $\xleftarrow[\Psi_n^2]{} Form_n^{slave}[CF_z^n\{(ce_w^z \in CE_z^z : pc_l^f \in (PE_f^n)) \in CF_z^n\}]$ де Ψ_l^i – «Lingustic Variable», який вказує на той чи інший z_q^o «Container Solution».

Відповідно до таблиці 4.1, в даному методі, пропонується виділити 2 основні групи зв'язків:

– група зв'язків всередині $Form_n$:

ζ_n – зв'язок існування, що показує якому параметру або події $Form_n$ належить певне значення (чисельне, зарезервоване, логічне) або «Lingustic Variable»;

φ_n – зв'язок, що показує, якому певному «Lingustic Variable» належить «Container Solution», за умови існування ζ_n.

– група зв'язків між $Form_1^{master},...,Form_n^{slave}$ в рамках одного P:

Ξ – зв'язок за даними якого, здійснюється ведення глобальної змінної певного типу (int, char), яка містить в собі дані, що отримуються у результаті виконання «Container Solution» і слугують для подальшої обробки в наступному «Container Solution»;

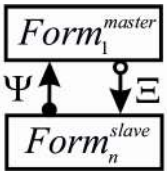
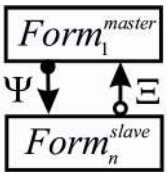
Ψ – зв'язок з керування через використання подій, притаманних елементам CD_x^n, які задаються за допомогою «Lingustic Variable» і містять певний «контейнер рішень» у вигляді програмного коду;

Ω – потік, виклик для вирішення тривалих «Container Solution» (при виконанні складних розрахункових завдань, обробки великих обсягів даних,

виконання завдань на рівнях $AEofS_i$ і передачі їх на рівень декомпозиції вище), які необхідно реалізувати за допомогою асинхронних і синхронних операцій, в даному випадку використовується синтез «контейнерів рішень».

Необхідно врахувати можливість існування комбінованих типів зв'язків між $Form_1^{master},...,Form_n^{slave}$. Коли використовується Ξ для передачі даних в $Form_n$ викликається відповідна реакція у вигляді ініціалізації CD_x^n на $Form_{n+1}$ через Ψ. Графічне представлення комбінованих типів зв'язків представлено в таблиці 4.2.

Таблиця 4.2 – Модель графічного представлення комбінованих типів зв'язків

Графічне представлення	Опис
	$Form_1^{master}$ взаємодіє за допомогою глобальної змінної ϖ, яка містити числове або логічне значення (тип зв'язку Ξ) на $Form_n^{slave}$, яка після обробки в «контейнері рішень» повертає результат обробки «контейнера рішень» на $Form_1^{master}$ у вигляді візуального відображення або реакцій з елементу CD_x^1 (тип зв'язку Ψ).
	$Form_1^{master}$ через подію графічного елементу інтерфейсу CD_x^1 з «контейнера рішень» (тип зв'язку Ψ) викликає $Form_n^{slave}$ за певною подією, до якої належить «контейнер рішень». Після виконання отриманий результат повертається через глобальну змінну ϖ (тип зв'язку Ξ) на $pc_n^i \in PC_n^i$ (без використання елементів CD_{x+1}^1 на $Form_1^{master}$).

4.4 Формальне подання властивостей і подій форм та їх компонентів

На базі запропонованого математичного опису множини $Form_nPE$ (4.52), як співвідношення $MP_n \xrightarrow{\zeta} PP_n$ і $ME_n \xrightarrow{\zeta} EA_n \xrightarrow{\varphi} Z_o$, а також графічних елементів CD_x^n (4.22) у вигляді співвідношень $PC_y^x \xrightarrow{\varepsilon} PP_t^x$ і

$CE_z^x \xrightarrow{\varepsilon} EA_p^x \xrightarrow{\varphi} Z_o$, в рамках даних досліджень, необхідно провести формальний опис значень і «лінгвістичних змінних», які відповідають опису властивостей і подій властивих формі і графічним елементам (GUI).

Для цього необхідно систематизувати типи подання значення і «лінгвістичні змінні», в залежності від їх задання, представлених в алгоритмі функціонування CPPS. В ході аналізу сучасних об'єктно-орієнтованих середовищ розробки було виділено такі типи представлення значень (табл. 4.3).

Таблиця 4.3 – Типи представлення значень PP_t^x, PP_n, EA_p^x, EA_n

Тип уявлення значень	Застосування	Приклад
1	2	3
Текстовий $(a,b,c,...,z)$ $(a,б,c,...,я)$	PP_t^x, PP_t	$pp_a^t \in PP_t^x \equiv PP_n = Name\ of\ the\ main\ form$ (назва форми для користувача задається розробником)
Лінгвістичний *(aaab bbcc,...,aabb)*	EA_p^n, EA_n	$ea_z^p \in EA_p^z =$" *AllCloseForm*" лінгвістичне ім'я контейнера рішень
Логічний $(true, false)$	PP_t^x, PP_n	$pp_a^t \in PP_t^x \equiv PP_n = false$ (невикористання певного параметра, обирається розробником)
Цілочисельний $(0,1,2,...,n)$ $(0,1,2,...,n)$	PP_t^x, PP_n	$pp_a^t \in PP_t^x \equiv PP_n = 380$ (розмір довжини $Form_n$, координати розташування)
Цілочисельний негативний (-1)	PP_t^x, PP_n	Застосовується для індексації, при значенні (-1) – індексація відсутня, а якщо є, то тип цілочисельний
Текстове слово, словосполучення $(aa,ab,...,aabb)$ (зарезервовані середовищем розробки)	PP_t^x, PP_n	$pp_a^t \in PP_t^x \equiv PP_n = clBtnFace$ (визначення значення кольору фону для $Form_n$ в середовищі RAD Studio XE3)

Текстовий тип – використовується для визначення призначення параметрів, які в більшості випадків, при розробці інтерфейсу адитивного

кібер-дизайну носять інформаційний статичний характер.

Лінгвістичний – привласнення унікального визначення для «лінгвістичного імені» EA_n певній події, яку потрібно обробити за допомогою «контейнера рішень». Z_o задається розробником. «Лінгвістичне ім'я» неповторне в множині EA_n.

Булевий тип – дозволяє розробнику вибрати активність / пасивність обраного параметра. У більшості випадків він використовується для завдання значень параметрів $mp_x^n \in MP_n$ і $pc_m^y \in PC_y^n$ для опису $Form_n$ і елементів CD_x^n.

Цілочисельний тип – використовується для опису значень, які задаються користувачем в *pix*. В основному служать для визначення розмірів $Form_n$, а також розмірів і координат розташування CD_x^n, в рамках $Form_n$.

Цілочисельний негативний тип – використовується для індексації кількості можливого вибору в реалізації графічного інтерфейсу адитивного кібер-дизайну. В основному застосовується при нумерації image, зустрічається як значення параметрів $mp_x^n \in MP_n$ і $pc_m^y \in PC_y^n$.

Текстове слово або словосполучення – значення у вигляді слів або скорочень, які суворо фіксовані в середовищі розробки. Кожному певному параметру $mp_x^n \in MP_n$, $pc_m^y \in PC_y^n$ може належати певний набір двох або більше значень pp_a^t [215].

Для спрощення формалізації подання значень PP_t^x, PP_n, EA_p^x, EA_n пропонується згрупувати їх в дві групи за ознаками опису параметрів форм MP_n і елементів PC_y^x, а також за подіями форм $me_h^n \in ME_n$ і подіями елементів $ce_w^z \in CE_z^n$:

– формальне подання значень PP_t^x, PP_n форм і графічних елементів:

1. Цілочисельне:

$$mp_x^n \vee pc_m^y = \begin{cases} a^1, \textit{якщо } a_i \le pp_a^n \le [\textit{значення}] \\ a^2, \textit{якщо } [\textit{значення}] \le pp_a^n \le [\textit{значення}] \\ \vdots \\ a^2, \textit{якщо } [\textit{значення}] \le pp_a^n \le a_o \end{cases}, \quad (4.55)$$

де $mp_x^n \vee pc_m^y$ – позначення $a-ого$ параметра для MP_n і PC_y^x відповідно;

$a^1.a^2,\ldots,a^n$ – ідентифікатори діапазонів значень;

a_i, a_j – граничні значення, які: $a_i \to \min$; $a_j \to \max$;

$[\textit{значення}]$ – виділені порогові значення параметра.

2. Булеве:

$$mp_x^n \vee pc_m^y = \begin{cases} a^1, \textit{якщо } pp_1^n = [\textit{false}] \\ a^2, \textit{якщо } pp_2^n = [\textit{true}] \end{cases}, \quad (4.56)$$

де $mp_x^n \vee pc_m^y$ – позначення $a-ого$ параметра для MP_n і PC_y^x відповідно;

$a^1.a^2$ – ідентифікатори діапазонів значень;

[*true*, *false*] – визначення логічного значення.

3. Текстове слово або словосполучення:

$$mp_x^n \vee pc_m^y = \begin{cases} a^1, \textit{якщо } pp_1^n = [\textit{слово, словосполучення}] \\ a^2, \textit{якщо } pp_2^n = [\textit{слово, словосполучення}] \\ \vdots \\ a^n, \textit{якщо } pp_a^n = [\textit{слово, словосполучення}] \end{cases}, \quad (4.57)$$

де $mp_x^n \vee pc_m^y$ – позначення $a-ого$ параметра для MP_n і PC_y^x відповідно;

$a^1.a^2,\ldots,a^n$ – ідентифікатор значень;

[*слово*, *словосполучення*] – визначення тексту або словосполучення.

4. Цілочисельне негативне:

$$mp_x^n \vee pc_m^y = \begin{cases} a^1, якщо\ a_i = pp_a^n = -1 \\ a^2, якщо\ [1] \le pp_a^n \le [значення] \\ \vdots \\ a^n, якщо\ [значення] < pp_a^n \le a_j \end{cases}, \quad (4.58)$$

де $mp_x^n \vee pc_m^y$ – позначення $a-ого$ параметра для MP_n і PC_y^x відповідно;

$a^1.a^2,\ldots,a^n$ – ідентифікатори діапазонів значень;

a_i, a_j – граничні значення, які: $a_i = -1$; $a_j \to \max$;

$[значення]$ – виділені порогові значення параметра.

Формальне подання подій форм ME_n і графічних елементів CE_z^x.

1. Лінгвістичне:

$$me_h^n \vee ce_w^z = \begin{cases} a^1, якщо\ ea_1^p = [слово] \\ a^2, якщо\ ea_2^p = [слово] \\ \vdots \\ a^n, якщо\ ea_z^p = [слово] \end{cases}, \quad (4.59)$$

де $me_h^n \vee ce_w^z$ – позначення $n-ої$ події для ME_n і CE_z^x відповідно;

$a^1, a^2,\ldots,a^n$ – ідентифікатори діапазонів значень;

[*слово*] – визначення, «лінгвістичне ім'я» контейнера рішень, яке є неповторним і унікальним в EA_p.

На базі вищевказаного, наведемо приклад математичного опису та графічного представлення зв'язків з таблиць 4.1 і 4.2.

Нехай існує P, як адитивний кібер-дизайн керування процесами в складних організаційно-технічних виробничих об'єктах, який складається з двох діалогових форм. $Form_1^{master}$ виступає у вигляді головної форми розроблювального кібер-дизайну, яка представлена множиною $Form_1^{master} PE$ (описує необхідний і достатній набір параметрів ME_n і їх значень PP_t^n, які

допустимі для кожного параметра $mp_1^n,..,mp_x^n$, при цьому кількість і назва параметрів залежить від обраного середовища розробки), якій відповідає певний $pp_1^t,...,pp_a^t$ набір значень допустимих кожному mp_x^n. Отже, можна записати наступну відповідність:

$$MP_n \xrightarrow{\zeta_i} PP_n \text{ як } \underbrace{\begin{cases} mp_1^n(Caption) \xrightarrow{\zeta_1} \\ mp_2^n(ClientHeight) \xrightarrow{\zeta_2} \\ \vdots \\ mp_x^n(Visible) \xrightarrow{\zeta_n} \end{cases}}_{параметр} \underbrace{\begin{cases} pp_1^n(textname) \\ pp_2^n(1,2,..,1200\,pix) \\ \vdots \\ pp_a^n(true, false) \end{cases}}_{значення}, \quad (4.60)$$

Для спрощення представлення $MP_n \xrightarrow{\zeta_i} PP_n$ введемо поняття приналежності, з точки зору опису присвоєння значень, які задаються розробником залежно від вимог замовника. Введемо визначення «основних параметрів» mp_x^n, які заповнюються обов'язково за умови, що середовище розробки не зможе побудувати $Form_n$ і «не основних» $\overline{mp}_n^k$, в яких присутність значень pp_a^n можна не вказувати (вказується в разі потреби або формується середовищем розробки за замовчуванням). Приклад математичного запису представлень у (4.50) запишемо як:

$$MP_n \xrightarrow{\begin{cases} mp_1^n \xrightarrow{\zeta_1=textname} pp_1^n \\ mp_2^n \xrightarrow{\zeta_2=640\,pix} pp_2^n \\ \overline{mp}_3^n \xrightarrow{\zeta_3=\text{bsSizeable}} pp_3^n \\ \dots \\ mp_x^n \xrightarrow{\zeta_n=true} pp_a^n \end{cases}} PP_n. \quad (4.61)$$

Це дозволяє нам описати для $Form_nPE$ всі необхідні параметри реалізації графічного відображення $Form_n$, які можна задати наступним чином:

$$Form_1^{master}\left[Form_1PE(MP_k^1)\xrightarrow{mp_1^k=textname,mp_2^k=640,mp_3^k=beSizeable,...,mp_x^k=true}PP_1\right]. \quad (4.62)$$

Ґрунтуючись на запропонованій формі запису (4.51), наведемо математичний опис основних параметрів необхідних і достатніх для реалізації порожньої (без елементів $CD_x^n \in CF_n$ і подій ME_n) $Form_n$ об'єктно-орієнтованого середовища розробки Rad Studio XE3.

$$Form_1PE[(MP_1)\xrightarrow{mp_3^1=alNone;\ mp_4^1=false;\ mp_5^1=false;\ mp_6^1=255;\ mp_8^1=false;}\wedge$$
$$\wedge\xrightarrow{mp_9^1=false;\ mp_{10}^1=bdLeftToRight;mp_{12}^1=bsSizeable;}\wedge$$
$$\wedge\xrightarrow{mp_{13}^1=0;\ mp_{14}^1=NameFormCaption;mp_{15}^1=212;\ mp_{16}^1=418;\ mp_{17}^1=clBtnFace;}\wedge$$
$$\wedge\xrightarrow{mp_{19}^1=true;mp_{20}^1=crDefault;mp_{22}^1=dmActiveForm;mp_{23}^1=false;}\wedge$$
$$\wedge\xrightarrow{mp_{24}^1=false;mp_{25}^1=dkDra;\ mp_{26}^1=dmManual;mp_{27}^1=true;mp_{29}^1=fsNormal;}\wedge$$
$$\wedge\xrightarrow{mp_{30}^1=TGlassFrame;mp_{31}^1=250;\ mp_{32}^1=0;mp_{35}^1=htContext;...;\ mp_{44}^1=Form_1;}\wedge$$
$$\wedge\xrightarrow{mp_{46}^1=false;mp_{48}^1=true;mp_{49}^1=true;\ mp_{50}^1=false;mp_{51}^1=96;\ mp_{53}^1=pmNone;}\wedge$$
$$\wedge\xrightarrow{mp_{55}^1=poDefaultPosOnly;mp_{56}^1=po\,\mathrm{Pr}oportional;mp_{57}^1=true;\ mp_{58}^1=false;}\wedge$$
$$\wedge\xrightarrow{mp_{59}^1=false;\ mp_{60}^1=10;mp_{62}^1=0;\ mp_{63}^1=tipDontCare;mp_{64}^1=0;...;\ mp_{66}^1=false;}\wedge$$
$$\wedge\xrightarrow{mp_{67}^1=clBlac;mp_{68}^1=false;\ mp_{70}^1=false;\ mp_{71}^1=434;mp_{73}^1=wsNormal}(PP_1)]$$

Можна зауважити, що параметри:

$$mp_4^1, mp_5^1, mp_8^1, mp_9^1, mp_{23}^1, mp_{24}^1, mp_{46}^1, mp_{50}^1, mp_{58}^1, mp_{59}^1, mp_{66}^1, mp_{68}^1, mp_{70}^1,$$
$$mp_{13}^1, mp_{32}^1, mp_{62}^1, mp_{64}^1, mp_{27}^1, mp_{48}^1, mp_{49}^1, mp_{57}^1,$$

мають однакові значеннями, отже, в рамках розроблюваного методу, пропонується згрупувати параметри за однаковими значеннями. Це дає можливість спростить математичне представлення опису до такого виду:

$$Form_1 \in [(MP_1 \in (mp_1^1,...mp_x^1) \xrightarrow{\zeta} PP_1 \in (pp_1^1,...,pp_a^1)) \in Form_1PE]. \quad (4.63)$$

$$\begin{aligned}
&Form_1PE[(MP_1) \xrightarrow{mp_3^1 = alNone;\ mp_4^1, mp_5^1, mp_8^1, mp_9^1, mp_{23}^1, mp_{24}^1, mp_{46}^1, mp_{50}^1, mp_{58}^1,} \wedge \\
&\wedge \xrightarrow{mp_{59}^1, mp_{66}^1, mp_{68}^1, mp_{70}^1 = false; mp_6^1 = 255; mp_{10}^1 = bdLeftToRight; mp_{12}^1 = bsSizeable;} \wedge \\
&\wedge \xrightarrow{mp_{13}^1, mp_{32}^1, mp_{62}^1, mp_{64}^1 = 0;\ mp_{14}^1 = NameFormCaption; mp_{15}^1 = 212;\ mp_{16}^1 = 418;} \wedge \\
&\wedge \xrightarrow{mp_{17}^1 = clBtnFace; mp_{19}^1, mp_{27}^1, mp_{48}^1, mp_{49}^1,\ mp_{57}^1 = true; mp_{20}^1 = crDefault;} \wedge \\
&\wedge \xrightarrow{mp_{22}^1 = dmActiveForm; mp_{25}^1 = dkDra;\ mp_{26}^1 = dmManual; mp_{29}^1 = fsNormal;} \wedge \\
&\wedge \xrightarrow{mp_{30}^1 = TGlassFrame; mp_{31}^1 = 250; mp_{35}^1 = htContext;...;\ mp_{44}^1 = Form_1; mp_{51}^1 = 96;} \wedge \\
&\wedge \xrightarrow{mp_{53}^1 = pmNone;\ mp_{55}^1 = poDefaultPosOnly;\ mp_{56}^1 = popopropoonal; mp_{60}^1 = 10;} \wedge \\
&\wedge \xrightarrow{mp_{63}^1 = tipDontCare;...;\ mp_{67}^1 = clBlac;\ mp_{71}^1 = 434; mp_{73}^1 = wsNormal} (PP_1)]
\end{aligned}$$

Розглянемо математичний опис подій $(me_1^n,\ldots,me_h^n) \in ME_n$ і «лінгвістичні змінні» $(ea_1^n,\ldots,ea_z^n) \in EA_n$, а також «контейнер рішень» $(z_z^o, z_2^o,\ldots,z_q^o) \in Z_o$, притаманні кожній $Form_i$ і приналежні множини $Form_1PE$.

$$\begin{cases}
me_1^n \xrightarrow{\zeta_1 = ea_1^n \text{''лінгвістична змінна 1''} = \varphi_1} z_1^o \\
me_2^n \xrightarrow{\zeta_2 = ea_2^n \text{"лінгвістична змінна 2''} = \varphi_2} z_2^o \\
me_3^n \xrightarrow{\zeta_3 = ea_3^n \text{''лінгвістична змінна 3''} = \varphi_3} z_3^o \\
\ldots \\
me_h^n \xrightarrow{\zeta_n = ea_z^n \text{"лінгвістична змінна n''} = \varphi_n} z_q^o
\end{cases}$$

$$ME_n \longrightarrow Z_o.$$

Врахуємо таку можливість, як використання однієї «лінгвістичної змінної» ea_z^n для опису двох і більше подій $(me_1^n,\ldots,me_h^n)\in ME_n$. Для $Form_n$ кожної ea_z^n «лінгвістичної змінної» належить свій єдиний $z_q^o\in Z_o$ «контейнер рішень», який містить програмний код.

$$Form_1PE[(ME_1)\xrightarrow{me_1^1,me_4^1="лінгвістична\ змінна\ 1",me_2^1="лінгвістична\ змінна\ 2"}(Z_o)] \quad (4.64)$$

Для опису параметрів PC_y^x і їх значень PP_t^x графічних елементів CD_x^n, які є невід'ємною частиною $Form_n$ пропонується використовувати тип запису (4.51), а опис взаємодії подій CE_z^x через множину «лінгвістичних імен» EA_n з «контейнером рішень» Z_o, відповідно до (4.63) .

На базі розробленого математичного опису $Form_1$, представленого вище, наведені приклади опису параметрів і подій $Form_1PE$ та елементів графічного керування даними CD_x^z, у вигляді Button1 (CD_1^1), за подією ce_6^1, яка дозволяє виконати процедуру закриття $Form_1$ через «лінгвістичну змінну 4», що посилається на «контейнер рішень 1», як представлено у виразі 4.65.

Фрагмент графічної моделі (таблиця 4.1, 4.2) взаємодії $Form_1^{master}$ і $Form_{2-9}^{slave}$ з використанням формального опису значень параметрів і подій представлений на рис. 4.9.

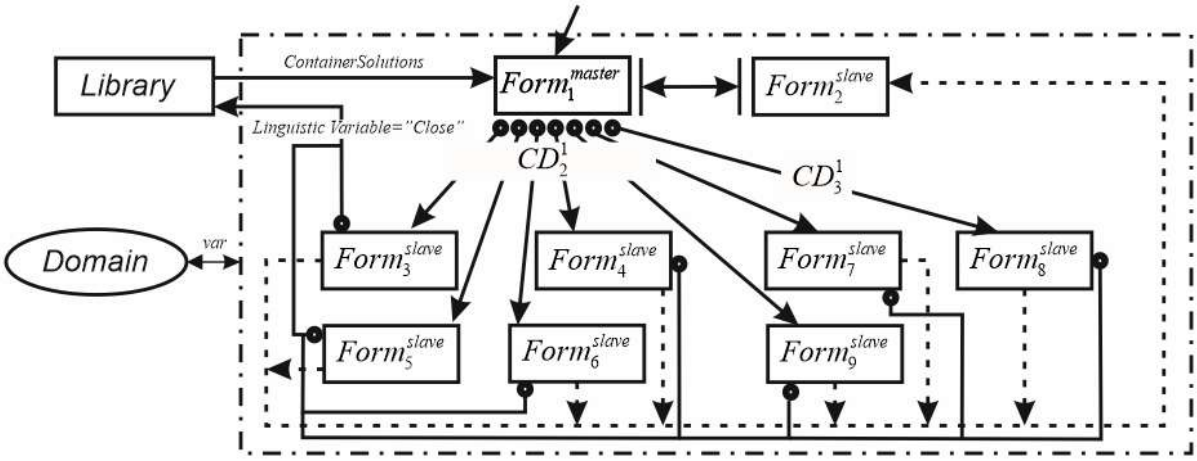


Рисунок 4.9 – Фрагмент графічної моделі взаємодії $Form_1^{master}$ і $Form_{2-9}^{slave}$.

$$
\begin{aligned}
&Form_1(Form_1PE[(MP_1)\xrightarrow{mp^1_4, mp^1_5, mp^1_8, mp^1_9, mp^1_{23}, mp^1_{24}, mp^1_{46}, mp^1_{50}, mp^1_{58}, mp^1_{59}, mp^1_{66}, mp^1_{68}, mp^1_{70} = false;}\wedge\\
&\wedge\xrightarrow{mp^1_3 = alNone;\ mp^1_6 = 255; mp^1_{10} = bdLeftToRight; mp^1_{12} = bsSizeable; mp^1_{13}, mp^1_{32}, mp^1_{62}, mp^1_{64} = 0;}\wedge\\
&\wedge\xrightarrow{mp^1_{14} = NameFormCaption; mp^1_{15} = 212; mp^1_{16} = 418; mp^1_{17} = clBtnFace; mp^1_{19}, mp^1_{27}, mp^1_{48}, mp^1_{49},\ mp^1_{57} = true;}\wedge\\
&\wedge\xrightarrow{mp^1_{20} = crDefault; mp^1_{22} = dmActiveForm; mp^1_{25} = dkDra;\ mp^1_{26} = dmManual; ontext;\ mp^1_{44} = Form_1;}\wedge\\
&\wedge\xrightarrow{mp^1_{29} = fsNormal; mp^1_{31} = 250; mp^1_{30} = TGlassFrame; mp^1_{35} = htCmp^1_{35} = htContext;\ mp^1_{44} = Form_1;}\wedge\\
&\wedge\xrightarrow{mp^1_{51} = 96; mp^1_{53} = pmNone;\ mp^1_{55} = poDefaultPosOnly; mp^1_{60} = 10;\ mp^1_{63} = tipDontCare; mp^1_{56} = po\,\mathrm{Pr}\,oportional;}\wedge\\
&\wedge\xrightarrow{mp^1_{67} = clBlac; mp^1_{71} = 434; mp^1_{73} = wsNormal}(PP_1)]\wedge[(ME_1)\xrightarrow{\mathrm{me}^1_1, \mathrm{me}^1_4 = "лінгвістична\ змінна\ 1";}\wedge\\
&\wedge\xrightarrow{me^1_2 == "лингвистическая\ переменная 2"; me^1_3 == "лінгвістична\ змінна\ 3"}(Z_0))])\wedge\\
&\wedge(CF_1[CD^1_1(PC^1_1)\xrightarrow{pc^1_2, pc^1_6, pc^1_{12}, pc^1_{14}, pc^1_{19} = alNone; pc^1_3, pc^1_{43}, pc^1_{53} = false; pc^1_7 = Close; pc^1_{10} = crDefault;}\wedge\\
&\wedge\xrightarrow{pc^1_{13}, pc^1_{27}, pc^1_{41}, pc^1_{42}, pc^1_{46} = -1; pc^1_{20}, pc^1_{35}, pc^1_{36}, pc^1_{37}, pc^1_{38}, pc^1_{39}, pc^1_{48}, pc^1_{51} = true; pc^1_{15} = crDrag; pc^1_{16} = dkDrag;}\wedge\\
&\wedge\xrightarrow{pc^1_{17} = dmManual; pc^1_{22} = 25; pc^1_{23}, pc^1_{47}, pc^1_{49} = 0; pc^1_{25} = htContext; pc^1_{26} = CloseForm1; pc^1_{28} = iaLeft; pc^1_{27} = 544;}\wedge\\
&\wedge\xrightarrow{pc^1_{33} = mrNone; pc^1_{34} = BoottonClose; pc^1_{44} = bsPushButton; pc^1_{50} = 328; pc^1_{52} = 75}(PP^1_1)\wedge\\
&\wedge(CE^1_1)\xrightarrow{ce^1_6 = "лінгвістична\ змінна\ \ 4"}(Z_o)])\wedge[CD^1_2(PC^1_2)\xrightarrow{pc^2_2.pc^2_6, pc^2_{12}, pc^2_{14}, pc^2_{19} = alNone; pc^2_3,}\wedge\\
&\wedge\xrightarrow{pc^2_{43}, pc^2_{53} = false; pc^2_7 = Close; pc^2_{13}, pc^2_{41}, pc^2_{46} = -1; pc^2_{10} = crDefault; pc^2_{15} = crDrag; pc^2_{16} = dkDrag; pc^2_{49} = 0;}\wedge\\
&\wedge\xrightarrow{pc^2_{34} = BoottonOpen; pc^2_{44} = bsPushButton; pc^2_{50} = 328; pc^2_{52} = 75}(PP^1_2)\wedge\\
&\wedge(CE^1_2)\xrightarrow{ce^2_6 = "лінгвістична\ змінна\ 5"}(Z_o)])
\end{aligned}
\tag{4.65}
$$

Висновки до розділу 4

1. Аналіз показав, що існуючі стандартні моделі ЖЦ і методології розробки ПП не дозволяють вирішити поставлену задачу, тому, в рамках даних досліджень, розроблено комплекс моделей і методів, який складається з:

– удосконаленої моделі ЖЦ «Jump» розробки адитивного кібер-дизайну (рис. 4.1);

– моделі формалізації ЖЦ «Jump» у вигляді кортежу параметрів, як сукупності правил до структури даних, вимог обмеження цілісності і мови подання маніпуляцій з даними (4.1), а також опис основних базових понять і види співвідношень (4.2)–(4.30).

– структурного представлення адитивного кібер-дизайну, як інтерфейсу користувача (HMI), на базі графічних елементів (GUI) кібернетичної складової CPPS (рис. 4.3);

– математичного опису розробки HMI на базі GUI елементів для об'єктно-орієнтованих мов високого рівня програмування (4.31)–(4.54).

2. Особливу увагу приділену вирішенню завдань автоматизації процесу розробки адитивного кібер-дизайну, шляхом реалізації функціональних вимог до CPPS з прив'язкою до подій GUI елементів, що дозволяє використовувати «Lingustic Variable», який посилається на певний «Container Solution» та містить програмний код для реалізації необхідних функцій.

3. Отримані результати дозволять спростити процес розробки кібернетичної складової CPPS, забезпечити автоматизацію даного процесу з метою скорочення часу реалізації програмних рішень за рахунок використання «Lingustic Variable» і «Container Solution».

Варто зауважити, що отримані результати можуть бути використані при вирішенні задач удосконалення існуючих кібернетичних складових CPPS, а також розробки нових, при цьому рівень автоматизації розробки буде напряму залежати від кількості фрагментів «Container Solution», збережених

в базі знань (БЗ), фрагменти реалізацій «Container Solution» представлений у додатку Г.

Список джерел, які використано у даному розділі, наведено у повному списку використаних джерел [200–215].

5 РОЗРОБКА СИНТАКСИЧНОЇ І СЕМАНТИЧНОЇ МОДЕЛІ МОВИ ВИЗНАЧЕННЯ І ОПИСУ МОДЕЛЮВАННЯ КІБЕРНЕТИЧНОЇ СКЛАДОВОЇ

Для вирішення поставленого завдання, в даній дисертаційній роботі, ґрунтуючись на запропонованій архітектурно-логічній моделі керування процесами в складних організаційно-технічних виробничих об'єктах і методах її декомпозиції, розроблена технологія процесу керування. На її базі запропоновані системні моделі опису, що дозволяють розробнику отримати алгоритми функціонування організаційно-технічних виробничих об'єктів, як в цілому так і за кожним етапом (2-3 розділи).

Запропоновано і формалізовано модель ЖЦ «Jump», на базі синтезу візуальних компонентів і формального подання властивостей і подій, що дозволяє автоматизувати розробку адитивного кібер-дизайну, на базі використання «контейнерів рішень» і «лінгвістичних змінних», представлених в 4 розділі дисертаційної роботи.

Варто зауважити, що запропоновані рішення складні для розуміння користувачів і є математичним ядром, для роботи з яким необхідно розробити формальну мову опису структури і параметрів розроблюваного адитивного кібер-дизайну, який буде максимально близьким і інтуїтивно зрозумілим деякій підмножині природної мови і застосовним в даній предметній області (ПрО).

З одного боку було виявлено, що розробники адитивного кібер-дизайну CPPS використовують прив'язку до елементів інтерфейсу користувача і мінімізують використання програмного коду. Даний код, в більшості випадків, є оброблювачем тієї чи іншої події, ініціалізованої користувачем або внутрішніми процесами, шляхом взаємодії з необхідними візуальними елементами для вирішення того чи іншого завдання.

З іншого боку – користувачі, які є фахівцями ПрО, можуть сформулювати семантично правильну структуру і внутрішню ієрархію всіх елементів інтерфейсу з необхідними значеннями параметрів і подій, які необхідно реалізувати для вирішення поставлених завдань в ТЗ.

Виникає необхідність розробити синтаксичну та семантичну моделі нової формальної мови, яка об'єднувала б в собі запропоновані в 2-4 розділі моделі і методи керування процесами в складних організаційно-технічних виробничих об'єктах, декомпозиції і формалізації CPPS в конструкцію побудови зрозумілого і простого (для фахівців і розробника) адитивного кібер-дизайну, близького деякій підмножині природної мови. Це буде сприяти вирішенню завдань автоматичної трансформації семантичних правил опису процесу розробки CPPS в синтаксично і термінологічно правильну структуру, для реалізації її компіляції в необхідному середовищі розробки.

5.1 Розробка синтаксичної та семантичної мовної моделі

Визначемо поняття мовної моделі (ММ) – це декларативна (не процедурна) мова, призначенням якої є визначення та опис термінології, в основі якої покладено запропоновані поняття моделі ЖЦ «Jump» і співвідношення між метаданими та даними предметної області, а також способів їх перетворення.

У даній дисертаційній роботі запропонована наступна специфікація мови моделей даних:

– **дозволені буквено-цифрові символи**, які підтримуються середовищами розробок мов високого рівня програмування і відповідають таблиці ASCII-кодів: + - \ . , ! “ < > = ( ) $ % & ~ * _ & @ пробіл; { };

– **ключові слова**: базове поняття у вигляді слів, зарезервованих в розроблюваній математичній моделі (ММ), які служать для опису ключових ознак (табл. 5.1);

Таблиця 5.1 – Ключові слова

Form	*ValueElement*
$Form^{master}$	*LingusticVariable*
$Form^{slave}$	*ContainerSolution*
ParameterForm	*parameter*
ElementForm	*value*
EvenForm	*name*
ParametrElement	*event*
EventElement	*cod*

– **ідентифікатори**, використовувані для позначення таких ознак:

- ознака приналежності параметрів та подій до доменного, або не доменного типів: *domen*, *not_domen*. Домени відповідних характеристик (значень), що належать до перелічуваного (облікового) типу, який має можливість вибору із заздалегідь сформованого списку. Прикладом можуть виступати деякі параметри *ParameterForm* відображення *Form*, які можуть набувати значень *true* чи *false*. Наприклад, параметр *Align*, що притаманний $dom(parameter^{z}_{p\_список})$ та $dom(parameter^{h}_{p\_список})$, з (4.27), може набувати значень, фіксованих середовищем розробки CPPS і зазначених в ТЗ;

- тип даних значень (*value*), який визначає характеристику параметрів *ParameterForm* та *ParametrElement* (текстове, булеве, цілочисельне, цілочисельне негативне, текстове, словосполучення), опис яких представлено в таблиці 4.3;

- лінгвістичний опис ознаки посилання *LingusticVariable* (*name*) на *ContainerSolution*, який містить необхідний *cod*;

- базові поняття моделі ЖЦ «Jump», які дають можливість зв'язати події *ElementForm* і *EventElement*, містять набір певних *event*, що належать певному візуальному графічного елементу з *ContainerSolution* (*cod*) через *LingusticVariable* (*name*).

Як можна побачити, на відміну від ключових слів, запропоновані ідентифікатори теоретично можна перевизначити, але це призводить до виникнення помилок, тому перераховані вище ідентифікатори входять в фіксований словник ключових слів.

– **літерали,** певний набір значень, які не представлені ідентифікатором.

Рядкові літерали представляються у вигляді послідовності дозволених символів з різним типом написання (великі та малі) літер. Наприклад, *name_form*, яка використовується в параметрах *Caption*, *Name* і.т.д., а також привласнення унікального імені (*name*) для кожного *LingusticVariable*, яке містить певний фрагмент програмного коду. Приклад «зберегти в БД», «розрахувати результат», і т.д., які задаються кінцевим користувачем з метою зручності методології, що розробляється.

Алгебраїчні літери – літери, що являють собою опис простих логічних операцій типу *True, False*, які дозволяють задати значення (*value*) того чи іншого параметра (*parameter*), який належить *ParameterElement*, *ParameterForm* та є необхідним і достатнім для опису властивостей візуальних елементів CPPS, або функції c-MES, відповідно до вимог ТЗ.

Зарезервовані літери являють слово, словосполучення або скорочення, яке дає можливість вибрати ту чи іншу властивість параметра, необхідну для досягнення умов заданих в алгоритмі функціонування. Прикладом може виступати властивість форми *WindowsState*, в середовищі розробки RadStudioXE16, якому можна обрати зарезервовані слова скорочень: *wsNormal,wsMinimized, wsMaximized*. Тобто, при першому запуску, розробник може задати вид відображення форми. *wsNormal* – відображення за замовчуванням, в тому вигляді, в якому вона створювалася на етапі проектування, *wsMinimized* – форма, що відображується в мінімізованому вигляді на панелі завдань, *wsMaximized* – при запуску форма розгортається на весь розмір робочого столу.

Зарезервовані літери можуть бути загальними для *ParameterForm* і *ParametrElement*, а також спеціалізованими, тобто належати певній

візуальній формі, яка визначає специфіку того чи іншого елемента. Але, слід зауважити, що зарезервовані літери для визначення значень того чи іншого параметра візуальних компонентів, які мають однакове призначення, можуть виконувати різні задані функції і обробляти події в одному середовищі розробки [216].

Типи представлених значень, в яких містяться деякі параметри *ParameterForm* і *ParametrElement*, допустимі в області застосування (таблиця 4.3).

Цілочисельний тип даних (integer) – дозволяє привласнити параметру *ParameterForm* і/або *ParametrElement* певне і необхідне цифрове значення розмірності або координат розміщення візуального елемента щодо *Form*. Використовується, в основному, для опису візуальних графічних елементів. Він є найменшим логічним елементом двовимірного цифрового зображення в растровій графіці (pixel). Довжина рядка залежить від роздільної здатності екрану і вимог ТЗ, висунутого замовником розробнику. Формальне представлення надано в (4.55).

Цілочисельний негативний – дозволяє привласнити параметру певне значення, що входить в діапазон $(-1,1,2,3,...,n)$, яке належить виключно *ParametrElement* і описує нумерацію в даному контексті:

-1 – нумерація відсутня, параметр не задіяний;

$1,2,3,...,n$ – нумерація графічного зображення (іконки), яка належить певному параметру (*parameter*) для *ElementForm*. Формальне представлення приведено в (4.58).

Текстові / лінгвістичне (char) – дозволяє привласнити параметру логічне впорядкування значень символів, які містять необхідні для користувача пояснення або назву графічних елементів, потрібних для зручності роботи з CPPS. Також даний тип подання значення використовується для завдання певного імені *LingusticVariable*, що присвоюється події *EvenForm*, *EventElement*. Формальне представлення надано в (4.59).

Логічний (булевий) – може приймати тільки два значення: *true* (правда) або *false* (брехня) і виконує роль перемикача використання того чи іншого параметра в *ParameterForm* і *ParametrElement*. Формальне представлення приведено в (4.56).

Текстове словосполучення (перелічуваний тип) – тип даних, заданий списком у вигляді домену (4.27)–(4.30), що дозволяє задати список зарезервованих слів в середовищі розробки або скорочень, які можуть приймати той чи інший *parameter* для *ParameterForm* і *ParametrElement*. Формальне представлення надано в (4.57).

Розділювачі – символьні позначення виділення основних елементів синтаксичної конструкції розроблюваної ММ.

<*Form*> (кутові дужки *Form*) – використовуються для визначення ключового слова, яке показує початок метаопису тієї чи іншої *Form* в конструкції ММ.

</*Form*> (слеш кутові дужки *Form*) – використовується для визначення ключового слова, яке показує завершення метаопису тієї чи іншої *Form* в конструкції ММ.

Для запропонованої конструкції ключового слова, на початку і завершенні метаопису *Form* накладені такі обмеження: назва *Form* може мати нумерацію як *Form*1, або буквене визначення, наприклад, *Form_master* або *Form_add_operat*. При цьому обов'язково ключове слово початку метаопису має збігатися з ключовим словом завершення метаопису тієї чи іншої *Form* в конструкції ММ. При невиконанні цієї вимоги до конструкції, інтерпретатор ММ не зможе сприйняти вміст, як метаопис всіх необхідних параметрів і подій властивих даній *Form*.

{ (відкриваюча фігурна дужка) – обов'язковий символ початку рядка метаопису *Form* і *ElementForm*.

} (закриваюча фігурна дужка) – обов'язковий символ завершення рядка метаопису *Form* і *ElementForm*.

(решітка) – після цього символу конструкція інтерпретатора ММ сприймає початок опису графічних візуальних елементів інтерфейсу користувача (*ElementForm*).

/# (слеш решітка) – після даної комбінацій символів інтерпретатор ММ вважає, що опис графічних візуальних елементів інтерфейсу користувача (*ElementForm*) завершений.

/ (слеш) – використовується для визначення ієрархій метаопису візуальних графічних елементів (*ElementForm*), відповідно до дерева побудови адитивного кібер-дизайну і застосовується всередині # /# метаопису *Form*. *ElementForm*1/*ElementForm*2 – необхідно розуміти як *ElementForm*2, що знаходиться всередині *ElementForm*1 і є її невід'ємною частиною.

[ ] квадратні дужки – використовується для метаопису необхідних параметрів і подій *ParameterForm*, *EvenForm*, *ParametrElement*, *EventElement*.

; (крапка з комою) – обов'язковий символ конструкції ММ, який показує, що присвоєння даному *parameter* або *event* відповідних *value* та *name* завершене, застосовується всередині [ ].

, , (перерахування через кому) – використовується для перерахування назв *parameter* для *ParameterForm*, *ParametrElement*, а також *event* для *EvenForm*, *EventElement* за умови, що для набору декількох *parameter* або *event* відповідні значення *value* та *name* однакові, застосовується всередині [ ].

= (знак рівності) – присвоює *parameter* певне значення *value* і застосовується для визначення *event* певного *name* з *LingusticVariable*, яке містить посилання на *cod*, або його фрагмент в *ContainerSolution*. Варто врахувати, що в залежності від контексту (логіки й змісту виконуваних дій), даний знак можна трактувати і як інструкцію присвоєння, згідно з якою для зазначеного базового параметра визначається значення, яке йому належить.

Коментарі – всі символи і рядки, записані всередині даної конструкції інтерпретатором ММ, ігнорується і сприймаються як коментарі. Допустимі до використання буквено-цифрові символи національних алфавітів, підтримуються операційною системою і середовищем розробки. Обмеженням для коментарів служить те, що послідовність не повинна перевищувати 255 символів.

?** (знак питання з двома зірочками) – показує, що після заданих символів буде коментар, який ігнорується інтерпретатором ММ.

**? (дві зірочки і знак питання) – показує, що після заданих символів закінчується коментар і далі йде текст, що не ігнорується інтерпретатором ММ.

Для адаптації розроблюємого синтаксису опису ММ, запропоновано використовувати форму Бекуса-Наура. Обґрунтуванням цього вибору служить те, що розширена форма Бекуса-Наура використовується для опису контекстно-вільних граматик [217] і дозволяє спростити і скоротити обсяг опису. Розширена форма Бекуса-Наура описана в міжнародному стандарті [218]. Аналіз якого показав, що дана форма дає можливість розробити інтуїтивно просту і адаптивну формальну мову подання та опису необхідних даних для розробки CPPS, на базі підходів до об'єктно-орієнтованого програмування.

Ґрунтуючись на запропонованій вище специфікації, мови моделей даних і основній концепції моделі ЖЦ «Jump», пропонується наступна синтаксична діаграма ММ адитивного кібер-дизайну (рис. 5.1).

Синтаксична діаграма, запропонованих в даному дослідженні типів представлення значень, які можуть належати ідентифікаторам, представлена на рисунку 5.2 [219].

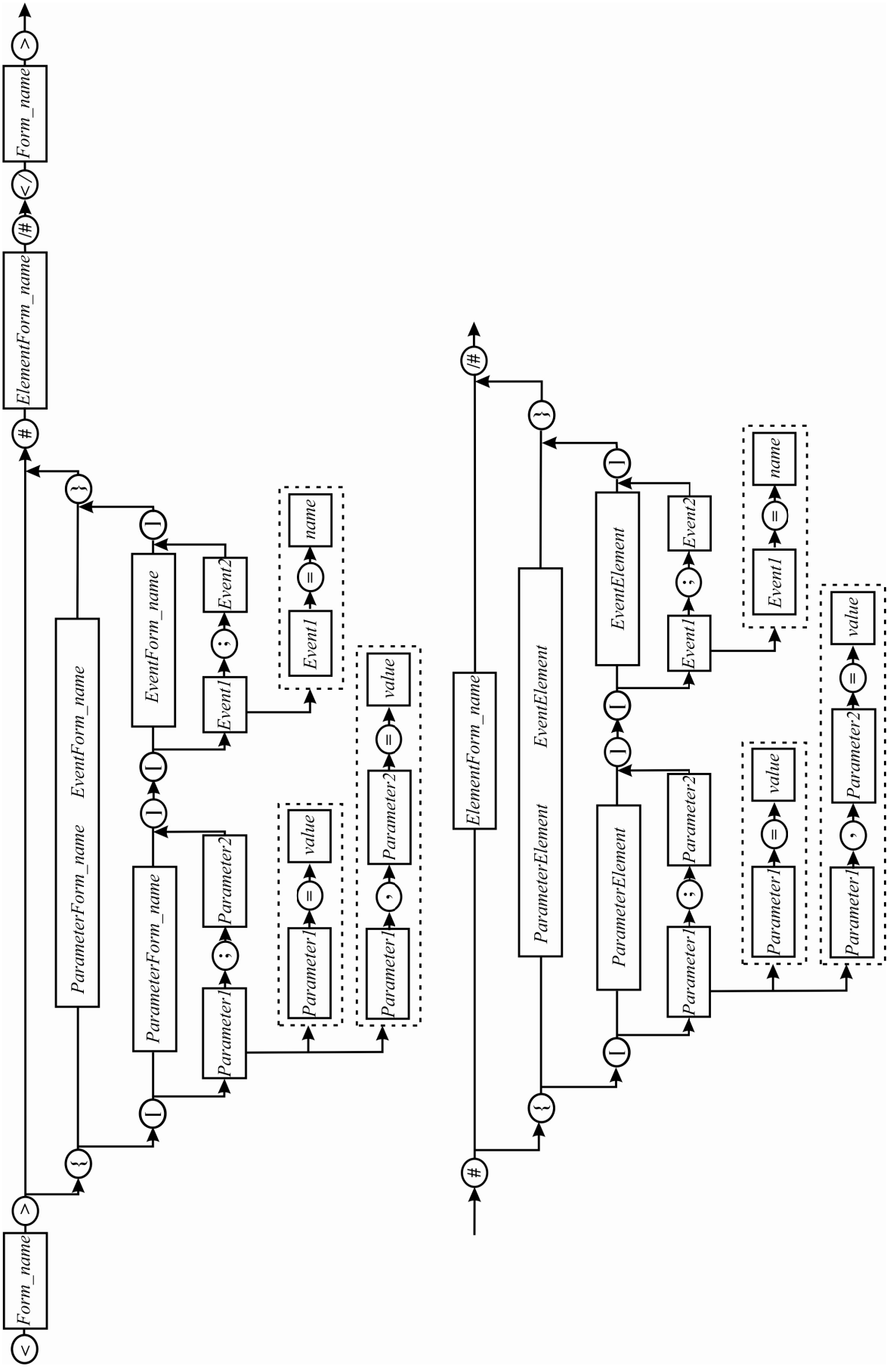


Рисунок 5.1 – Синтаксична діаграма ММ, що розробляється

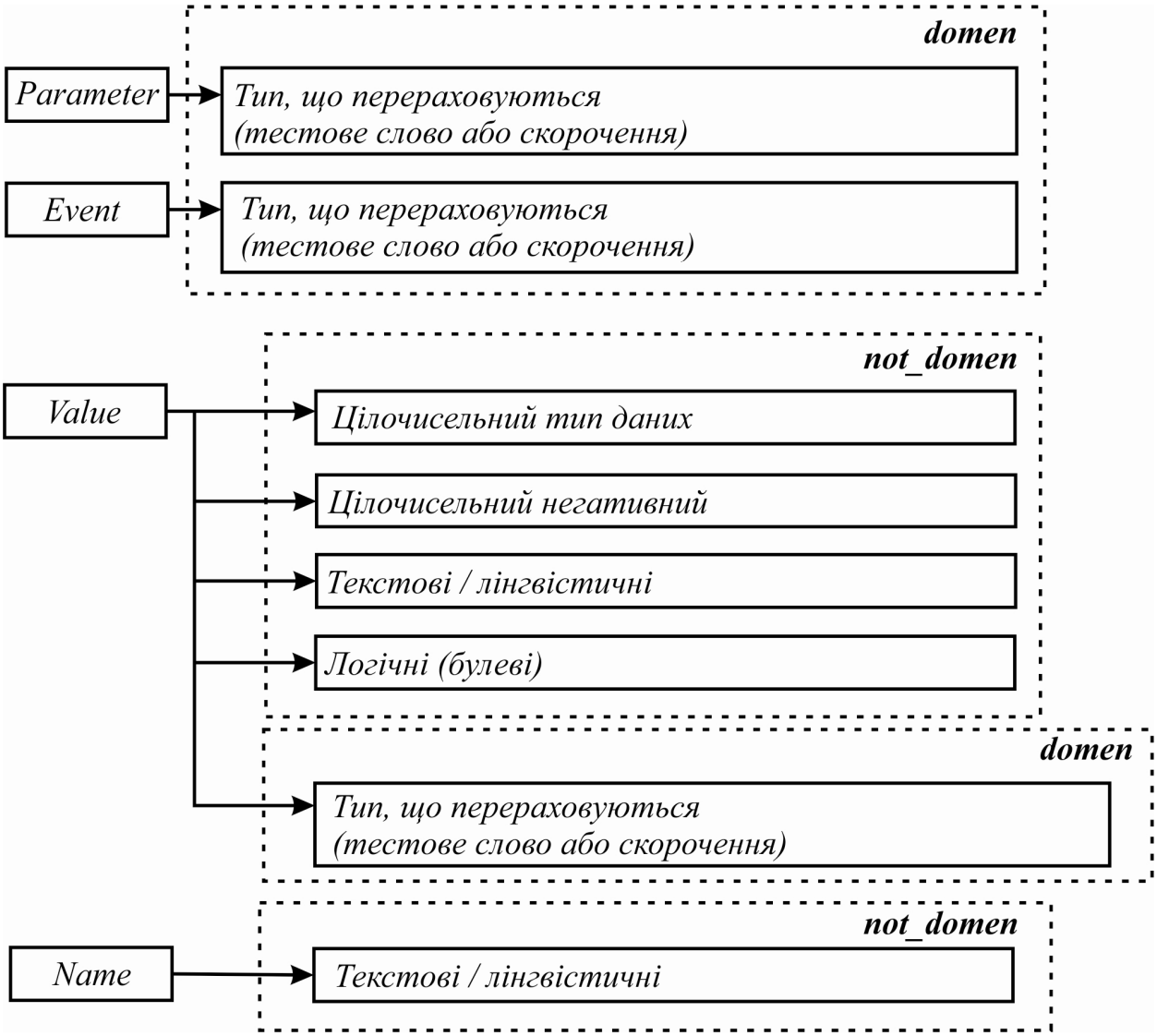


Рисунок 5.2 – Синтаксична діаграма типів представлення значень ідентифікаторів

Як можна бачити з рисунка 5.2 ідентифікатор *Parameter*, *Event* відноситься до *domen*-ого (списочного) типу і представляється у вигляді текстового слова або скорочення, що відносяться до *ParameterForm_name* та *EvenForm _ name*, а також *ParameterElement* и *EvenElement*, відповідно до рисунка 5.1.

Список параметрів (*parameter*) і подій (e*vent*), які належать *ParameterForm_name*, *EvenForm _ name*, в рамках одного середовища розробки, є постійним і незмінним. Для списку параметрів (*parameter*) і подій (e*vent*), які належать *ParametrElement*, *EventElement* відповідно,

однаковим є обмеження, що ці візуальні графічні елементи мають однакові призначення в рамках одного середовища розробки. Варто звернути увагу, що до даного типу відноситься *value* для *ParametrElement* і *ParameterForm_name*, які містять зарезервоване середовищем розробки текстове слово або скорочення.

Ідентифікатори *value* і *name* відносяться до *not_domen*-ому (не облікового) типу. Це обґрунтовано тим, що значення *value* можуть задаватися розробником залежно від вимог, що висуваються ТЗ на адитивний кібер-дизайн. Для ідентифікатора *name*, який входить *LingusticVariable*, ім'я, яке посилається на *ContainerSolution*, містить необхідний фрагмент або частину програмного коду (*cod*). Воно задається користувачем, з урахуванням його логічних переваг і зручністю застосування.

Для зручності читання та подання розробленої декларативної мови (рис. 5.1–5.2) необхідно щоб він мала якості розуміння і читання. Цього можна досягти, використовуючи мінімум три принципи подання мови [220], а саме:

– бути максимально лінійною;

– бути короткою;

– бути само-документованою.

Ґрунтуючись на запропонованих допущеннях і рекомендаціях, для розроблюваної декларативної мови CPPS пропонується наступний тип стилю запису ММ, який дає можливість спростити та стандартизувати код.

Приклад 5.1

< Form _ master >

{ ?** відкриття блоку опису параметрів і значень, а також подій та імен *LingusticVariable* для *Form _ master* **?

[$parameter1 = value; parameter2, parameter3 = value$]

[$event1 = name; event2 = name$]

} ?** закриття блоку опису параметрів і значень, а також подій та імен *LinguisticVariable* для *Form _ master* **?

“ ім'я елемента в середовищі розробки ”

?** відкриття блоку опису візуальних графічних елементів *Form_master* **?

{ ?** блок опису *Element*1*_Form_master* **?

[$parameter1 = value; parameter2, parameter3 = value$]

[$event1 = name; event2 = name$]

} ?** закриття блоку опису *Element*1*_Form_master* **?

{ ?** блок опису *Element*2*_Form_master* **?

[$parameter1 = value; parameter2, parameter3 = value$]

[$event1 = name; event2 = name$]

} ?** закриття блоку опису *Element*2*_Form_master* **?

/# ?** закриття блоку опису візуальних графічних елементів *Form_master* **?

$< / Form\_master >$

При необхідності реалізації ієрархії (дерева побудови) приналежності візуальних графічних елементів *ElementForm*1/*ElementForm*2 пропонується наступний фрагмент структури метаопису:

Приклад 5.2.

“ ім'я елемента в середовищі розробки ” ?** відкриття блоку опису візуальних графічних елементів *Form_master* **?

{ ?** блок опису *Element*1*Form _ master* **?

[$parameter1 = value; parameter2, parameter3 = value$]

[$event1 = name; event2 = name$]

} ?** закриття блоку опису *Element*1*Form _ master* **?

/ “ ім'я елемента в середовищі розробки ”

{ ?** блок опису $Element2Form\_master$ **?

[$parameter1 = value; parameter2, parameter3 = value$]

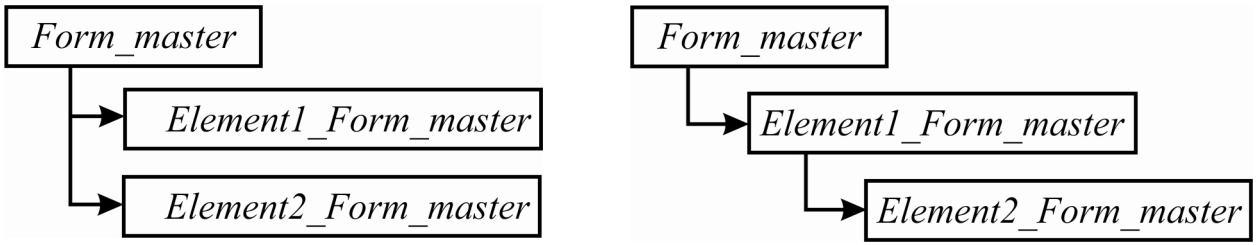
[$event1 = name; event2 = name$]

} ?** закриття блоку опису $Element2Form\_master$ **?

/# ?** закриття блоку опису візуальних графічних елементів $Form\_master$ **?

Використання “/” (слеш) дозволить інтерпретатору ММ визначити ступінь вкладенності (приналежності) візуального елемента в інший, тобто можливість реалізувати дерево структури (Structure) адитивного кібер-дизайну в середовищі розробки. На рисунку 5.3 наведено графічне представлення структури дерева побудови $Form\_master$ CPPS (приклад 1 (а) і приклад 2 (б)).

Для визначення відповідних $value$ і $name$, в наведених вище прикладах, в $LinguisticVariable$ для $parameter$ та $event$ відповідно, після $(=)$ задається тип значення. При відсутності значень, або використанні значень, зарезервованих середовищем розробки за замовчуванням, даний параметр в метаописі не декларується (не позначається).



а) б)

а) $Element1Form\_master$ і $Element2Form\_master$ однозначно належить $Form\_master$;

б) $Element2Form\_master$ належить $Element1Form\_master$;

Рисунок 5.3 – Графічне представлення дерева побудови структури CPPS

На прикладі 5.3 показано метаопис створення порожньої форми в середовищі RadStudio XE6 для VLC Form Application.

Приклад 5.3

$$
\begin{aligned}
&<Form1> \\
&\quad \{ \\
&\quad\quad [Caption = example\ 1, ClientHeight = 464, ClientWidth = 687, \\
&\quad\quad Height = 503, Name = Form\_master, Width = 703] \\
&\quad \} \\
&</Form1>
\end{aligned}
\tag{5.1}
$$

На базі наведеного в (5.1) метаопису було згенеровано графічне представлення найпростішої користувальницької форми (рис. 5.4).

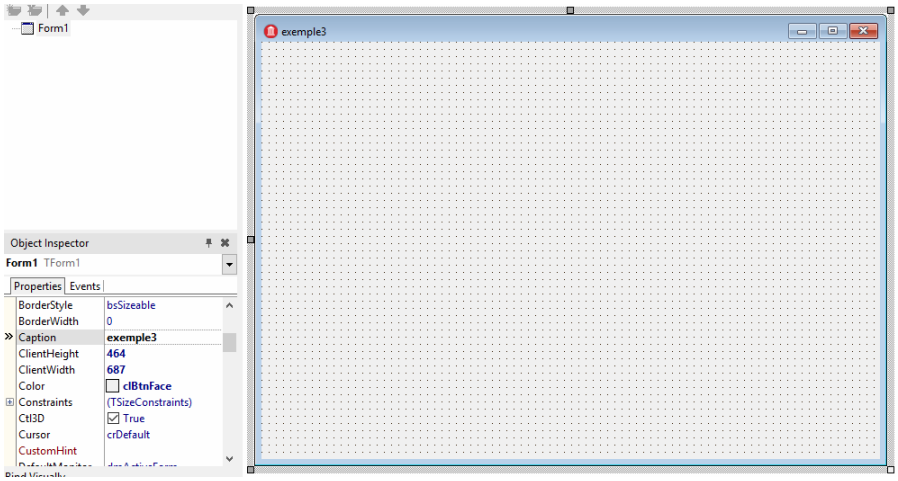


Рисунок 5.4 – Фрагмент середовища розробки RadStudio XE6 найпростішої користувальницької форми

Фрагмент метаопису додаткових візуальних графічних елементів виду *Standard- Button* (користувальницької кнопки, яка виконує певну подію) представлено в (5.2).

 # "Buttion_close"

$$[Caption = Close, Height = 33, Top = 408, Left = 560, \\ Name = Buttion\_close, Width = 91] \quad (5.2)$$

/#

На рисунку 5.5 наведено приклад реалізації форми з елементом Button, метаопис якого наведено в (5.1) і (5.2) відповідно.

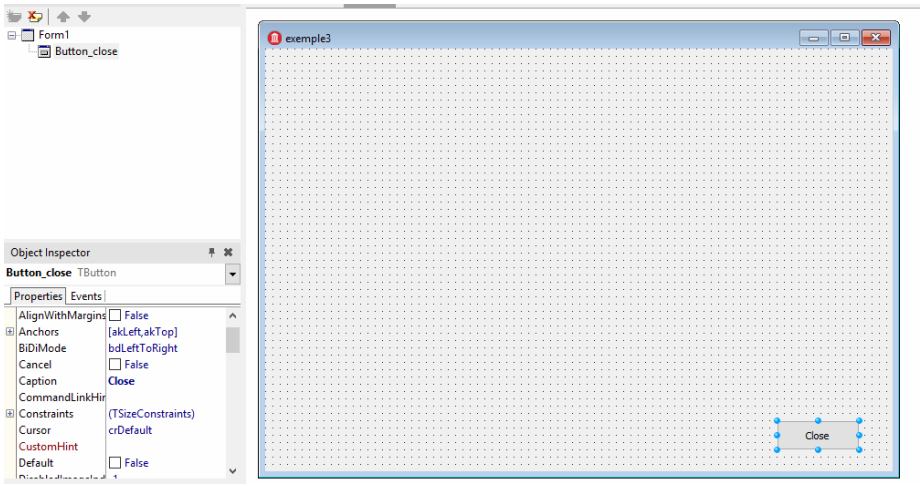


Рисунок 5.5 – Фрагмент середовища розробки з реалізацією форми і графічним елементом Button

Крім реалізації графічного візуального інтерфейсу, наведеного на рис. 5.4–5.5, на базі метаописів (5.1)–(5.2) був згенерований програмний код мовою Pascal, представлений на рис. 5.6.

```
unit Unit1;

interface

uses
  Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,
  Vcl.Controls, Vcl.Forms, Vcl.Dialogs;

type
  TForm1 = class(TForm)
    Button_close: TButton;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

end.
```

Рисунок 5.6 – Програмний код мовою Pascal

Кожен елемент опису ММ, наведений в синтаксичній моделі (5.1)–(5.2), записується відповідно до синтаксичної діаграми (рис. 5.1) і діаграми типів представлення значень ідентифікаторів (рис. 5.2). Семантична модель ММ являє собою систему значень, приписуваних конструкціям і розробленій синтаксичній ММ (інтерпретації конструкції). Ця модель подається в процесі тлумачення (розбору) запропонованих правил опису та подання специфікації ММ, символів і їх комбінацій.

Розглянемо метаопис прикладу (5.2), для визначення необхідності реалізації вкладень (приналежності) одного візуального елемента в інший, як показано на рисунку 5.3,б. Відповідно до запропонованої синтаксичної моделі (рис. 5.1), метаопис прийме наступний вигляд:

$$
\begin{aligned}
&\#\ \textit{“GrupBox1”}\\
&\quad [\textit{Caption} = \textit{GroupBox}1, \textit{Height} = 186, \textit{Top} = 272,\\
&\quad \textit{Left} = 4, \textit{Name} = \textit{GroupBox}1, \textit{Width} = 678]\\
&\quad /\ \textit{“Buttion1”}\\
&\quad [\textit{Caption} = \textit{Close}, \textit{Height} = 32, \textit{Top} = 146, \textit{Left} = 585,\\
&\quad \textit{Name} = \textit{Buttion}1, \textit{Width} = 86]\\
&/\#
\end{aligned}
\qquad (5.3)
$$

У середовищі розробки даний метаопис дозволяє реалізувати ступінь вкладенності візуальних графічних елементів один в одного і побудувати «дерево» $\textit{Form}1$, на базі якого розробляється інтерфейс користувача, відповідно ТЗ на CPPS і алгоритму функціонування. На рисунку 5.7 наведено фрагмент згенерованого інтерфейсу користувача, відповідно до метаопису (5.3) у середовищі розробки RadStudio XE6.

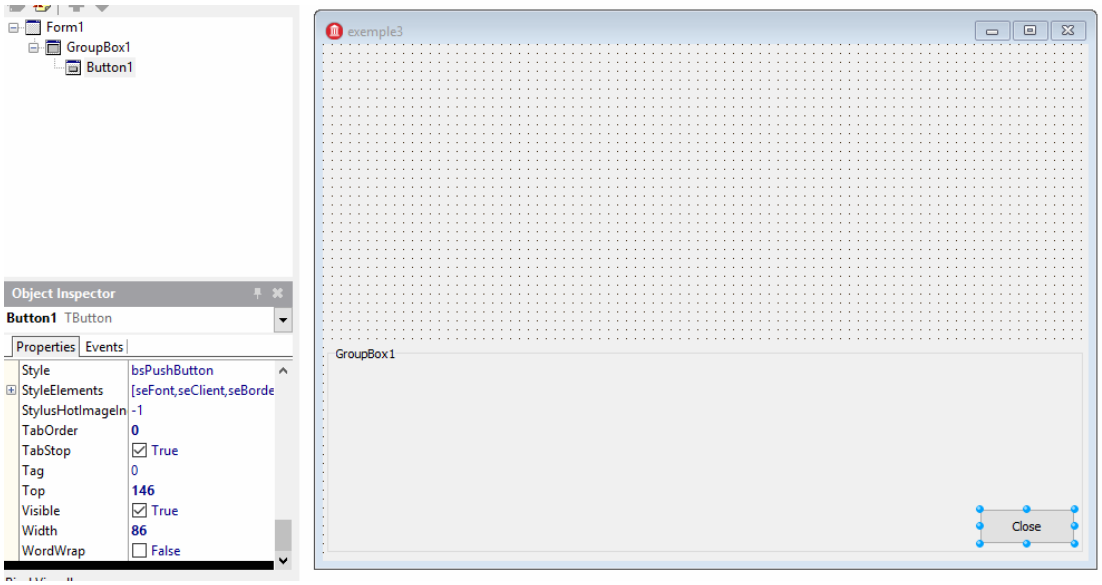


Рисунок 5.7 – Реалізація інтерфейсу *Form*1

З наведених вище фрагментів метаописів (5.1)–(5.3) і рис. 5.3 можна задати будь-яку глибину вкладення графічних елементів інтерфейсу користувача. Це дає можливість реалізувати, за допомогою запропонованої синтаксичної діаграми ММ (рис. 5.1), структуру CPPS будь-якого ступеня складності і тим самим спростити процес розробки візуальної складової на базі об'єктно-орієнтованого підходу до програмування.

5.2 Розробка синтаксису метаопису подій

Ґрунтуючись на запропонованій синтаксичній діаграмі, представленій на рисунку 5.1, пропонується наступний метаопис подій (*event*) для *Form* та *ElementForm*. На базі (5.1) наведемо приклад простого метаопису спрацьовування на елементі *Buttion* події *Onclick*, на відпрацювання *LingusticVariable* з іменем *Close_All*.

Приклад 5.4

< *Form*1 >

{

$$
\begin{aligned}
&[Caption = example\,1, ClientHeight = 464, ClientWidth = 687,\\
&Height = 503, Name = Form\_master, Width = 703]\\
&[Caption = Close, Height = 33, Top = 408, Left = 560,\\
&Name = Buttion\_close, Width = 91, OnClick = Close\_All]\\
&\}\\
&</Form1>
\end{aligned}
\tag{5.4}
$$

Як можна бачити з метаопису на прикладі (5.4), розробник описує існування події на елементі *Buttion* з ім'ям $Name = Buttion\_close$, *LingusticVariable* під ім'ям $Close\_All$ на подію *Onclick*. Дане представлення дозволяє розробнику реалізувати модель ЖЦ процесу управління розробкою CPPS, яке представлене в 4 розділі у вигляді послідовності зв'язків:

$$event \rightarrow LingusticVariable \rightarrow ContainerSolution \rightarrow cod \tag{5.5}$$

За результатами виконання метаопису (5.4) розробник отримує згенерований програмний код, представлений на рисунку 5.8.

```
procedure TMaster.Button_closeClick(Sender: TObject);
begin
Close;
end;
```

Рисунок 5.8 – Результат генерації програмного коду

Розглянемо фрагмент – приклад метаопису на прикладі (5.6) реалізації більш складної конструкції коду, який був згенерований в процесі проектування «Автоматизована система нормування «НОРМА» [221].

На *Form*1 існує елемент *DBGrid_operac* для відображення інформації з бази даних (БД). В *Properties_DBGrid_operace* вказана прив'язка до візуального випадаючого елементу інтерфейсу *PopupMenu_operac*.

Необхідно згенерувати програмний код видалення з БД обраного запису в $DBGrid\_operac$.

Приклад 5.5

$<Form1>$

$\vdots$

$\#$ *"DBGrid_operace"*

$\{$

$[DataSurce = Form1.IBDataSet\_Nak\_Operac,$
$Height = 120, Left = 344, Top = 24, Width = 320,$
$Name = DBGrid\_operace,$
$PopupMenu = PopupMenu\_operac]$

$\}$

$\vdots$

$\#$ *"DBGrid_operace"*

$\{$

$[Name = PopupMenu\_operac, Items = 28,]$

$[OnClick = N28Click]$

$\}$

$\vdots$

$\#$ *"N28"*

$\{$

$[Caption = Delete, Name = 28]$

$[OnClick = Delete\_select\_operace]$

$\}$

$\vdots$

$</Form1>$

Приклад згенерованого програмного коду реалізації події на елемент *PopupMenu _operac*, при одноразовому натисканні, в випадаючому меню опцій *Delete* з внутрішньою індексацією 28, спрацьовує вибір коду з *LingusticVariable = Delete _select _operace*. Приклад згенерованого програмного коду після редагування розробником представлений на рис. 5.9.

```
procedure TForm1.N28Click(Sender: TObject);
begin
 if messageDlg('Удалить операцию из базы данных?',mtConfirmation,[mbYes,mbNo],0)=mrYes
 then Form1.IBDataSet_NAK_Operac.Delete;
 //Form1.IBDataSet_NAK_Operac.Post;
 Form1.IBDataSet_NAK_Operac.Close;
 Form1.IBDataSet_NAK_Operac.Open;
 DataModule_redaktor.IBStoredProc_update_detal_norma.ParamByName('detal_id').AsInteger := IBDataSet_NAK_DETAL.fieldByName('id_detal').AsInteger;
 DataModule_redaktor.IBStoredProc_update_detal_norma.ExecProc;
 IBDataSet_NAK_DETAL.Refresh;
 IBDataSet_NAK_Operac.Refresh;
 DBGridEh1.Refresh;
end;
```

Рисунок 5.9 – Фрагмент коду після редагування розробником

Ґрунтуючись на прикладах наведених вище, розробник отримує можливість, на базі запропонованої моделі та методів розробки CPPS (2-3 розділ дисертації), а також на базі розробленої моделі ЖЦ (4 розділ), за допомогою метаопису створити і реалізувати адитивний кібер-дизайн, з можливістю «часткової» генерації програмного коду [222]. Повнота генерації програмного коду безпосередньо залежить від змісту DB *“Container Solutions”*, в якому знаходяться приклади реалізації подій (*cod*), які можуть виконуватися в процесі роботи з розробки адитивного кібер-дизайну. Дане рішення дозволяє адаптувати запропонований метаопис до будь-якої об'єктно-орієнтованої мови, а також дає можливість розробнику розширювати БЗ новими *“Container Solutions”*, що дозволить скоротити витрати часу на етапі розробки адитивного кібер-дизайну для керування процесами в складних організаційно-технічних виробничих об'єктах.

Висновки до розділу 5

Аналіз запропонованих математичних моделей та методів розробки адитивного кібер-дизайну, отриманих в 4 розділі дисертаційної роботи,

показав, що вони досить складні в освоєнні і в такому вигляді не сприяють вирішенню завдання автоматизації процесу управління, а також можуть використовуватися як математичне ядро. Це зумовило необхідність розробки нової декларативної мови визначення і маніпулювання даними, близької до деякої підмножині природної мови. Тому, в даному розділі дисертаційної роботи, вирішені наступні задачі:

1. Розроблена специфікація мови моделей даних (ключові слова, ідентифікатор, літерали, типи представлених значень, і т.д.).

2. Запропонована синтаксична діаграма розробленої мови, яка дає можливість описати параметри, властивості, значення, а також події притаманні як *WindowsForm* так і GUI елементам HMI інтерфейсу користувача адитивного кібер-дизайна.

3. Розроблена синтаксична діаграма типів представлення значень ідентифікаторів.

4. За результатами узгодженості розробленого термінологічного базису синтаксичних конструкцій ММ і термінів, розроблено інтерпретатор, який здатний автоматично перекладати складений в термінології близької до деякої підмножині природної мови HMI, в формат команд середовища розробки та мови високого рівня програмування. При цьому самі синтаксичні конструкції розробленої мови, представляють собою скриптову послідовність і є досить простими для розуміння як спеціалістам так і звичайними розробниками (приклади 5.1–5.5). Це дає підстави стверджувати, що використання розробниками запропонованої ММ дає можливість скоротити час розробки адитивного кібер-дизайну керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS.

Список джерел, які використано у даному розділі, наведено у повному списку використаних джерел [216–222].

6 РОЗРОБКА СИСТЕМИ АВТОМАТИЗАЦІЙ ПРОЦЕСІВ КЕРУВАННЯ РОЗРОБКОЮ КІБЕРНЕТИЧНОЇ СКЛАДОВОЇ ТА ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

6.1 Структура системи автоматизації процесів керування організаційно-технічним виробничим об'єктом

Ґрунтуючись на запропонованих моделях і методах керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS, наведених в 2-5 розділах дисертаційної роботи, для підтвердження правильності прийнятих наукових рішень і проведення експериментальних досліджень, необхідно реалізувати їх в систему розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом.

Для системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом, що розробляється, запропонована наступна структура, яка представлена на рисунку 6.1.

Дана структура складається з наступних блоків:

а) текстовий редактор мови моделювання / модуль аналізу вхідних даних з MS Excel (* .xls). Редактор призначений для введення даних в систему за допомогою розробленої ММ і дає можливість забезпечити зчитування вихідних даних альтернативним способом через MS Excel (* .xls), за умови необхідності розробки тільки кібернетичної складової. Формат вхідних даних має жорстку деревоподібну ієрархію, яка дає можливість відразу визначити структуру об'єктів візуалізації і їх взаємозв'язок, підпорядкованість в HMI на базі GUI;

б) модуль аналізу синтаксису вхідних даних, який проводить: перевірку на адекватність і правильність завдання вихідних даних; їх повноту, яка

необхідна для реалізації. При виникненні критичних помилок невідповідності логіки або браку даних, формується файл помилки, в якому відбувається запис про помилки або попередження.

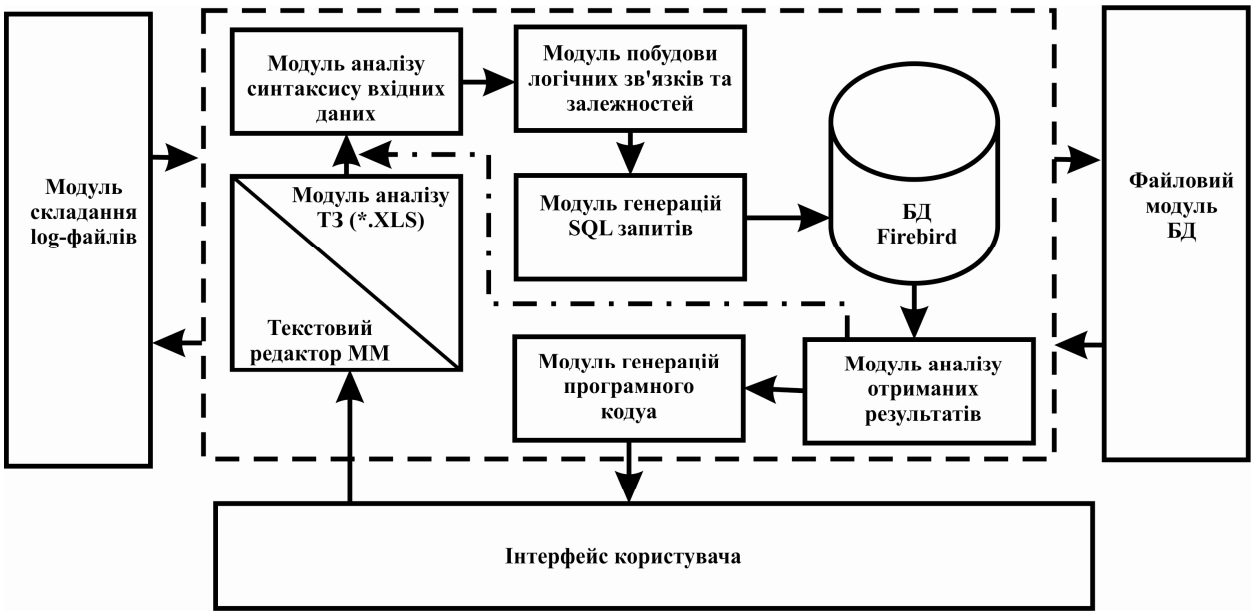


Рисунок 6.1 – Структура системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом

в) модуль побудови логічних зв'язків і залежностей між візуальними компонентами інтерфейсу і подіями, які накладаються розробником на дії для даних подій. Перевірка проводитиметься з використанням контролю взаємозв'язків виконання, відповідно алгоритмів функціонування та вимог ТЗ, обмеженням функціональних можливостей кожного компонента, які залежать від повноти БД і можуть бути доповнені;

г) модуль генерації запитів до БД – генерує запити на мові SQL, відповідно до використаних компонентів і подій (дій), які вони повинні виконувати. В ході виконання запиту формується список шаблонів програмного коду, що підходять під умови вибірки і логіки роботи модуля;

д) модуль аналізу отриманих результатів з БД – проводить синтаксичний аналіз отриманих результатів, визначає максимально

відповідний фрагмент програмного коду, який підходить під умови вибірки та залежнить від заданих умов пошуку;

е) модуль генерації вихідного файлу програмного коду – підключає необхідні бібліотеки, дотримує синтаксис структури, правила оформлення і представляє його у вигляді пакету файлів, які необхідні для відкриття розроблюваного ПП в середовищі розробки, заданих користувачем;

ж) модуль складання log-файлів, які містять звіти по роботі кожного модуля на етапах проектування, а саме, містить в собі такі дані:

1) результати перевірки синтаксису і повноти опису адитивного кібер-дизайну, а також аналіз результатів даних отриманих альтернативним способом через MS Excel (* .xls). У звіті міститься номер рядка і номер комірки, в якій необхідно провести коригування даних або задати їх, рядок може приймати два значення. Значення *error* – критична помилка, в ході якої система не може використовувати дані і *warning* – попередження користувача про можливість некоректного представлення даних, що може вплинути на результат генерації вихідного коду розроблюваного адитивного кібер-дизайну;

2) процентне співвідношення ймовірності генерації програмного коду, відповідно до тих чи інших завдань та залежне від повноти БД за кожною подією (дією) над об'єктом графічного інтерфейсу;

з) БД (Firebird) – вільна кросплатформенна реляційна система управління базами даних, в якій зберігається вся необхідна інформація і посилання на файли, які містять фрагменти програмного коду *("Container Solutions"*);

і) файловий модуль БД, призначений для зберігання файлів *"Container Solutions"*.

6.2 Обґрунтування вибору середовища розробки та мови високого рівня

Для розробки системи кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом був проведений аналіз мов високого рівня програмування і середовищ розробки. Основними критеріями для вибору середовища розробки були обрані наступні параметри:

– підтримка роботи з розподіленими хмарними базами даних;

– робота з мінімальними втратами в машинних ресурсах при обробці великих обсягів інформації;

– можливість розробки своїх візуальних і бібліотечних компонентів.

Ґрунтуючись на вище перерахованих параметрах, був проведений аналіз середовищ розробки під ОС Windows, таких як Red Studio X3 (мова високого рівня програмування Pascal) і Visual C# (мова програмування C ++) [223,224].

З метою скорочення часу реалізації розроблених моделей і методів, а також збільшення економічного ефекту на етапі програмування «Системи розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом», для проведення комп'ютерного експерименту було обране середовище програмування Red Studio X3, яке повністю відповідає всім вимогам щодо реалізації доступу до баз даних.

6.3 Обґрунтування вибору баз даних FireBird

Для реалізації роботи з великими обсягами інформації, а також її зберігання, необхідно обрати сучасну БД, яка буде відповідати наступним вимогам до розроблюваної «Система розробки кібернетичної складової для

автоматизації процесів керування організаційно-технічним виробничим об'єктом»:

– можливість зберігання великих обсягів інформації;

– можливість додавання / видалення інформації;

– швидкість доступу до інформації;

– можливість віддаленого доступу до інформації.

В ході аналізу були розглянуті такі БД:

– реляційна база даних MS SQL;

– реляційна база даних Oracl;

– реляційна база даних PostgreSQL;

– реляційна база даних Interbase;

– не реляційна база даних MongoDB.

Перевагами розробки БД на реляційній моделі полягає в наступному:

– модель даних відображає інформацію в найбільш простій для користувача формі;

– заснована на розвиненому математичному апараті, який дозволяє досить лаконічно описати основні операції з даними;

– дозволяє створювати мови маніпулювання даними не процедурного типу;

– маніпулювання даними на рівні вихідної БД і можливість зміни;

– сучасні PLM системи використовують реляційні бази даних і побудовані на вільно розповсюджуємій системі управління базами даних (Firebird), що дає можливість зменшити загальну вартість «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом», як на момент її впровадження так і безпосередньо під час її розробки.

До основних недоліків можна віднести наступне:

– трудомісткість розробки;

– невелика швидкість доступу до даних.

Не зважаючи на недоліки, які були виявлені в ході аналізу сучасних баз даних, для вирішення поставленого завдання розробки «Системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом», було прийнято рішення реалізувати базу даних на основі Firebird, а в якості СУБД використовувати графічний інтерфейс IBExpert.

Обґрунтуванням вибору БД Firebird і СУБД IBExpert слугувало те, що даний вид БД успішно застосовується провідними виробниками ПЗ з вирішення задач розробки інформаційно-аналітичних модулів для CAx-систем та модулів CAE-аналізу SolidWork.

Дана СУБД легко імпортується і інтегрується з роботою сучасних реляційних баз даних. Тому пропонується використовувати для розробки БД – Firebird 3.0.2 і IBExpert Version 2017.04.24 [225].

6.4 Розробка логічної і фізичної моделі бази даних

За розробленими математичними моделями і методами, а також запропонованою технологією, необхідно оперувати великими масивами інформації, які мають взаємопов'язану структуру з інформаційними блоками прийняття рішень, візуалізацією і складно-підлеглими залежностями в ієрархії побудови і генерації вихідного програмного коду.

Для спрощення і систематизації зберігання інформації, необхідної для вирішення поставленого завдання, було прийнято рішення розбити логічну структуру БД на модулі, які будуть поєднані по «суті» зберігаємої інформації, а зв’язок між ними буде реалізовано з використанням «зовнішніх ключів» типу підключення «один-нескінченність». Запропонована логічна структура БД представлена на рисунку 6.2.

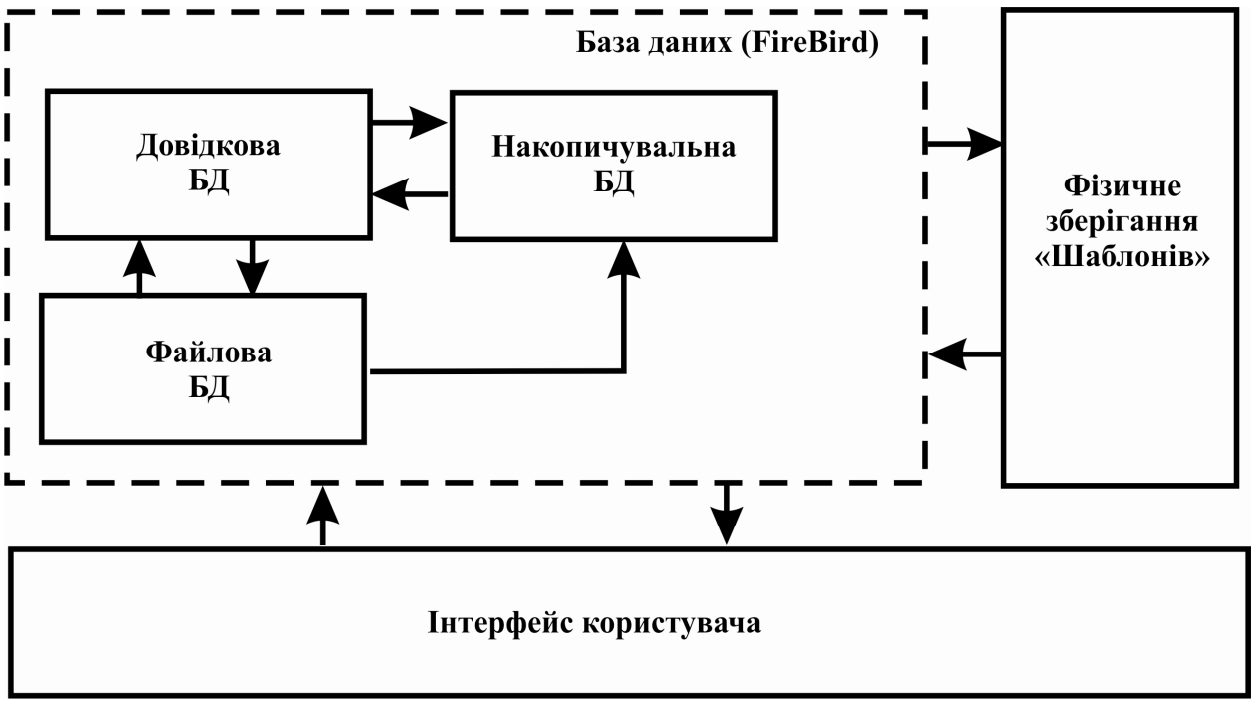


Рисунок 6.2 – Логічна структура БД

Логічна структура БД представлена у вигляді модулів, які виконують такі функції:

– довідкова БД являє собою ієрархію таблиць, в яких міститься: інформація про мови програмування; структура коду; бібліотеки і взаємозв'язок їх використання; необхідні процедури і функції, які можна використовувати; події і можливі обробники дій над графічними елементами інтерфейсу; вбудовані процедури. Всі збережені дані в довідковій БД підкорюються критерію вибору середовища розробки та мови програмування. Довідкова БД реалізована з урахуванням розширення і адаптації до того чи іншого середовища розробки, користувач з правами адміністратора може додавати нові шаблони коду і прив'язувати їх до подій, або до внутрішніх процедур чи функцій;

– накопичувальна БД призначена для зберігання поточних проектів та проектів, реалізованих в даній системі. Структура накопичувальної БД є жорстко структурованою залежністю від: даних замовника; назви проекту і технічного завдання; згенерованого вихідного коду і інтерфейсу

користувача; можливістю редагування будь-якого елементу конструкції і шаблонів, які застосовані в даному проекті. Користувач системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом, може редагувати визначення і сутності даного проекту, змінювати взаємозв'язок і послідовність вставки кожного шаблону (контейнера) з програмним кодом, в залежності від вимог замовника;

– фізичне зберігання «шаблонів» являє собою сукупність сутностей, що підкоряються довідковій БД. Ця сукупність містить в собі посилання на фізичну папку, в якій знаходяться файли з мінімальною і необхідною структурою кожної мови програмування для компіляції порожнього проекту.

Уявімо кожен модуль у вигляді фізичної моделі БД, яка реалізована в «Системі розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом». Запропонована фізична структура побудови зв'язків таблиць заснована на використанні металінгвістичної формули Бекуса-Наура (мова БНФ). Вона дає можливість систематизувати мови високого рівня програмування у вигляді блоків, заголовків програми і тіла програм, як «сутності» (елементи мови, організацію дій над даними, організацію даних і субпідлеглих «сутностей»: алфавіт, лексеми, синтаксис, оформлення програмного коду, елементи введення-виведення даних (елементи інтерфейсу, роботи з БД, робота з файлами), обробка даних (події над елементами форм (операції і вирази, оператори, організація і використання підпрограм)) [226].

Інформаційна модель взаємозв'язку основних «сутностей» довідкової БД представлена на рисунку 6.3

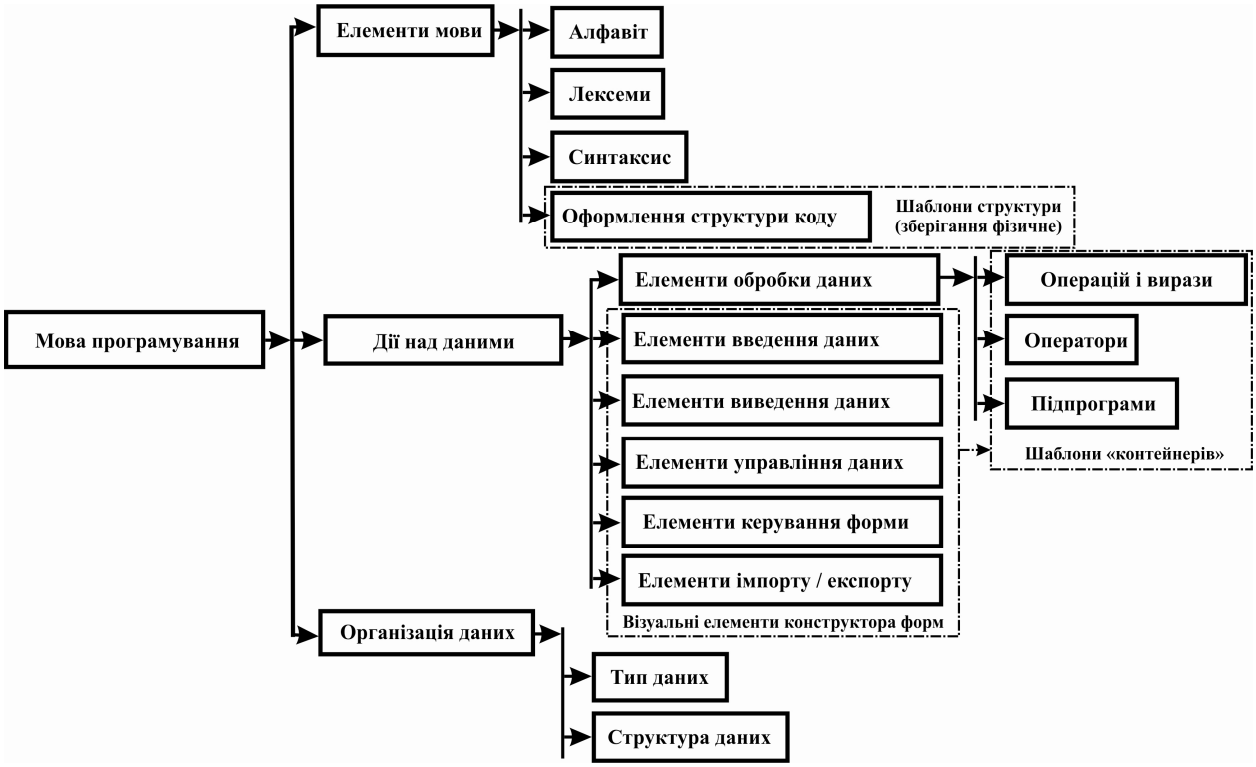


Рисунок 6.3 – Інформаційна модель взаємозв'язку основних «сутностей» довідкової БД

Структура фізичної моделі довідкової БД за зв'язками сутностей: «Мова програмування», «Елементи мови», «Алфавіт», «Лексеми», «Синтаксис», «Оформлення структури коду» представлено на рисунку 6.4.

На рисунку 6.5 представлена фізична модель довідкової БД за зв'язками сутностей: «Дія над даними»→«Елементи введення даних», «Елементи виведення даних», «Елементи управління даними», «Елементи управління форми», «Елементи імпорту / експорту» а також зв'язкам: «Елементи обробки даних», «Операції і вирази», «Оператори», «Підпрограми».

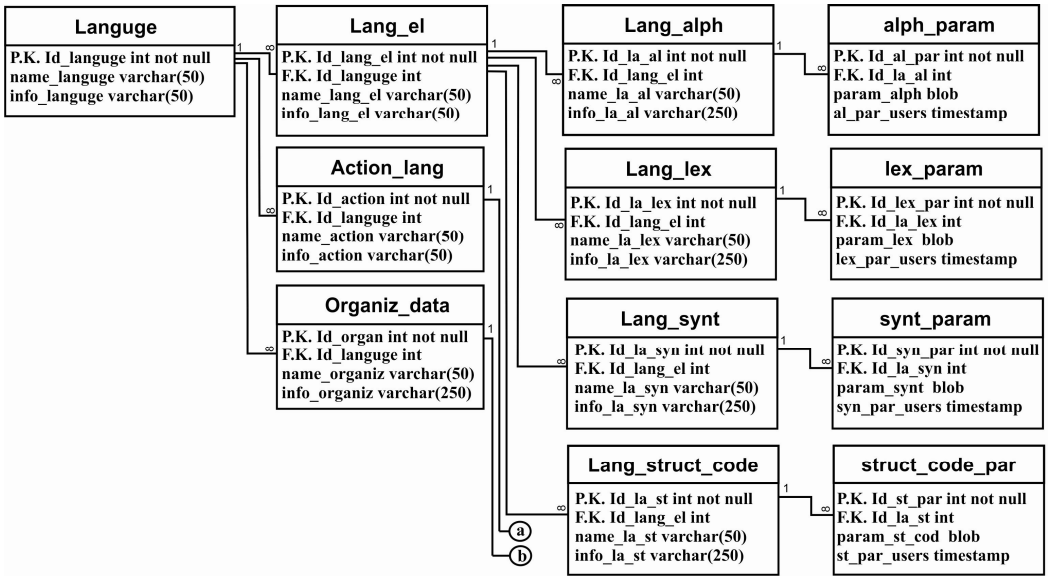


Рисунок 6.4 – Фрагмент структури фізичної моделі довідкової БД за зв'язками сутностей: «Мова програмування», «Елементи мови», «Алфавіт», «Лексеми», «Синтаксис», «Оформлення структури коду»

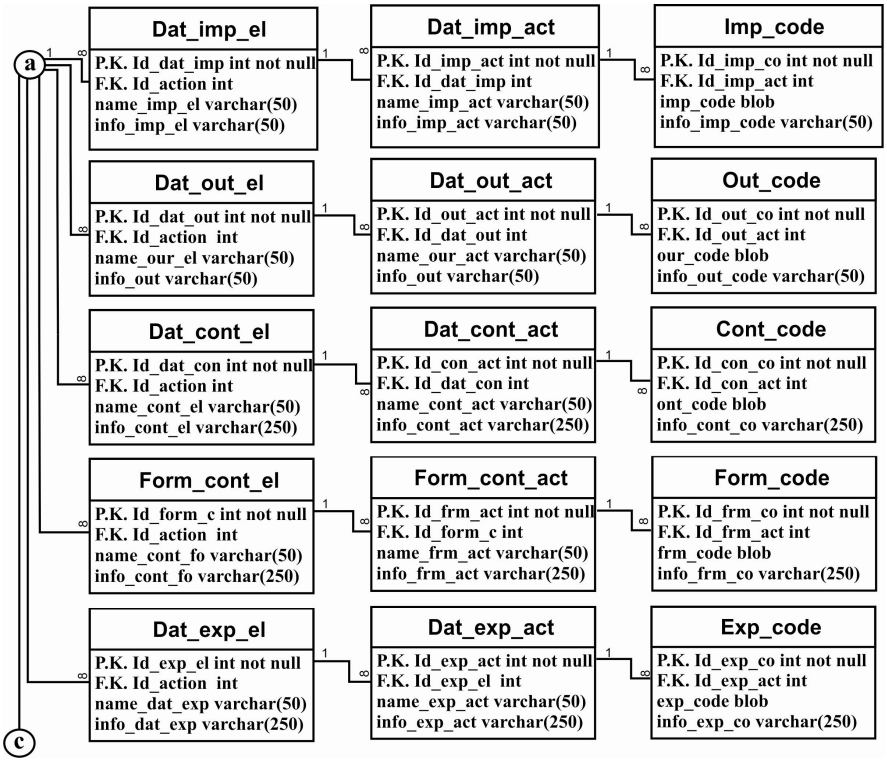


Рисунок 6.5 – Фрагмент структури фізичної моделі довідкової БД за зв'язками сутностей: «Дія над даними»→«Елементи введення даних», «Елементи виведення даних», «Елементи управління даними», «Елементи управління формами», «Елементи імпорту / експорту»

Реалізація фрагменту зв'язків зберігання інформації в довідковій БД за сутностями «Елементи обробки даних»→«Операції і вирази», «Оператори», «Підпрограми» представлена на рисунку 6.6.

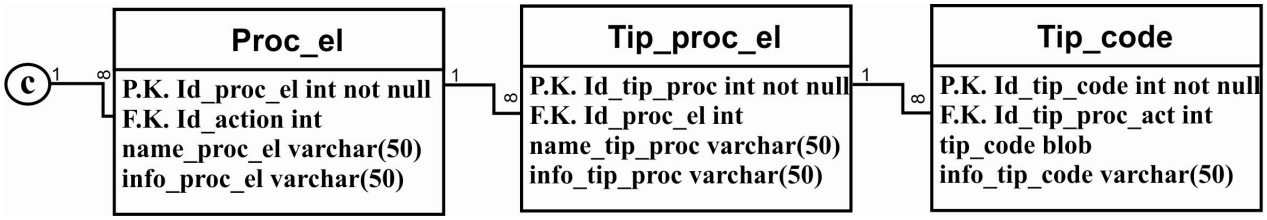


Рисунок 6.6 – Фрагмент зв'язків зберігання інформації в довідковій БД по сутностей «Елементи обробки даних»→«Операцій і вирази», «Оператори».

Фрагмент структури фізичної моделі довідкової БД для зв'язків сутностей «Організація даних»→«Тип даних», «Структура даних» представлена на рисунку 6.7.

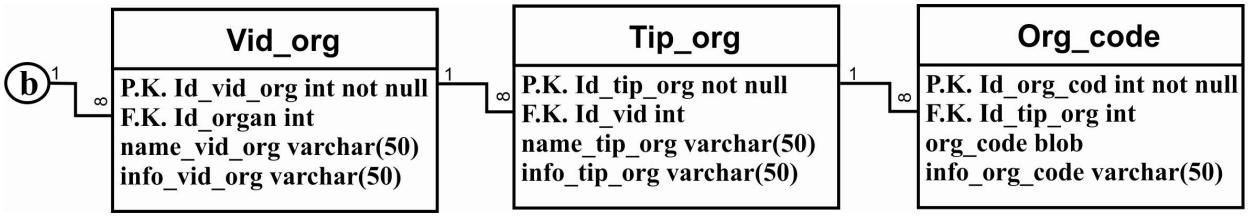


Рисунок 6.7 – Фрагмент структури фізичної моделі довідкової БД для зв'язків сутностей «Організація даних»→«Тип даних», «Структура даних»

Для зберігання довідкової інформації «шаблонів» і синтаксису мови SQL пропонується наступна інформаційна модель (рис. 6.8).

На базі запропонованої інформаційної моделі розроблена фізична модель БД, яка складається з наступної послідовності зв'язків сутностей: «SQL»→«Тип команд»→«Синтаксис команди»→«Шаблони коду». Дане рішення дозволить користувачу систематизувати «контейнери» під загальний синтаксис команд мови SQL.

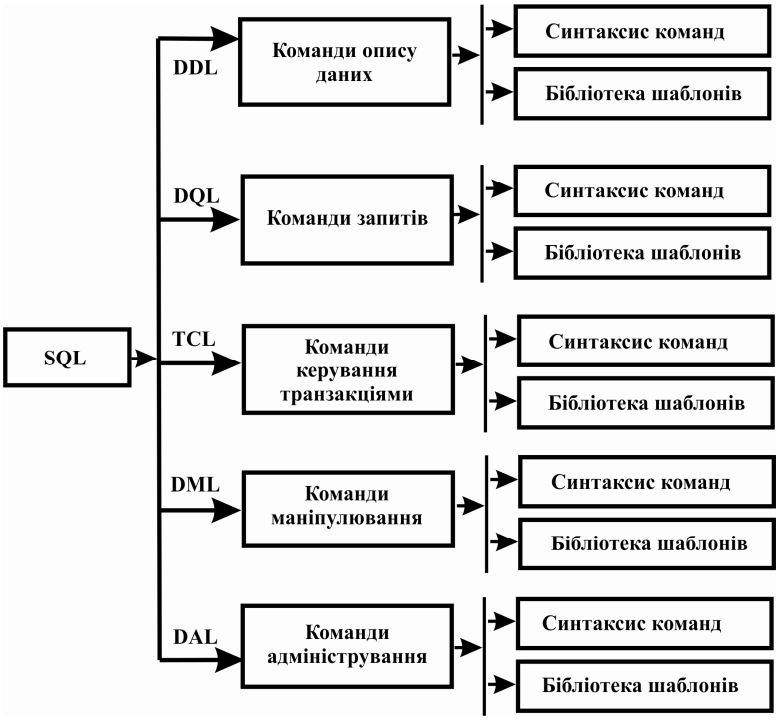


Рисунок 6.8 – Інформаційна модель довідкової БД для зберігання SQL

Поле «Id_action» в таблиці «Tip_sq» зберігає значення первинного ключа таблиці «Action_lang» і дає можливість зв'язати команди SQL з візуальними елементами і діями над даними.

Структура фізичної моделі представлена на рисунку 6.9.

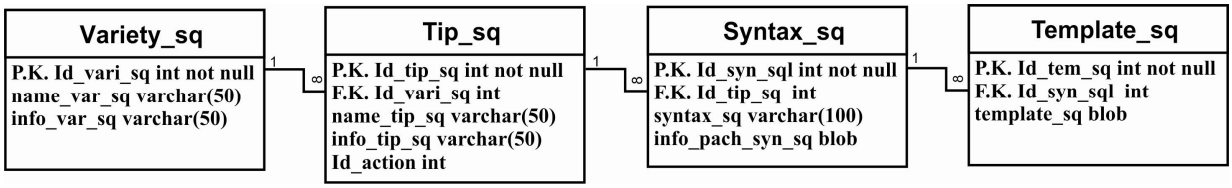


Рисунок 6.9 – Фізична модель довідкової БД для зберігання шаблонів команд SQL

Запропонована фізична структура довідкової БД реалізована у вигляді 37 таблиць, що містять більше 110 полів зберігання і взаємодії інформаційних залежностей, які повністю відповідають законам нормалізації (відсутність надмірності і неузгодженості між сутностями). Все це робить БД гнучкою і адаптивною в роботі.

Для системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом запропонована накопичувальна БД, яка реалізує функції роботи із замовником. У ній зберігається інформація про поточні проекти CPPS, що дозволяє проводити коригування вихідного коду, а також використовувати накопичену базу в якості шаблонів для створення нових проектів з мінімальним коригуванням. Це дозволить скоротити час проектування і домогтися максимальної економічної ефективності використання «Системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом». Інформаційна модель накопичувальної БД представлена на рисунку 6.10.

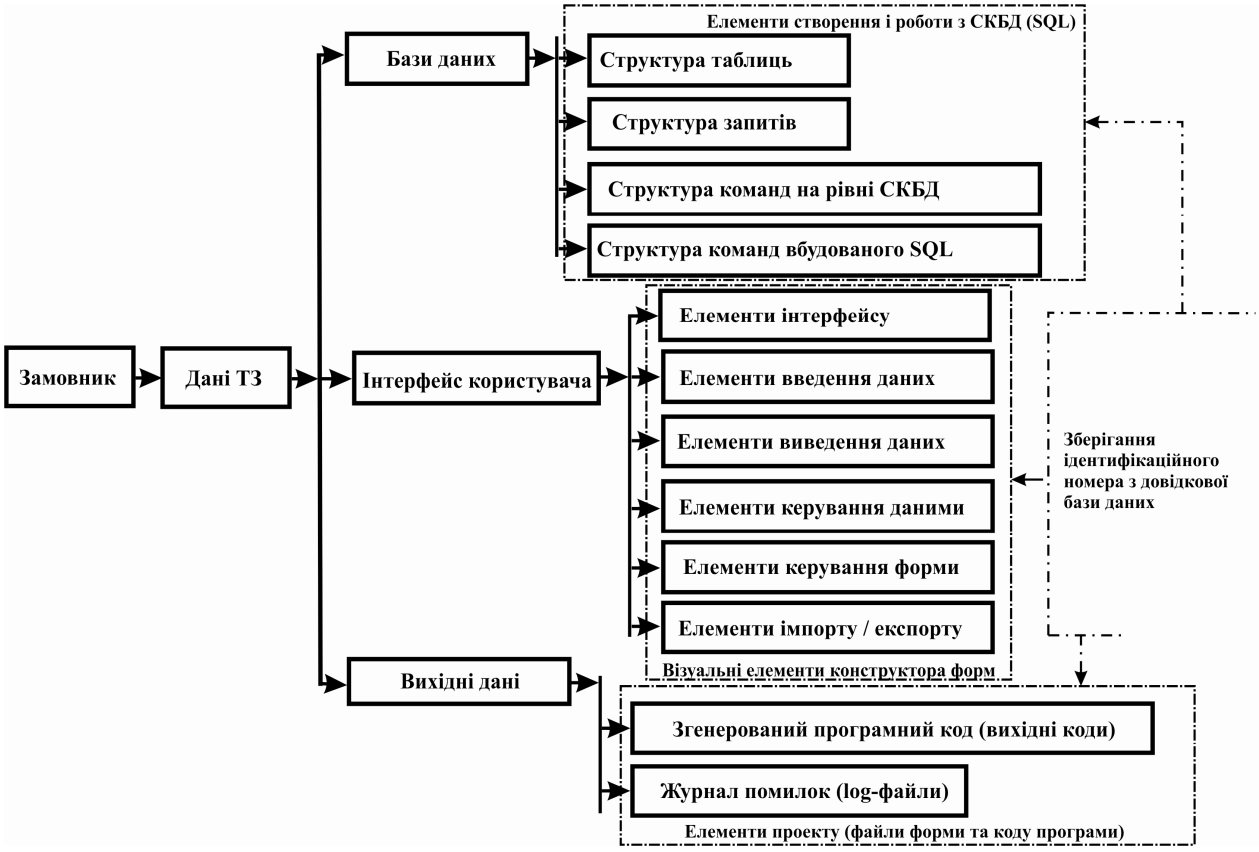


Рисунок 6.10 – Інформаційна модель накопичувальної БД

Спроектована фізична модель накопичувальної БД представлена на рисунку 6.11.

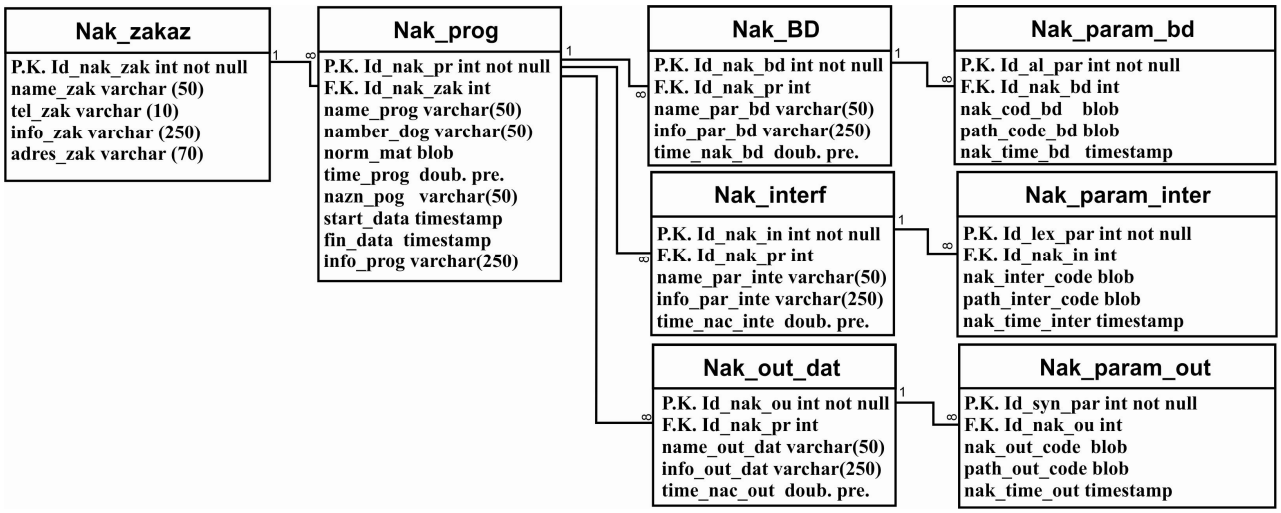


Рисунок 6.11 – Фізична модель накопичувальної БД

6.5 Розробка системи автоматизації процесу управління розробкою кібернетичної складової CPPS

Для апробації запропонованих моделей та методів, а також їх програмної реалізації у вигляді системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом, був реалізований інтерфейс користувача, з урахуванням ергономічності і інтуїтивності розуміння користувачем.

Враховуючи, що користувачу необхідно одночасно оперувати великими об'ємами інформації, запропоновано використовувати метод MDI інтерфейсу, який дозволить контролювати велику кількість інформації і максимально логічно провести її групування для зручності доступу і контролю.

На рисунку 6.12 представлене вікно ініціалізації і завантаження бібліотек для запуску системи.

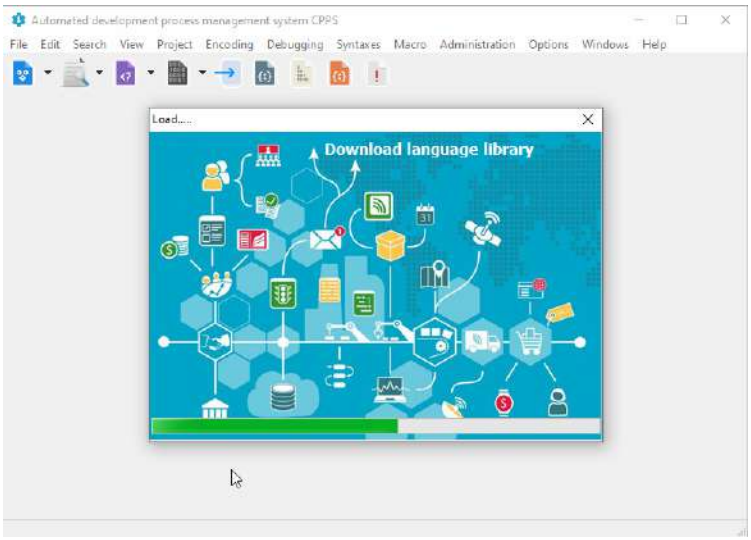


Рисунок 6.12 – Вікно ініціалізації і завантаження бібліотек з БД

Після успішної ініціалізації і підключення бібліотек: «Language library», «CPPS Structure», параметрів «Physical world parameters» і «Cyber world parameters», а також «Software modules» і «Language compilers», інформація для яких зберігатися в БД, користувачу надається Select menu вибору основних функцій системи, який представлено на рисунку 6.13.

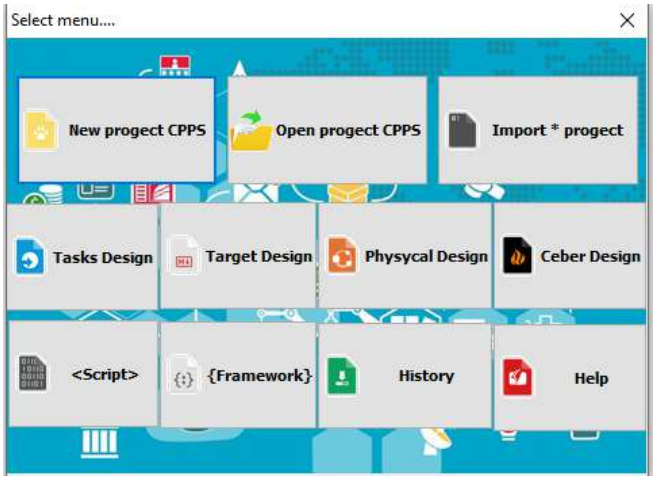


Рисунок 6.13 – Select menu вибору основних функцій системи

До основних функцій системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом, представленим в Select menu відносяться:

– New progect CPPS – модуль створення нового проекту з п'ятого рівня опису головної мети розробки CPPS, а також цілей на кожному рівні розробки і відповідності завдань, вирішення яких дозволяють досягти головної мети. Послідовність процесу керування розробкою CPPS представлений на рисунку 2.1;

– Open progect CPPS – відкриття вже існуючого проекту, який було збережено у вигляді окремих файлів. В контексті даної реалізації пропонуються наступні розширення файлів доступних до завантаження в систему:

– * .tas – файл містить інформацію про цілі розробки CPPS (Tasks Design), опис яких представлено в підрозділі 2.2;

– * .tar – файл містить інформацію про завдання розробки CPPS (Target Design), опис яких представлено в підрозділі 2.2;

– * .phy – файл містить інформацію про фізичну складову CPPS (Physycal Design), опис представлено в підрозділі 2.3;

– * .cyb – файл містить інформацію про кібернетичну складову CPPS (Cyber Design), опис представлено у підрозділі 2.4 та у розділі 4;

– * .scr – файл містить інформацію про синтаксичну та семантичну мовні моделі CPPS (<Script>), опис яких представлено у 5 розділі дисертаційної роботи;

– * .frm – файл визначає структуру і правила побудови архітектури процесу управління розробкою CPPS, який задає поведінку за умовчанням, на початковому етапі розробки (рис. 2.1) і ускладнюється з кожною ітерацією, відповідно до запропонованої технології керування розробкою CPPS, яку представлено на рисунку 2.7 ({Framework}).

– модуль Import * progect – дає можливість проводити імпорт даних для реалізації візуалізації інтерфейсу користувача, в рамках кіберскладової CPPS, з альтернативних джерел з «жорсткою» структурою подання інформації (* .xml - Imports from XML, *. xls- Imports from Excel 2003 and under, *. xlsx - Imports from Excel 2007 and over).

Дане рішення обумовлене завданням швидкої розробки прототипу візуалізації інтерфейсу кіберскладової CPPS при безпосередній участі замовника. Це дає можливість вже на ранньому етапі процесу управління розробкою CPPS отримати вихідні дані для чіткого розуміння вимог до графічного інтерфейсу.

Варто зауважити, що розроблений метод (підрозділ 6.6) уявлення інформації для розробки інтерфейсу користувача має один істотний недолік, який міститься у необхідності розробки «жорстко» фіксованої, підпорядкованої структури представлення параметрів і значень візуальних форм і об'єктів інтерфейсу користувача, залежно від мови та середовища розробки CPPS;

– модуль Tasks Design і Target Design дають можливість розробити і визначити цілі розробки адитивного кібер-дизайну на кожному рівні, а також обрати необхідні завдання та вимоги до параметрів і характеристик їх досягнення;

– Physycal Design – модуль, який дозволяє користувачу описати послідовність процесу керування розробкою і вибору параметрів і зв'язків на функціональному і організаційно-технічному етапі, а також визначає технічні характеристики атомарних елементів (фізичні і електричні параметри датчиків і виконавчих пристроїв, відповідно до їх специфікації), які дозволять досягти вимог до завдань, що ставляться перед ними;

– модуль Cyber Design – дає можливість, на базі обраних і реалізованих параметрів адитивного кібер-дизайну, розробити інфологічну і інформаційну структуру. Це дасть можливість отримати алгоритми функціонування і керування процесами в складних організаційно-технічних виробничих об'єктах, на базі яких можна реалізувати автоматизацію отримання прототипу інтерфейсу користувача адитивного кібер-дизайну і його програмний код для обраного середовища розробки;

– модуль <Script> – представляє собою текстове середовище розробки адитивного кібер-дизайну, на базі запропонованих синтаксичних

правил і семантичної мовної моделі, фрагменти якої представлені в 5 розділі;

– {Framework} – модуль («каркас») архітектури процеса керування розробкою CPPS, на якому користувач може бачити не тільки процес розробки, а й взаємозв'язки між усіма етапами і рівнями, перевірки цілісності досягнення головної мети розробки CPPS. Основним завданням модуля {Framework} є послідовна візуалізація зв'язків, від початкового етапу процесу керування розробкою, до отримання алгоритмів функціонування і реалізація на їх базі візуальних компонентів та генерації програмного коду розроблюваного CPPS;

– модуль History містить текстову інформацію з даними про типи та види внесених змін в проекти адитивного кібер-дизайну, час внесення і ім'я користувача, що дає можливість відстежувати зміни та некоректні правки, щоб зробити відкат до останньої робочої версії;

– модуль Help містить повну довідкову інформацію про архітектуру, послідовності, правила роботи з системою.

Розглянемо основні модулі, які необхідні для реалізації процесу управління розробкою кіберскладової CPPS. На рис. 6.14 представлено вікно модуля «New progect CPPS».

Як можна бачити, на даному етапі задаються такі дані: «Project number», «Name», «Customer». У випадаючому меню «Level» розробник обирає етап розробки, який може бути: «Target» – етапи цілей і завдань, «Physical» – етапи функціонального рівня і організаційно-технічної структури, «Cybernetic» – етапи інфологічної і інформаційної структури управління, а також алгоритмів функціонування, «Full life cycle "Jump"» – процес управління розробкою CPPS з «нуля»; «Specificity» – містить найменування і типи ЧПК обладнання, опис головної мети і завдань які повинна вирішувати CPPS; «Language» і «Development environment» – вибір мови та середовища розробки; вибір типу БД відбувається у

випадаючому меню «DB type», а вибір системи управління баз даних (СУБД) у «DBMS» відповідно.

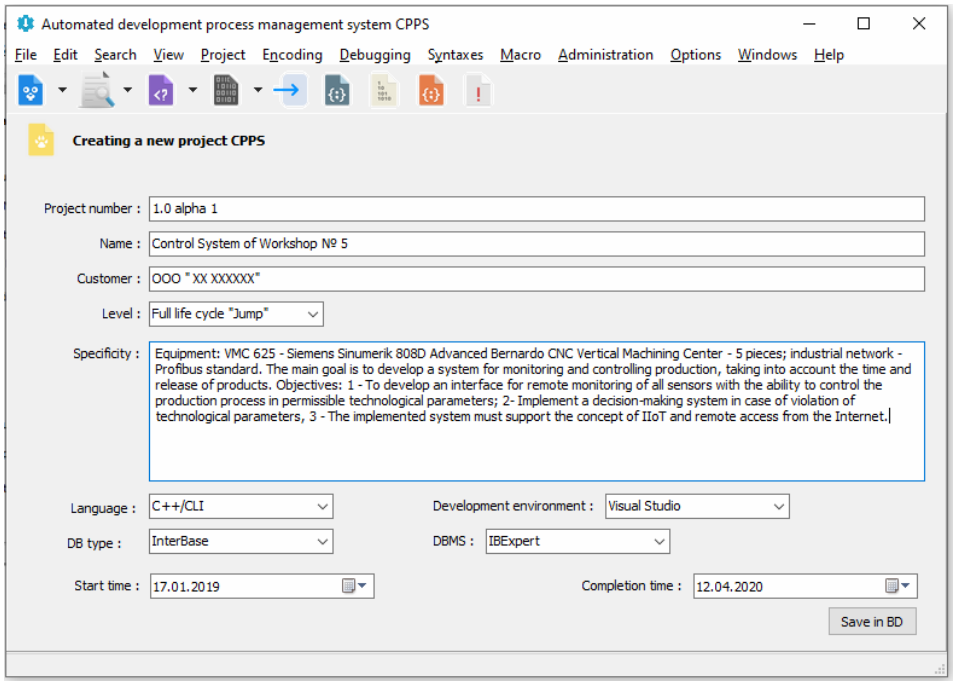


Рисунок 6.14 – Вікно модуля «New project CPPS»

Після збереження нового проекту в БД, за допомогою кнопки «Save in BD» відкривається вікно, представлене на рисунку 6.15, яке дозволяє розробнику обрати етап «Level»=«Full life cycle "Jump"». Отже, розробнику доступні в «Project tree» всі етапи і рівні розробки адитивного кібер-дизайну. Обираючи необхідний етап і рівень розробник може використовувати текстове середовище розробки <Script>, в даному випадку це кіберскладова CPPS на етапі «Control algorithms» на рівні «Decomposition level 0», в якій за допомогою семантичної мовної моделі описуються всі необхідні параметри і зв'язки для реалізації графічного інтерфейсу користувача адитивного кібер-дизайну для керування процесами в складних організаційно-технічних виробничих об'єктах.

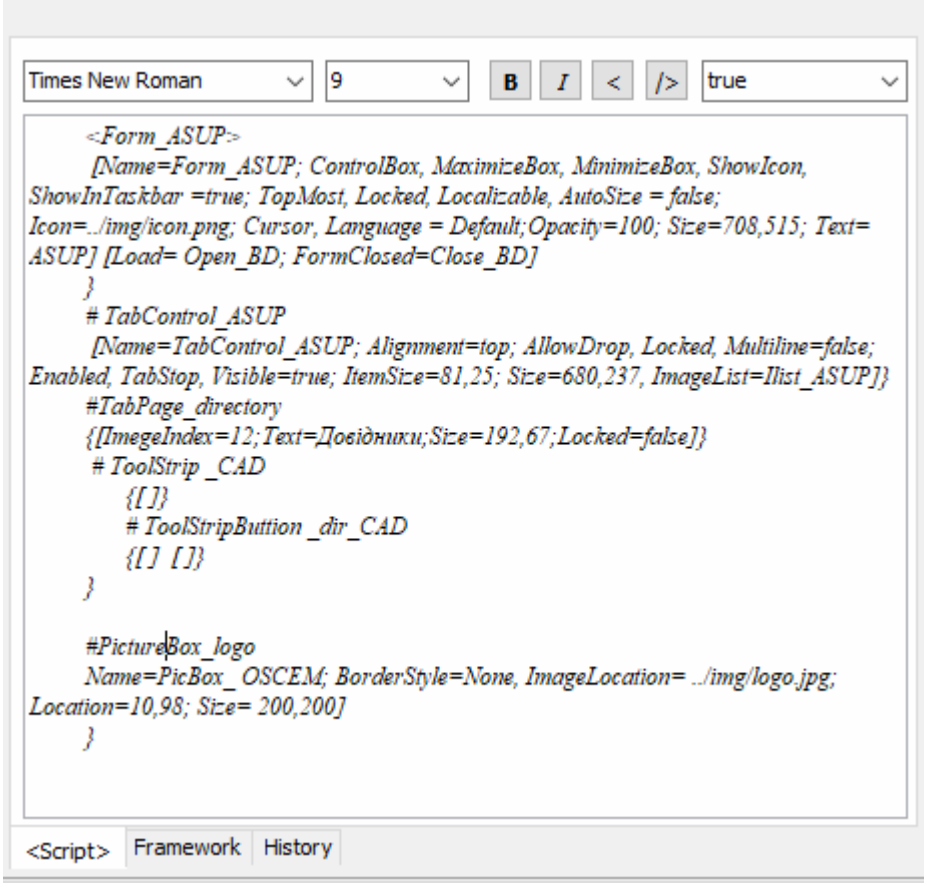


Рисунок 6.15 – Вікно на етапі «Control algorithms» рівня «Decomposition level 0»

При виконанні команди «Debugging» в системі була реалізована синтаксична і логічна перевірка написаного коду з виділенням рядків, де сталася критична помилка, або видається попередження про можливу логічну невідповідність вимогам, яка в подальшому позначитися при загальній компіляції проекту адитивного кібер-дизайну.

Обираючи закладку {Framework} розробник стає доступною візуалізація зв'язків від початкового етапу процесу керування розробкою до отримання алгоритмів функціонування, з точною прив'язкою до рівня «Decomposition level 0». Це дозволяє, в процесі керування розробкою, візуально спостерігати зміну кількості зв'язків їх характеристик і прив'язок по всій архітектурно-логічній моделі керування процесами в складних організаційно-технічних виробничих об'єктах.

Детальніше принцип роботи в системі розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом буде приведено в розділі 6.7, в якому будуть проводитись експериментальні дослідження з розробки фрагменту реалізації адитивного кібер-дизайну, у відповідності з господарським договором.

6.6 Розробка структури представлення даних для створення прототипу графічного інтерфейсу адитивного кібер-дизайну

Одним із завдань, даного дослідження, є створення візуального графічного інтерфейсу користувача у вигляді прототипу, із зазначенням всіх необхідних і достатніх об'єктів відображення інформації. Розроблюваний прототип повинен відповідати вимогам по інформативності та зручності подання даних, ергономічності і зручність.

Для вирішення даного завдання рекомендується запрошувати представника замовника і обговорювати всі вимоги до розміщення графічних елементів інтерфейсу, їх тип і функції, які вони повинні виконувати. Одним зі складних моментів у роботі з замовником є швидкість розробки прототипу адитивного кібер-дизайну і внесення змін та коригувань в режимі реального часу й демонстрація отриманого результату.

Виходячи їх цього, в даній дисертаційній роботі, розроблена структура представлення даних для створення прототипу адитивного кібер-дизайну, яка реалізована в модулі Import * progect (рис. 6.13). Ґрунтуючись на позиції, що структура представлення даних для створення прототипу адитивного кібер-дизайну повинна дотримуватися всіх правил і вимоги до об'єктно-орієнтованого програмування і прив'язки до мови та середовища розробки, робимо припущення, що структура повинна мати «жорстко»

підпорядковану ієрархію представлення даних і властивостей, а також подій по кожному графічному об'єкту інтерфейсу.

Виходячи з висунутих припущень пропонується використовувати наступні формати файлів, які дозволяють реалізувати вимоги:

– * .xml – Imports from XML;

– *. xls – Imports from Excel 2003 and under;

– *. xlsx – Imports from Excel 2007 and over.

Варто зауважити, що інформація про дані, властивості, події і значення по кожному графічному об'єкту інтерфейсу для різних середовищ розробки може відрізнятися, отже це говорить необхідність створення шаблонних структур для кожного середовища розробки.

Структура представлення даних для створення прототипу адитивного кібер-дизайну повинна повністю описувати його функціональність, специфіку, і додаткові функції, які можна буде інтерпретувати як запити до БЗ (базі знань), за допомогою запропонованих математичних моделей і методів опису та прийняття рішень. Виходячи з цього, була розроблена структура представлення даних (підрозділи 4.3-4.4), у вигляді безлічі взаємно залежних параметрів.

Дані параметри, в сумі, дають можливість окреслити адитивний кібер-дизайн, що розробляється, на базі простого природної мови опису даних в легко доступній і інтуїтивно зрозумілій формі, використовувати стандартні прикладні програмні рішення, що відповідають міжнародним форматам експорту / імпорту даних, і при цьому мати структуроване представлення даних.

Структуризацію властивостей, параметрів і подій, відповідно до запропонованих рішень, можна уявити, як складнопідпорядковану структуру з прив'язкою до подій, які виникають при роботі з інтерфейсом, залежно від дій користувача.

На рисунку 6.16 представлена графічна структура зв'язків властивостей і параметрів, подій і дій для $Form_1^{master}$.

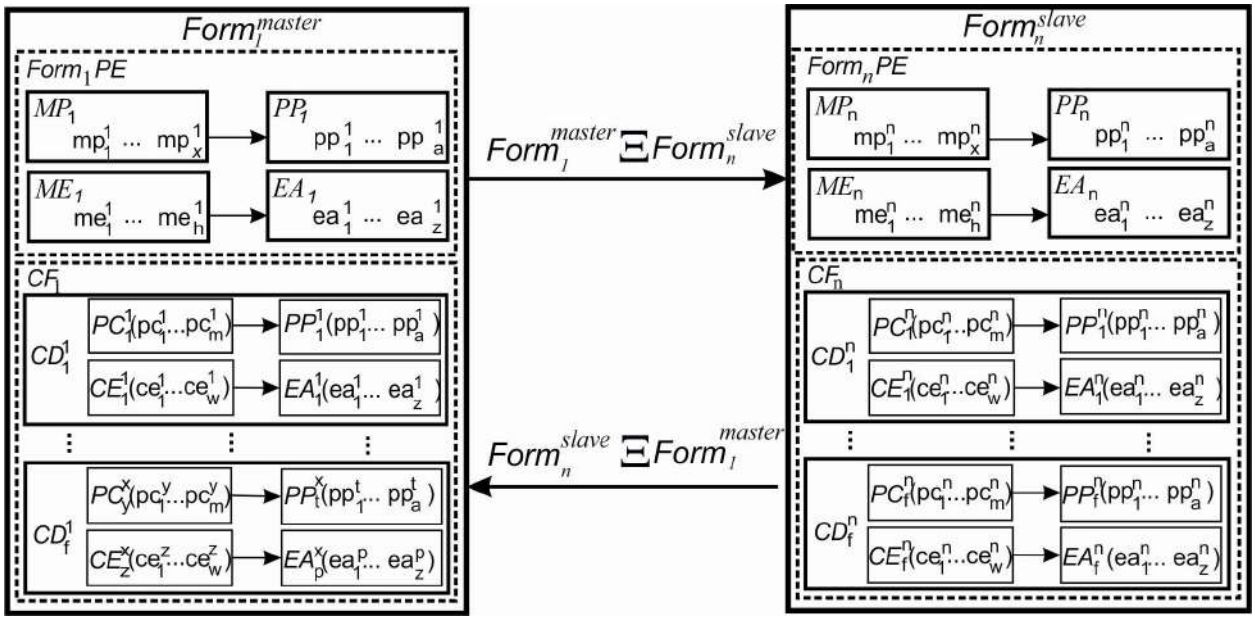


Рисунок 6.16 – Графічна структура зв'язків властивостей, параметрів, подій і дій для $Form_1^{master}$

Представлена концепція побудована на застосуванні графічних елементів інтерфейсу як базових носіїв інформацій про розроблюваний адитивного кібер-дизайну для керування процесами в складних організаційно-технічних виробничих об'єктах.

Під $Form_1^{master}$ розуміється головна форма адитивного кібер-дизайну, яка характеризується сукупністю основних блоків: $Form_1 PE$, CF_1. Блок $Form_1 PE$ складається із взаємозв'язаних структурних елементів MP_1: параметри $mp_1^1,...,mp_t^1 \to PP_1$: значення $pp_1^1,...,pp_q^1$. Кожному mp_t^1 відповідає одне pp_q^1, яке може набувати значень цифрових, логічних операцій (false, true) або зарезервованих параметрів опису значень pp_q^1 під певну мову високого рівня програмування (Top, Batten, і.т.д.). Набір таких параметрів суворо обмежений і несе в собі інформацію про умови візуального відображення $Form_1^{master}$ для користувача (по центру робочого столу, розгорнута на весь робочий стіл, і т.д.). Блок ME_1 являє собою сукупність

подій $me_1^1,...,me_h^1$, які можна накласти на $Form_1^{master}$, при обробці дій у вигляді «Create form», «Close form» і т.д. Набір подій і параметрів у вигляді програмного коду строго визначений для кожної мови і середовища розробки. Кожному з подій $me_1^1,...,me_h^1$ відповідають $ea_1^1,...,ea_z^1$ блоку EA_1, які можуть набувати значень у вигляді функцій і процедур взаємодії з користувачем, описаних у вигляді фрагментів програмного коду.

Кожна $Form_1^{master}$ має нескінченний набір CF_1, який представляє собою структуру $Form_1^{master}$ у вигляді дерева побудови, де кожен елемент дерева – набір візуальних компонентів, призначених для роботи з даними і елементами групування (елементи введення (Edit) і відображення даних (Grid), елементи інтерфейсу (Menu), елементи групування (GroupBox)).

Кожен елемент представлений в блоці CD_1^1 є взаємозв'язком $PC_1^1 \to PP_1^1$. Ґрунтуючись на запропонованій структурі PC_1^1, яка володіє такими ж властивостями як елемент MP_1, але при цьому CE_1^1 володіє нескінченною варіацією дій EA_1^1, які можуть співпадати з EA_1 (звернення до БД, розрахунки, закриття форми, і т.д.) при виклику певної події (натискання на компонент «Button», наведення мишки, і т.д.).

Залежно від загальної структури побудови адитивного кібер-дизайну (скільки елементів типу, $Form_1^{master} ... Form_n^{slave}$, буде використовуватися) необхідно врахувати передачу глобальних змінних і функцій переходу між вікнами інтерфейсу. Внаслідок цього взаємодію між елементами $Form_1^{master} \Xi Form_n^{slave}$ має бути враховано в рамках розроблюваного дизайна, у середньому кількість елементів типу $Form_1^{master}$ і $Form_n^{slave}$ може коливатися від 1 ... 25-30 і вище, вони можуть викликатися спливаючим всередині головної $Form_1^{master}$ і посилатися на неї.

Ґрунтуючись на запропонованій графічній структурі зв'язків властивостей і параметрів, подій і дій для $Form_1^{master}$, розробимо структуру представлення даних в пакеті Excel 2003, для створення прототипу адитивного кібер-дизайну мовою програмування Pascal у середовищі розробки Red Studio X5, яку представлено на рисунку 6.17.

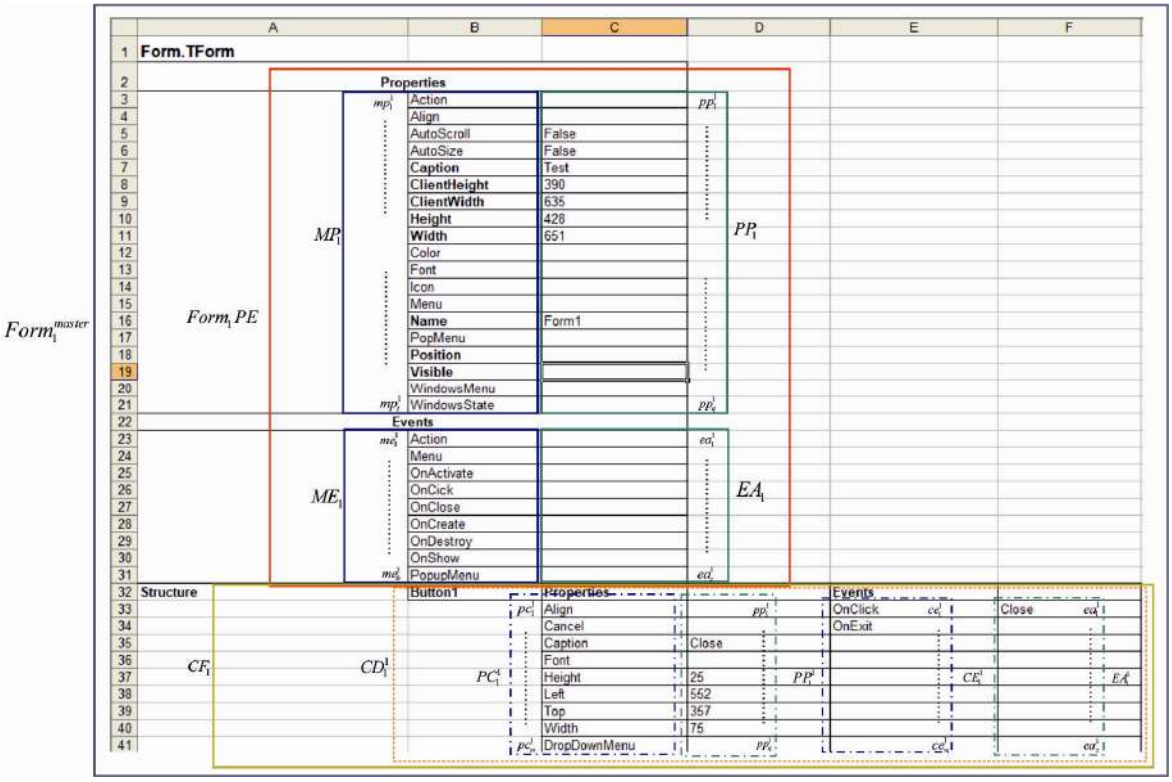


Рисунок 6.17 – Приклад структури представлення даних в пакеті Excel 2003 мовою Pascal в середовищі розробки Red Studio X5 для створення прототипу адитивного кібер-дизайну через модуль Import * progect

Фрагмент заповненої структури представлення даних по створенню прототипу $Form_1^{master}$ у блоці $Form_1 PE$ із зв'язками $MP_1 \rightarrow PP_1$ і $ME_1 \rightarrow EA_1$ приведений на рисунку 6.17.

Приклад наведеного фрагменту структури представлення даних (рис. 6.18) дозволить в модулі Import * progect згенерувати програмний код мовою Pascal для середовища розробки Red Studio X5 для головного вікна із заданими розмірами і параметрами, відповідно вимог замовника.

Генерований програмний код відкривається в середовищі розробки і уточняються вимоги до візуального графічного інтерфейсу.

1	**Form1.TForm**		
2	**Properties**		
3		Action	
4		Align	
5		AutoScroll	False
6		AutoSize	False
7		**Caption**	Test
8		**ClientHeight**	390
9		**ClientWidth**	635
10		**Height**	428
11		**Width**	651
12		Color	
13		Font	
14		Icon	
15		Menu	
16		**Name**	Form1
17		PopMenu	
18		**Position**	
19		**Visible**	
20		WindowsMenu	
21		WindowsState	
22	**Events**		
23		Action	
24		Menu	
25		OnActivate	
26		OnCick	
27		OnClose	
28		OnCreate	
29		OnDestroy	
30		OnShow	
31		PopupMenu	

Рисунок 6.18 – Фрагмент заповненої структури представлення даних по створенню прототипу $Form_1^{master}$

Для створення об'єктів керування або візуалізації інформації до блоку CF_1 (рис. 6.17) додається елемент керування або візуалізації CD_1^1. Фрагмент заповненої структури представлення даних по створенню візуального графічного елементу «Button1», а також елемент групування візуальних елементів «GroupBox1», до якого входить елемент вводу текстової інформації «Edit1», представлено на рисунку 6.19.

У стовпець «Events», який є блоком CE_1^1, в рядку «OnClick» (ce_1^1), реалізована подія на одноразове натискання графічного елементу «Button» (ea_1^1), при якому викликається «*LingusticVariable*» з ім'ям «Close», яка посилається на «*ContainerSolutions*», який містить готовий програмний код функції яка закриває $Form_1^{master}$. Більш детально математичний опис механізму взаємодії описаний в (4.52) і (4.53).

32	**Structure**	**Button1**	**Properties**		**Events**	
33			Align		OnClick	Close
34			Cancel		OnExit	
35			Caption	Close		
36			Font			
37			Height	25		
38			Left	552		
39			Top	357		
40			Width	75		
41			DropDownMenu			
42		**GroupBox1**	**Properties**		**Events**	
43			Align		OnClick	
44			Cancel		OnExit	
45			Caption	Test		
46			Font			
47			Height	343		
48			Left	8		
49			Top	8		
50			Width	619		
51			DropDownMenu			
52			TabOrder	1		
53			**Edit1**	**Properties**		**Events**
54				Align	alNone	OnClick
55				Cancel		
56				Text	Search	
57				Font		
58				Height	21	
59				Left	16	
60				Top	24	
61				Width	585	

Рисунок 6.19 – Фрагмент заповненої структури представлення даних блоку CF_1, який містить елементи CD_1^1 «Button1» і CD_2^1 «GroupBox1»

У залежності від вибраної мови і середовища розробки адитивного кібер-дизайну, буде змінюватися назва і призначення елементів в блоках $MP_1(mp_1^1,...,mp_t^1)$ і $ME_1(me_1^1,...,me_h^1)$, тому це твердження правомірно для елементів блоків $CD_1^1,...,CD_f^1$. Але при цьому структура представлення даних для створення прототипу графічного інтерфейсу адитивного кібер-дизайну буде однакова за умови, що будуть використовуватися об'єктно-орієнтовані мови програмування [227].

6.7 Експериментальні дослідження і аналіз отриманих результатів

Для проведення експериментальних досліджень і перевірки правильності запропонованих математичних моделей і методів, в даній дисертаційній роботі, на базі розробленої «Система розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом», було прийнято рішення провести розробку фрагментів кібер-фізичних виробничих систем, відповідно вимог, поставлених у договорах на проведення науково-дослідних робіт за

замовленням підприємств: № 15-03 04 ТОВ «ЗЕО «Сокіл»» (Автоматика) «Розробка програми візуалізації та електроавтоматики зерносушарки»; № 14-04 ТОВ «ЗЕО «Сокіл»» (Автоматика) «Розробка комплексної системи оперативно-диспетчерського керування виробничим підприємством ТОВ «ЗЕО «Сокіл» і порівняти отримані результати з існуючими стандартними підходами до розробок кібер-фізичних виробничих систем або їх фрагментами [228].

Базуючись на вимогах, висунутих замовником до розроблюваної «Комплексної системи оперативно-диспетчерського контрольного управління виробництвом підприємства ТОВ «ЗЕО «Сокіл» (OSCEM), проведемо визначення «головної мети» (Aim_i) і завдань ($Task_j$) розробки CPPS, які представлені в таблиці 6.1.

Проводячи аналіз ми можемо декомпозувати Aim_i на $Task_j$ за рівнями: мультисистеми ($MS_0^{"}$), підсистеми (Sub_S_k) і групи ($G(AEofS)_j$), при цьому атомарним елементом $(AEofS)_j$ для $G(AEofS)_j$ буде елементарний графічний елемент і / або подія, яка викликається користувачем як реакція кіберскладової на якусь дію або потік вхідної / вихідної інформації.

За результатами декомпозиції Aim_i, які представлені таблиці 6.1, можна на початковому етапі сформувати аналітично-логічну структуру CPPS, яка визначає основні функції і властивості, що дозволить визначити в майбутньому кількість візуальних графічних об'єктів ($Form_1^{master}$, $Form_2^{slave}$, елементів CF_1) і зв'язок між ними (рис. 6.19).

На наступному етапі розробки «OSCEM» потрібно визначити необхідний і достатній набір $AEofS_j$ для кожної $G(AEofS)_j$, який описує Sub_S_k і дозволяє досягти $Task_j$ у відповідності до вимог ТЗ замовника. Для цього розробник, базуючись на аналітично-логічній структурі (рис. 6.20), проводить опис всіх $AEofS_j$ та їх взаємозв'язки. Наведемо як приклад фрагмент вибору $AEofS_j$ і $Task_j$ для = «Довідники», Sub_S_k = «Довідник КТД» для $G(AEofS)_j$ = «Пристрій 1».

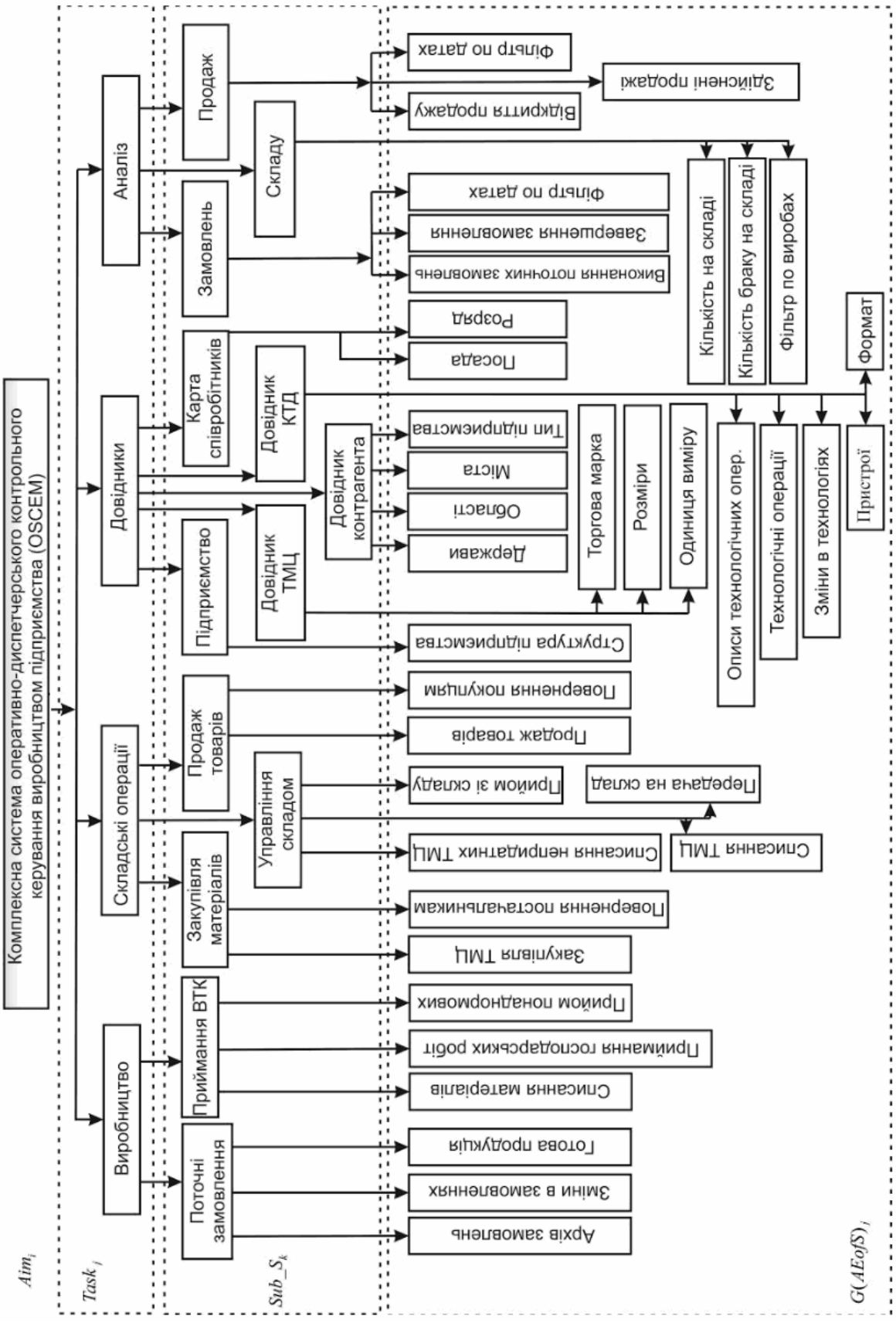


Рисунок 6.20 – Аналітично-логічна структура зв'язків «OSCEM»

Таблиця 6.1 – Визначення Aim_i та $Task_j$ на початковому етапі процесу керування розробкою кібернетичної складової «OSCEM»

Aim_i	$Task_j$	Sub_S_k	$G(AEofS)_j$
Комплексна системи оперативно-диспетчерського контрольного управління виробництвом підприємства ТОВ «ЗЕО «Сокіл» (OSCEM)	Виробництво	Поточні замовлення	Архів замовлень
			Зміни в замовленнях
			Готова продукція
		Приймання ВТК	Списання матеріалів
			Приймання господарських робіт
			Прийом понаднормових
	Складські операції	Закупівля матеріалів	Закупівля ТМЦ
			Повернення постачальникам
		Продаж товарів	Продаж товарів
			Повернення покупцям
		Управління складом	Передача на склад
			Прийом зі складу
			Списання ТМЦ
			Списання непридатних ТМЦ
	Довідники	Підприємство	Структура підприємства
		Карта співробітників	Посада
			Розряд
		Довідник ТМЦ	Торгова марка
			Розміри
			Одиниця виміру
		Довідник КТД	Формат
			Пристрої
			Технологічні операції
			Описи технологічних операцій
			Зміни в технології
		Довідник контрагента	Держава
			Область
			Місто
			Тип підприємства

Продовження табл. 6.1

Aim_i	$Task_j$	Sub_S_k	$G(AEofS)_j$
	Аналіз	Замовлення	Виконання поточних замовлень
			Завершення замовлення
			Фільтр по датах
		Продаж	Здійснені продажі
			Відкриті продажі
			Фільтр по датах
		Склад	Кількість на складі
			Кількість браку на складі
			Фільтр по виробам

Ґрунтуючись на результатах отриманих в табл. 6.1 і аналітично-логічної структури зв'язків «OSCEM», представленої на рис. 6.20, для $Task_j$ = «Довідники», Sub_S_k = «Довідник КТД» для $G(AEofS)_j$ = «Пристрій 1», а саме фрагмент вибору $AEofS_j$ для $G(AEofS)_j$ = "Пристрій 1» → «деталь 1» → «Конструкторська документація» та «Технологія», визначимо що логічною буде наступна ієрархія HMI (Human machine interface) зв’язків для «OSCEM» (рис. 6.21).

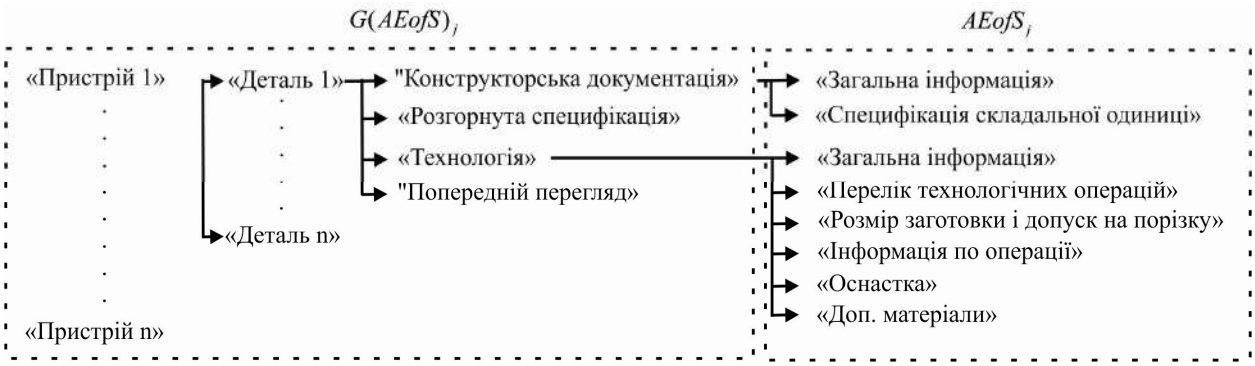


Рисунок 6.21 - Фрагмент вибору $AEofS_j$ для $G(AEofS)_j$=«Пристрій 1»→«Деталь 1» → «Конструкторська документація» та «Технологія»

У відповідності до вимог ТЗ замовника необхідно визначити «назву параметрів» і «значення» які входять в кожен $AEofS_j$. У таблиці 6.2 представлені вибрані розробником параметри і тип значення для $G(AEofS)_j$ = «Пристрій 1» → «деталь 1» → $AEofS_j$ = «Технологія», які були затверджені замовником і дозволяють досягти мети та вимог ТЗ на розроблювану «Систему оперативно-диспетчерського керування виробничим підприємством ТОВ «ЗЕО «Сокол».

Таблиця 6.2 – Моделі взаємодії $G(AEofS)_j$ = «Пристрій 1» → «Деталь 1» → $AEofS_j$ = «Технологія»

$AEofS_j$	Параметри	Тип значення
1	2	3
«Загальна інформація»	Назва замовлення	Текстове
	Назва деталі / збірки	Текстово-числове
	Основний матеріал	Числове
	Кількість деталей (шт.)	Числове
	Число дет. з 1 загот.	Числове
	Вага деталі (кг)	Числове
	Вага заготовки (кг)	Числове
«Список технологічних операцій»	Ділянка	Текстове
	Операція	Текстове
	Устаткування	Текстове
	Розробка	Числове
	Кількість робіт	Числове
	Партія	Логічне
	Одиниця норми	Числове
	Час підготовчий/заключний, $T_{ПЗ}$	Числове
	Час штучний, $T_{шт}$	Числове
«Розмір заготовки та допуск на різання»	Довжина (мм)	Числове
	Ширина (мм)	Числове
«Інформація по операції»	Дата	Дата
«Оснащення»	Найменування	Текстове
	Кількість	Числове

Продовження таблиці 6.2.

1	2	3
«Додаткові матеріали»	Найменування	Текстове
	Одиниця вимірювання	Числове
	Кількість	Числове

Наступним кроком, в процесі керування розробкою CPPS, йде розробка аналітично-логічної структури зв'язків «OSCEM» (рис. 6.19) у вигляді моделі взаємодії (табл. 6.2) основних вікон HMI. Варто зауважити, що в даному випадку необхідно використовувати вимоги до промислового дизайну:

– зручність і простота доступу до основних функцій і форм даних (розміщення елементів управління, змісту, оформлення виведених повідомлень і формат введення);

– мінімізація кроків доступу до внутрішнього функціоналу системи, що розробляється, за допомогою призначеного для користувача інтерфейсу.

Ґрунтуючись на перерахованих вище вимогах було прийнято рішення реалізувати HMI з використанням головного вікна додатку ($Form_{ASUP}^{master}$) та допоміжних вікон ($Form_{techn}^{directoty\_CAD},...,Form_{TMC}^{ASUP}$), виклик яких буде проводитися через графічні елементи $CD_1^{ASUP},...,CD_x^{ASUP}$, розташованих на $Form_{ASUP}^{master}$ за подією ініційованою користувачем або системою. Для зручності подання структури зв'язків «OSCEM» між головним вікном $Form_{ASUP}^{master}$ і допоміжними вікнами $Form_{techn}^{directory\_CAD},...,Form_{TMC}^{ASUP}$ розробник будує графічну модель зв'язків Windows Form, на базі удосконаленої методології Константайна (табл. 4.1–4.2). Представимо в табл. 6.3 позначення і опис допоміжних (що викликаються) Windows Form, всі інші елементи будуть реалізовані в рамках головного вікна $Form_{ASUP}^{master}$ на базі елементів GUI (Graphical user interface) – Panel, GroupBox, і т.д.

Таблиця 6.3 – Позначення і опис допоміжних Windows Form з головного вікна $Form_{ASUP}^{master}$.

Позначення	Опис
$Form_{accept}^{ASUP}$	Прийом ВТК
$Form_{purchase}^{ASUP}$	Закупка ТМЦ
$Form_{sale}^{ASUP}$	Продаж товарів
$Form_{transfer}^{ASUP}$	Передача на склад
$Form_{reception}^{ASUP}$	Прийом зі складу
$Form_{write\text{-}off}^{ASUP}$	списання ТМЦ
$Form_{write\text{-}off\_TMC}^{ASUP}$	Списання непридатних ТМЦ
$Form_{TMC}^{ASUP}$	довідник ТМЦ
$Form_{directory\_CAD}^{ASUP}$	довідник КТД
$Form_{contractors}^{ASUP}$	довідник контрагентів
$Form_{techn}^{directory\_CAD}$	Технологія виготовлення деталі

Отже всі $Task_j$, Sub_S_k, $G(AEofS)_j$, які не показані в графічній моделі структури (рис. 6.22), будуть реалізовані у головному вікні GUI «OSCEM» ($Form_{ASUP}^{master}$), відповідно до аналітично-логічної структури зв'язків (рис. 6.20), з використанням наступних графічних елементів, що представлені в табл. 6.4 .

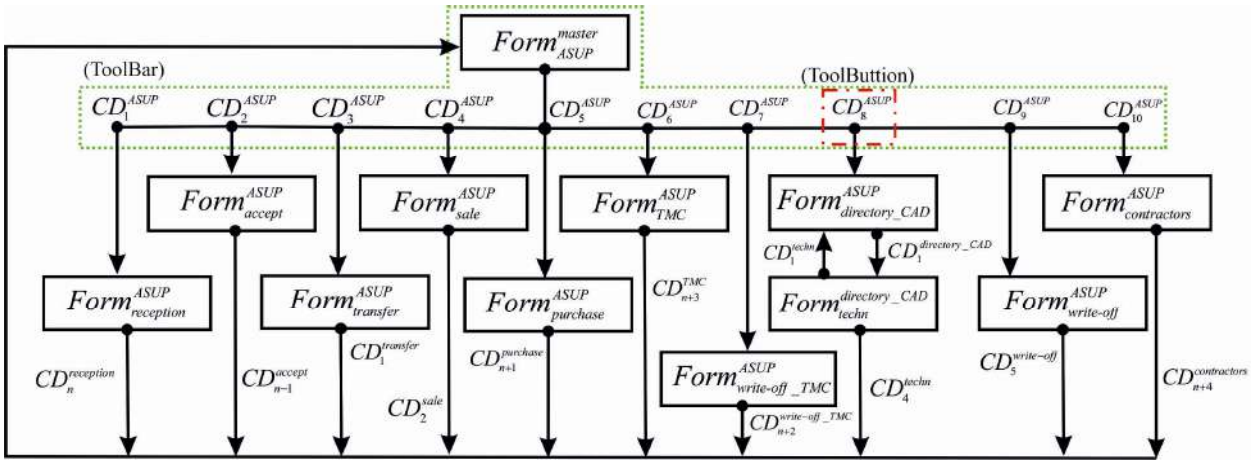


Рисунок 6.22 – Графічна модель структури зв'язків Windows Forms в «OSCEM»

Таблиця 6.4 – Графічна модель структури зв'язків Windows Forms в «OSCEM» $Form_{ASUP}^{master}$.

Позначення	Назва графічного елемента		Опис
1	2		3
$Form_{ASUP}^{master}$	Windows Forms		Головне вікно GUI «OSCEM»
$Task_j$	TabControl		Елемент управління, який використовується для відображення декількох вкладок (TabPage), аналогічних розділювачам в записнику або мітках з набору папок картотечного блоку
Sub_S_k	ToolBar (ToolStrip)		Елемент графічного інтерфейсу користувача, призначений для розміщення на ньому кількох інших елементів (кнопки, меню) швидкого доступу
$G(AEofS)_j$	ToolButtion (ToolStripButtion)		Дочірній елемент ToolBar (ToolStrip) подають у вигляді кнопки з зображенням
$AEofS_j$	Panel	Label	Відображає текст, який не може бути змінений користувачем
		Edit	Призначений для введення призначених для користувача даних і являє собою однорядкове поле
		DBGrid	Відображає дані з БД і дозволяє вносити і зберігати зміни
	GroupBox	DBMemo	Являє собою багатостроковий текстовий редактор, що дозволяє редагувати текст вікна, інформація якого зберігатися в БД
		DBListBox	Дозволяє користувачу вибрати один або кілька елементів із заздалегідь визначеного списку інформації що зберігається в БД
$AEofS_j$	Button		Являє стандартну кнопку, яку користувач може натиснути, щоб виконати дію

Графічний елемент *Panel* – служить контейнером (батьківським компонентом), що об'єднує групу керуючих елементів, введення / виводу та відтворення інформації.

GroupBox – контейнер з рамкою і написом, який об'єднує групу пов'язаних органів управління.

Ґрунтуючись на розробленій аналітично-логічній структурі зв'язків (рис. 6.19) і обраних графічних елементах GUI (табл. 6.4), представимо реалізацію GUI «Основного меню управління» ($Form_{ASUP}^{master} \rightarrow Task_j \rightarrow$ $\rightarrow Sub\_S_k \rightarrow G(AEofS)_j \rightarrow AEofS_j$) у вигляді графа (рис. 6.23).

ImageList – невізуальний компонент, що являє собою набір зображень однакових розмірів, на які можна посилатися за індексами для відображення їх в *ToolButtion*.

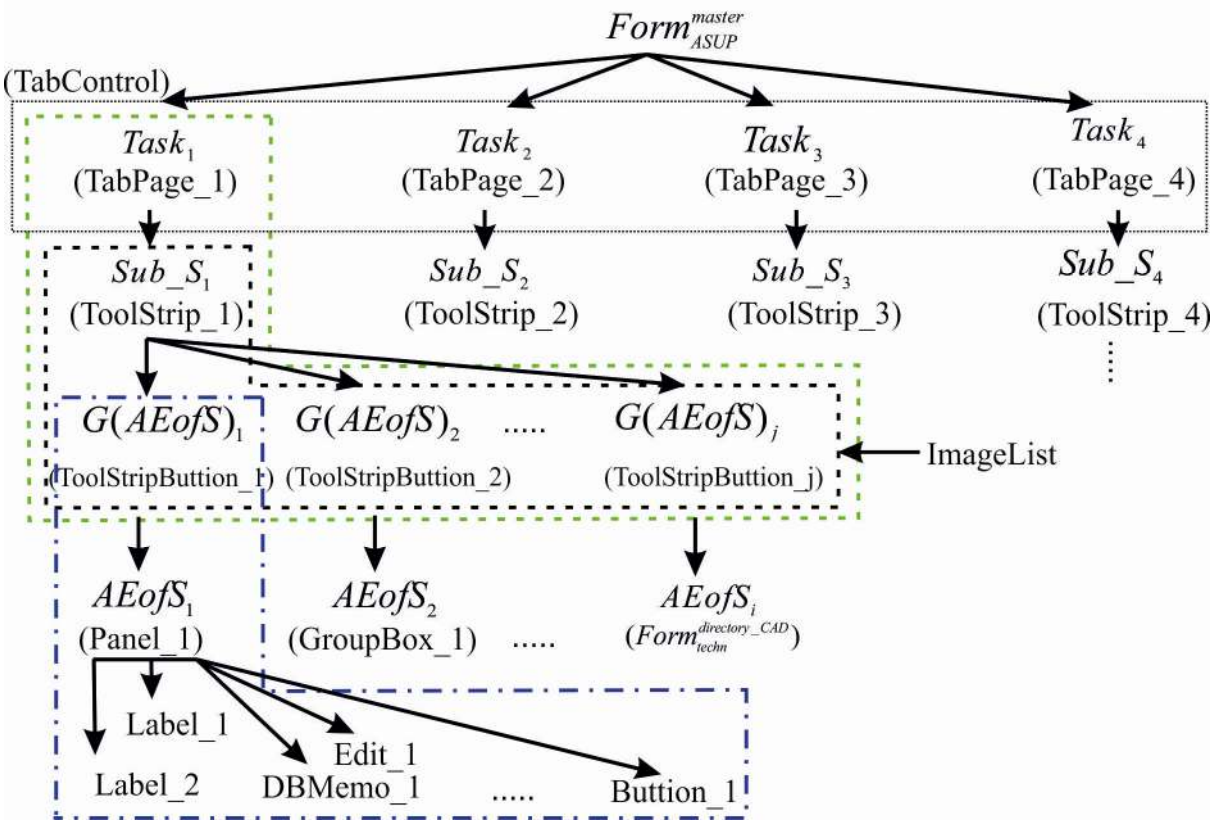


Рисунок 6.23 – Граф реалізації GUI «Основного меню управління» головного вікна «OSCEM» ($Form_{ASUP}^{master}$)

Із запропонованих рішень (рис. 6.20-6.21), а також обраних графічних елементів (табл. 6.4) і графу реалізації GUI головного вікна «OSCEM» (рис. 6.23) можна провести математичний опис структури головного вікна «OSCEM» ($Form_{ASUP}^{master}$). Фрагмент математичного уявлення реалізації Windows Forms ($Form_{ASUP}^{master}$) представлений в (6.1).

На базі MP_{ASUP}^{master} розробник задає значення, які повинні належати до параметрів, що задають всі необхідні властивості головного вікна GUI «OSCEM» та вказує ім'я «*LingusticVariable*» у ME_{ASUS}^{master} (*Open_BD, Close_BD*), що дозволяє задати дію на подію (*Events*) «*Load*» і «*FormClosed*», тобто провести підключення і відключення від БД відповідно.

$$Form_{ASUP}^{master}(Form_{ASUP}PE)[(MP_{ASUP}^{master}) \xrightarrow{mp_3^{ASUP}=Form\_ASUP;mp_4^{ASUP},mp_5^{ASUP},mp_8^{ASUP},} \wedge$$
$$\wedge \xrightarrow{mp_9^{ASUP},mp_{46}^{ASUP}=true;mp_6^{ASUP},mp_{10}^{ASUP},mp_{12}^{ASUP},mp_{13}^{ASUP}=false;mp_{32}^{ASUP}=./img/icon.png;mp_{62}^{ASUP},} \wedge$$
$$\wedge \xrightarrow{mp_{64}^{ASUP}=Default;mp_{14}^{ASUP}=100;mp_{63}^{ASUP}=708,582;mp_{71}^{ASUP}=ASUP;} (PP_{ASUP}^{master})] \quad (6.1)$$
$$[(ME_{ASUP}^{master}) \xrightarrow{me_1^{ASUP}="Open\_BD"me_{12}^{ASUP}=Close\_BD} (EA_{ASUP}^{master})]$$

На наступному кроці необхідно у (6.1) додати математичний опис параметрів GUI елемента *TabControl* (CD_1^{ASUP}), відповідно до графу реалізації GUI «Основного меню управління» головного вікна «OSCEM» (рис. 6.23).

$$CF_{ASUP}[(CD_1^{ASUP})(PC_1^{ASUS}) \xrightarrow{pc_2^1=TabControl\_ASUP,pc_{14}^1,=Top;pc_{19}^1,pc_3^1,pc_{43}^1=false;} \wedge$$
$$\wedge \xrightarrow{pc_{13}^1,pc_{27}^1,pc_{41}^1=true;pc_{20}^1=81,25;pc_{51}^1=680,237;pc_{51}^1=Ilist\_ASUP;} (PP_1^{ASUP})] \quad (6.2)$$

На базі запропонованого прикладу математичного опису (6.1) і (6.2) розробник може задати всі параметри, значення і події, що необхідні для реалізації аналітично-логічної структури зв'язків «OSCEM» (рис. 6.20) для

всіх GUI елементів, представлених в табл. 6.4, у відповідності до розробленого графу реалізації (рис. 6.23).

Розглянемо приклад представлення параметрів, значень і подій $MP_{techn}^{directory\_CAD}, PP_{techn}^{directory\_CAD}, ME_{techn}^{directory\_CAD}, EA_{techn}^{directory\_CAD}$ створення візуального вікна «Технологія» ($Form_{techn}^{directory\_CAD}$) і всіх необхідних графічних об'єктів $CD_1^{techn},\ldots, CD_x^{techn}$ для введення/виведення і відображення необхідної інформації, відповідно табл. 6.2, підрозділу 4.4 та рис. 6.21. Вираз (6.3) описує параметри створення порожнього візуального вікна «Технологія» ($Form_{techn}^{directory\_CAD}$) «OSCEM».

$$
\begin{aligned}
&Form_{techn}^{directory\_CAD}(Form_{techn}PE)[(MP_{techn}^{directory\_CAD}) \xrightarrow{mp_3^{ASUP}=Form\_ASUP;mp_4^{ASUP},} \wedge \\
&\wedge \xrightarrow{mp_5^{ASUP},mp_8^{ASUP},mp_9^{ASUP},mp_{46}^{ASUP}=true;mp_6^{ASUP},mp_{10}^{ASUP},mp_{12}^{ASUP},mp_{13}^{ASUP}=false;mp_{32}^{ASUP}=./img/ico.png;} \wedge \\
&\wedge \xrightarrow{mp_{62}^{ASUP},mp_{64}^{ASUP}=Default;mp_{14}^{ASUP}=100;mp_{63}^{ASUP}=708,582;mp_{71}^{ASUP}=\text{Деталь};} (PP_{techn}^{directory\_CAD})] \qquad (6.3) \\
&[(ME_{techn}^{directory\_CAD}) \xrightarrow{me_1^{techn}="PostBD"} (EA_{techn}^{directory\_CAD})]
\end{aligned}
$$

Елемент $AEofS_j$ «Загальна інформація» є візуальним елементом, який належить $Form_{techn}^{directory\_CAD}$ і містить в собі набір необхідних елементів інтерфейсу ($CD_1^{techn},\ldots, CD_n^{techn}$) для управління даними. Допишемо (6.3) математичним описом елемента CD_1^{techn}:

$$
\begin{aligned}
&CF_{techn}[(CD_1^{techn})(PC_1^{techn}) \xrightarrow{pc_2^1,pc_6^1,pc_{12}^1,pc_{14}^1,pc_{19}^1=alNone;pc_3^1,pc_{43}^1,pc_{53}^1=false;pc_7^1=Close;pc_{10}^1=crDefault;} \wedge \\
&\wedge \xrightarrow{pc_{13}^1,pc_{27}^1,pc_{41}^1,pc_{42}^1,pc_{46}^1=-1;pc_{20}^1,pc_{35}^1,pc_{36}^1,pc_{37}^1,pc_{38}^1,pc_{39}^1,pc_{48}^1,pc_{51}^1=false;pc_{51}^1=true;pc_{15}^1=crDrag;pc_{16}^1=dkDrag;} \wedge \qquad (6.4) \\
&\wedge \xrightarrow{pc_{17}^1=dmManual;pc_{44}^1='\text{Копировать}';pc_{50}^1=328;pc_{52}^1=75;pc_3^1=button\_copy} (PP_1^{techn}) \wedge \\
&\wedge (CE_1^{techn}) \xrightarrow{ce_6^1="Copy\_to\_clipboard"} (EA_1^{techn})
\end{aligned}
$$

За допомогою (6.4) описуються параметри (візуалізація) і події (реакція графічного об'єкта на взаємодію з ним користувача) на одиночне натискання елемента інтерфейсу *Booton*, який ініціалізує «Лінгвістичну

змінну» – *Copy_to_clipboard*, на ім'я якої в базі знань (БЗ) проводитися пошук фрагмента коду для реалізації функцій копіювання даних в буфер обміну даними.

Таким чином можемо представити структуру будь-якого візуального вікна і всіх необхідних графічних елементів для реалізації повнофункціонального інтерфейсу розроблюваної CPPS.

Для спрощення маніпуляцій з великими обсягами записів і зв'язків всіх GUI елементів, вся логіка і послідовність побудови прикладу математичних описів (6.1)–(6.4), в тому числі параметрів, структури і подій графічних елементів інтерфейсу, приховані від розробника і реалізовані на рівні ядра розробленої «Системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом». Тому (6.1)–(6.2) можна уявити у вигляді наступного запису, на базі математичної моделі, представленої нижче.

<Form_ASUP>

*{?** Створення головної форми із заданими параметрами та подіями на створення та закриття форми **?*

[Name=Form_ASUP; ControlBox, MaximizeBox, MinimizeBox, ShowIcon, ShowInTaskbar =true; TopMost, Locked, Localizable, AutoSize = false; Icon=../img/icon.png; Cursor, Language = Default;Opacity=100; Size=708,515; Text= ASUP] [Load= Open_BD; FormClosed=Close_BD]

}

 # TabControl_ASUP

*{?** Створення GUI TabControl із заданими параметрами з підключенням невізуального компоненту ImageList **?*

[Name=TabControl_ASUP; Alignment=top; AllowDrop, Locked, Multiline=false; Enabled, TabStop, Visible=true; ItemSize=81,25; Size=680,237, ImageList=Ilist_ASUP]}

...

#TabPage_directory

*?** Опис параметрів закладинок «Довідники» **?*

{[ImegeIndex=12;Text=Довідники;Size=192,67;Locked=false]}

 # ToolStrip _CAD

*?** Пояснення **?*

{[ ]}

 # ToolStripButtion _dir_CAD

*?** Пояснення **?*

{[ ] [ ]]}

}

...

#PictureBox_logo

*{?** Пояснення **?*

[Name=PicBox_ OSCEM; BorderStyle=None, ImageLocation= ../img/logo.jpg; Location=10,98; Size= 200,200]

}

....

}

</ Form_ASUP>

Результати компіляції згенерованого програмного коду в Visual Studio 2019, на базі розробленої «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом» головного вікна $Form_{ASUP}^{master}$ «OSCEM», представлено на рис. 6.23.

Як можна бачити з рис. 6.24 дана реалізація HMI, на базі обраних GUI елементів, дозволяє реалізувати зручність доступу до необхідного функціоналу, у відповідності з аналітично-логічною структурою зв'язків «OSCEM» (рис. 6.20). На рис. 6.25 показано використання GUI елементів з табл. 6.4 для реалізації $Task_j$, Sub_S_k, $G(AEofS)_j$ рівнів доступу до функціоналу «OSCEM».

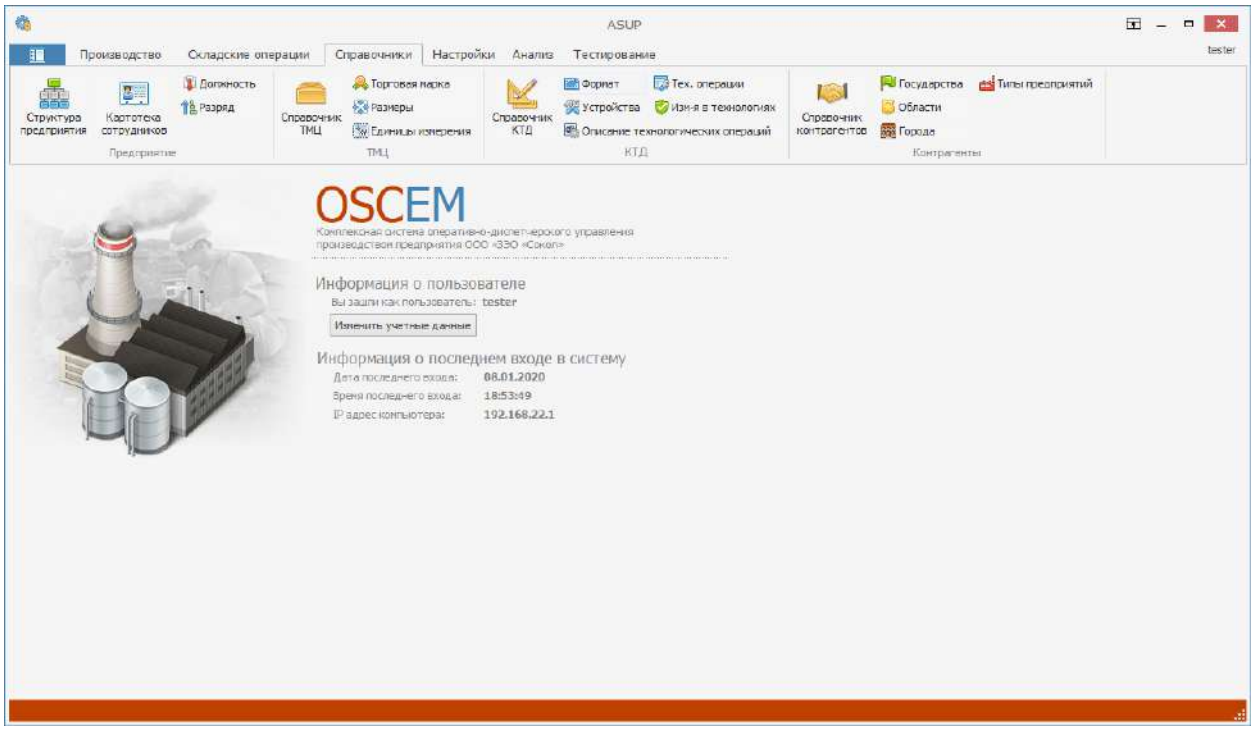


Рисунок 6.24 – Скомпільований результат генерації головного вікна $Form_{ASUP}^{master}$ «OSCEM»

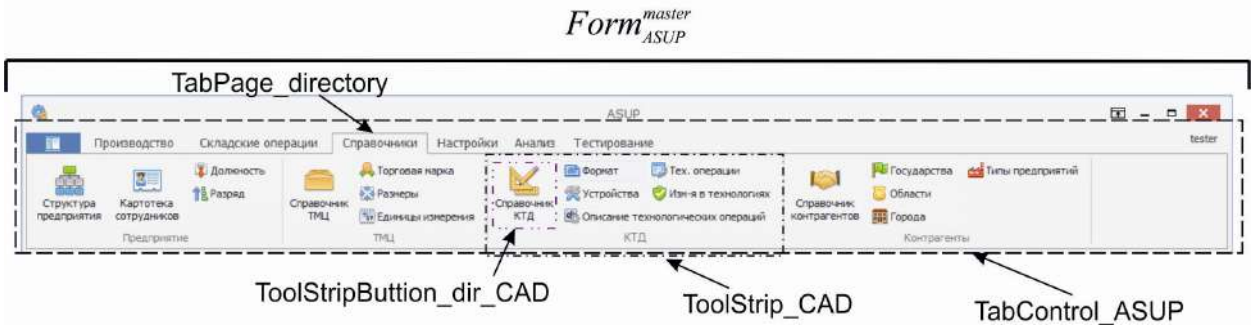


Рисунок 6.25 – Фрагмент реалізації $Task_j$, Sub_S_k, $G(AEofS)_j$ рівнів доступу до функціоналу «OSCEM»

Базуючись на отриманих результатах (рис. 6.23) і вимогах ТЗ до функціональності, дана реалізація GUI «Основного меню управління» головного вікна «OSCEM» ($Form_{ASUP}^{master}$) буде незмінною і закріпленою, а всі інші елементи $AEofS_j$ будуть розміщені на *Panel* в $Form_{ASUP}^{master}$, або викликатися окремими Windows Form, відповідно до графічної моделі структури зв'язків (рис. 6.22).

Наведемо приклади реалізації HMI інтерфейсу «OSCEM» «Довідники»→«Довідник КТД», на базі GUI елементу *Panel*, на якому розміщені всі візуальні елементи в рамках $Form_{ASUP}^{master}$ (рис. 6.26).

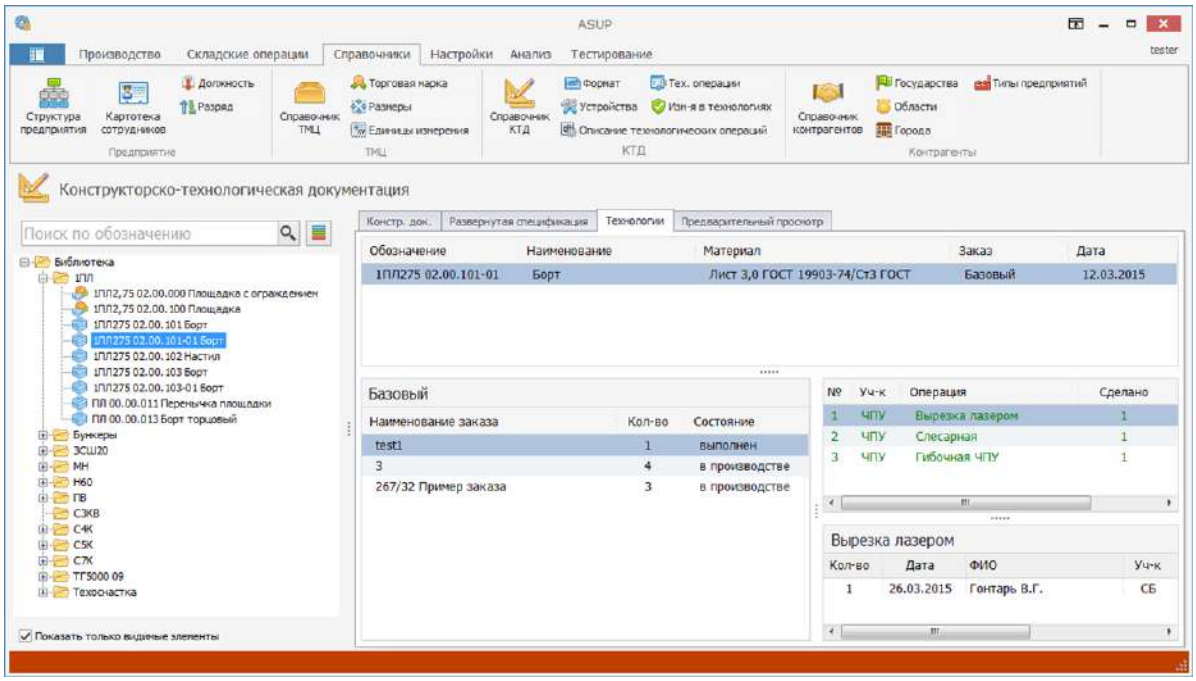


Рисунок 6.26 – Приклад реалізації HMI інтерфейсу «OSCEM» «Довідники»→«Довідник КТД»

Наведемо, як приклад, результат скомпільованого вікна (Windows Form) реалізації інтерфейсу на базі GUI елементів (рис.6.21, табл. 6.2, 6.4) для «Технологія» ($Form_{techn}^{directory\_CAD}$). Для прикладу (рис. 6.27) окремо викликаються вікна з «Довідник КТД» при виборі користувачем *TabControl* «Технологія» та інформації в DBGrid (табл. 6.4).

Ґрунтуючись на отриманих наукових результатах для перевірки правильності прийнятих рішень були проведені дослідження процесу управління розробкою «Комплексної системи оперативно-диспетчерського контрольного управління виробництвом підприємства» (OSCEM) на базі «Система розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом» і класичного методу процесу управління розробкою, які велися паралельно з іншою

групою розробників, в рамках виконання наукової теми № 14-04 за замовленням підприємства ТОВ «ЗЕО «Сокіл» (OSCEM).

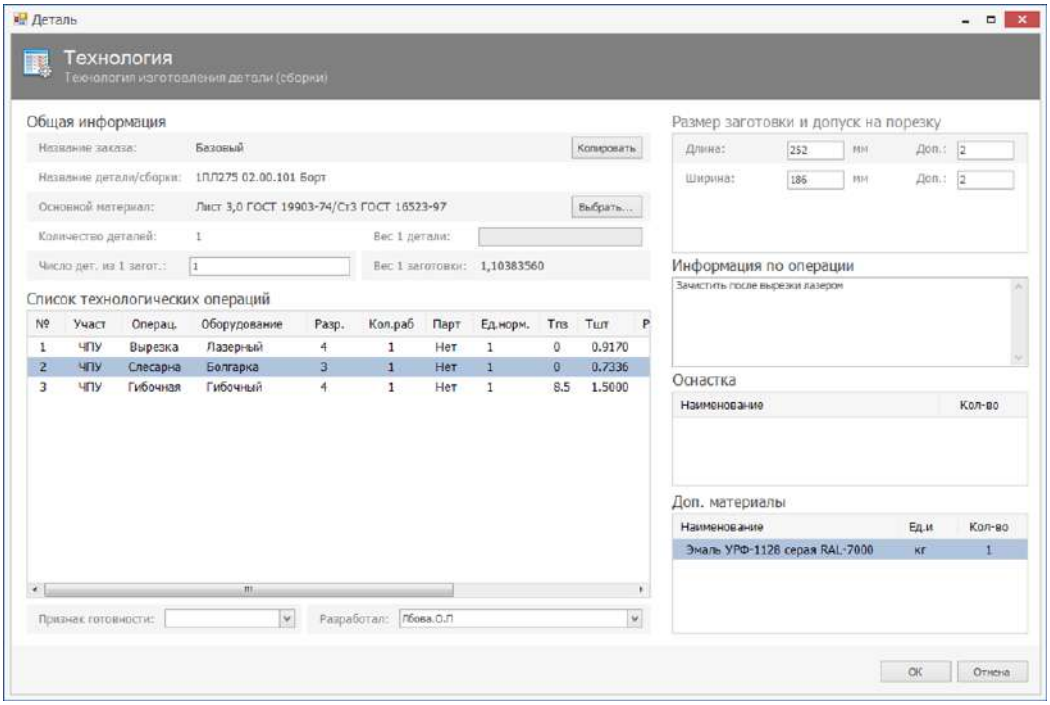


Рисунок 6.27 – Приклад реалізації вікна «Технологія» ($Form_{techn}^{directory\_CAD}$)

Для наочності отриманих результатів проведеного експерименту класичним процесом управління розробкою «OSCEM» була побудована таблиця порівняння (табл. 6.5) за такими параметрами і критеріями:

– етапи, підетапи;

– час початку кожного підетапу (від початку ініціалізації процесу управління розробки);

– тривалість кожного етапу в днях;

– обсяг роботи, в годинах, при умові тривалості робочого дня 8 годин.

Ґрунтуючись на отриманих експериментальних даних при розробці «OSCEM» класичним методом процесу управління (табл. 6.5) була побудована діаграма Ґанта, яку наведено на рис. 6.28.

Таблиця 6.5 – Класичний процес управління розробкою «OSCEM»

Етапи проекту	Підетапи	Час початку, дні	Тривалість, дні	Обсяг, год.
1	2	3	4	5
Ініціалізація	Загальна постановка завдань проєкту	0	15	120
	Визначення вимог	0	10	80
	Визначення специфікації	3	15	120
Планування	Визначення ТЗ	4	40	320
	Постановка мети	15	30	240
	Уточнення результатів	25	34	272
	Визначення складу робіт	32	40	320
	Формування календарного плану	40	47	376
	Проведення оцінки ризиків	5	70	560
Розробка	Визначення конфігурації	40	64	512
	Технічні способи досягнення мети	53	20	160
Реалізація	Створення інтерфейсу	65	40	320
	Написання коду	70	70	560
	Контроль метрик	80	60	480
Тестування	Тестування	120	30	240
	Перевірка на відповідність вимогам ТЗ	140	34	272
	Виправлення зауважень і помилок	170	30	240
Моніторинг і завершення проекту	Впровадження	200	23	184
	Виправлення зауважень і помилок	220	30	240
	Підтримка результатів проєкту	240	40	320

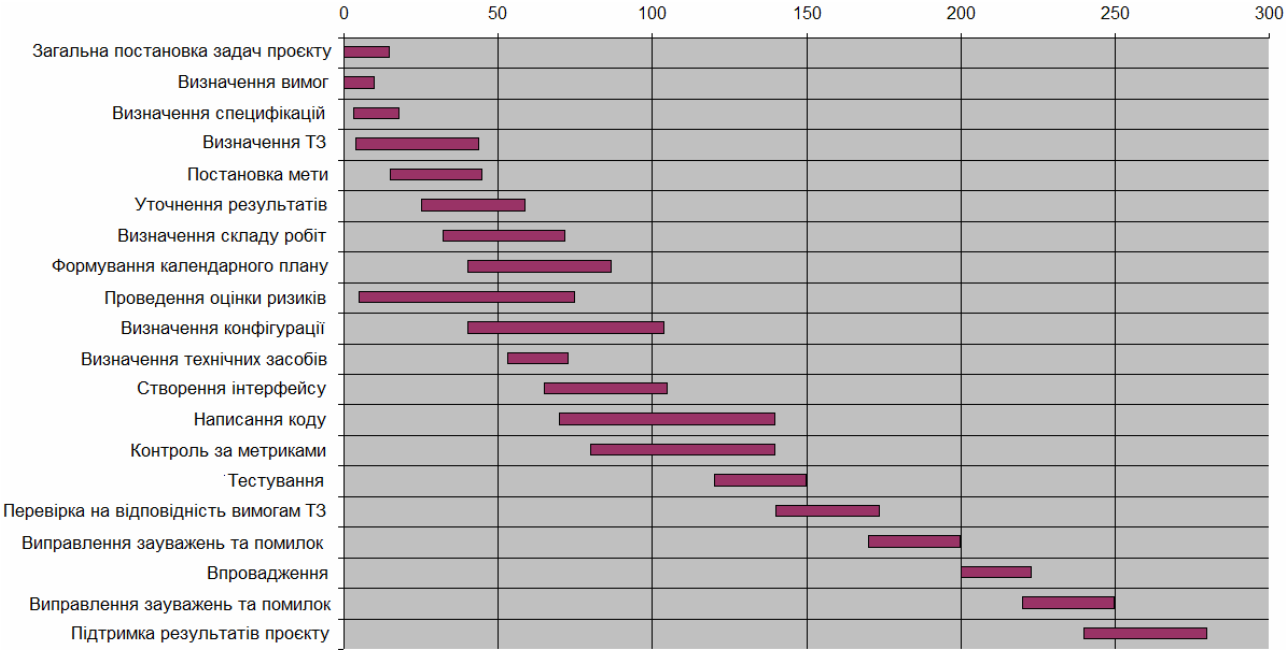


Рисунок 6.28 – Процесу управління розробкою «OSCEM» класичним методом

Відповідно, для зручності подання отриманих результатів, процес управління розробкою «Комплексною системою оперативно-диспетчерського контрольного управління виробництвом підприємства» (OSCEM), на базі запропонованих моделей реалізованих в системі «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом», була побудована табл. 6.6 за такими параметрами і критеріями:

– етапи, підетапи;

– час початку кожного підетапу (від початку ініціалізації процесу управління розробки);

– тривалість кожного етапу в днях;

– обсяг роботи, в годинах, при умові тривалості робочого дня 8 годин.

На базі отриманих експериментальних даних (табл. 6.6) побудована діаграма Ґанта (рис. 6.29) процесу управління розробкою «OSCEM» за допомогою розробленої системи.

Таблиця 6.6 – Процес управління розробкою «OSCEM» на базі «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом»

Етапи проекту	Підетапи	Час початку, дні	Тривалість, дні	Обсяг, год.
1	2	3	4	5
Цільовий	Постановка головної мети	0	15	120
	Визначення подцілей	15	20	160
	Визначення завдань для подцілей	30	31	248
	Математичний опис елементарних завдань	59	50	400
Фізичний	Визначення функцій кожного рівня	108	20	160
	Розробка організаційно-технічної структури	120	8	64
Кібернетичний	Розробка інфологічної структури	128	10	80
	Розробка інформаційної структури	138	14	112
	Розробка алгоритмів керування	149	19	152
Розробка	Розробка опису CPPS на мові моделювання	165	10	80
	Доопрацювання коду і HMI інтерфейсу	173	30	240
	Перевірка досягнення головної мети	200	5	40
Тестування та моніторинг	Тестування	205	15	120
	Впровадження та моніторинг	219	10	80
	Супровід	220	40	320

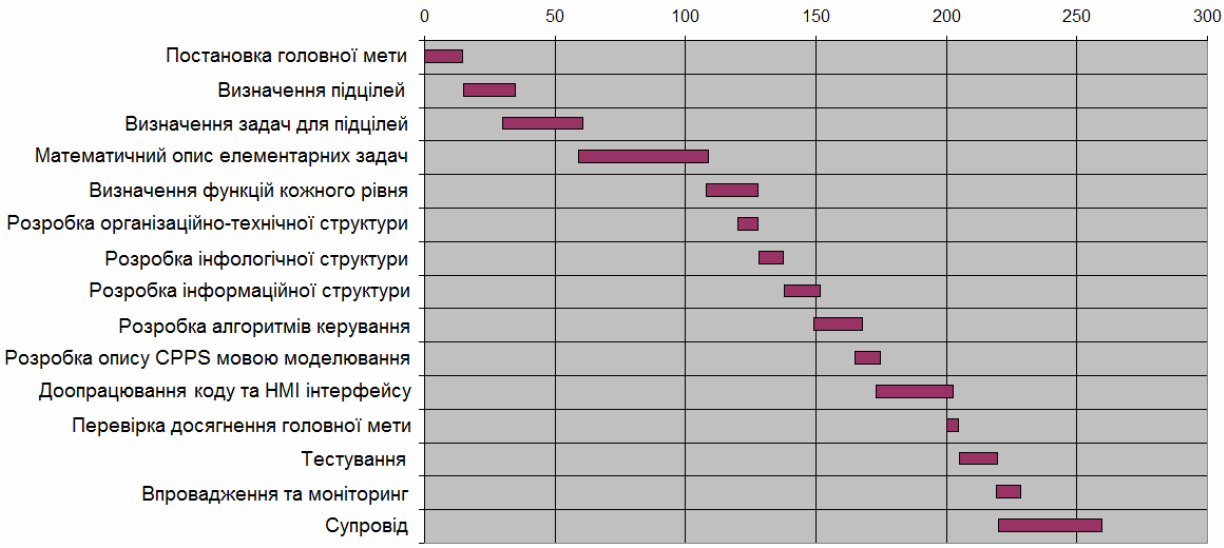


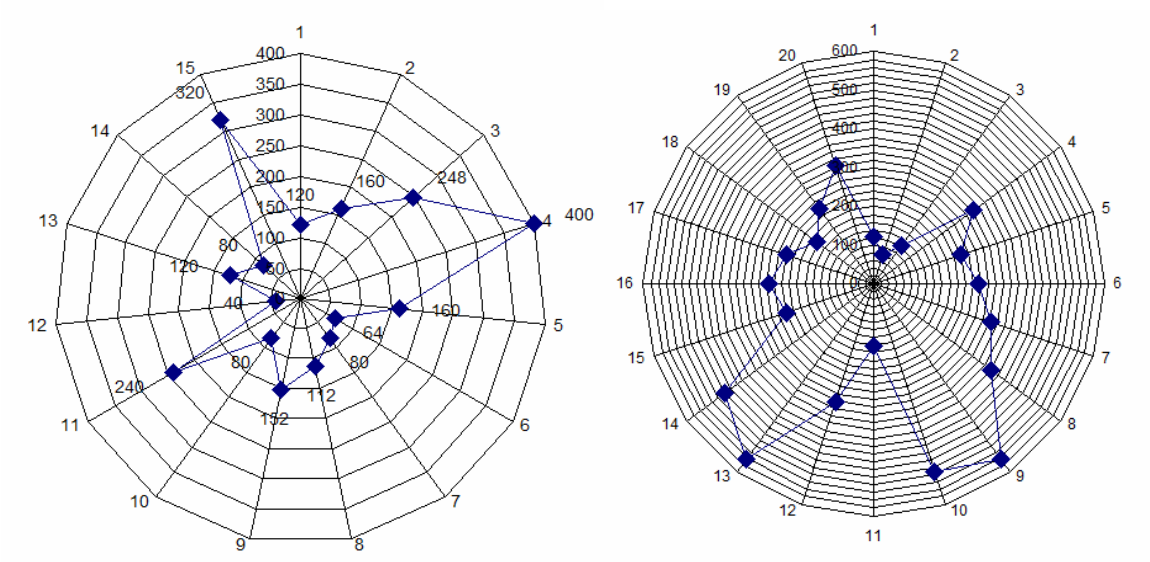
Рисунок 6.29 – Процес управління розробкою «OSCEM» на базі «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом»

Для зручності порівняння отриманих даних, в ході експериментальних досліджень, під час розробки «OSCEM», побудуємо пелюсткову діаграму, в якій позначимо через *1,2,..., n* підетапи розробки та обсяг годин, що витрачаються на кожен з них для розробленого процесу управління (рис. 6.30, а) і класичного (рис. 6.30,б).

Як можна бачити з рис. 6.30 розроблений автоматизований процес управління, реалізований в «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом», має ряд переваг перед класичним методом управління:

– кількість етапів і підетапів в розробленому технологічному процесі управління розробкою CPPS менше ніж в класичному методі;

– розроблена «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом» може досягти максимального обсягу часу, що необхідний для вирішення завдань в підетапі «Математичний опис елементарних завдань» – 400 годин.



а) б)

а) розроблений процес управління розробкою CPPS;

б) класичний метод управління розробкою CPPS

Рисунок 6.30 – Обсяг годин витрачений на кожному підетапі процесу управління розробкою «OSCEM»

Проведемо об'єднання отриманих експериментальних результатів для класичного і розробленого методу управління, шляхом накладення отриманих даних, у вигляді пелюсткової діаграми (рис. 6.31).

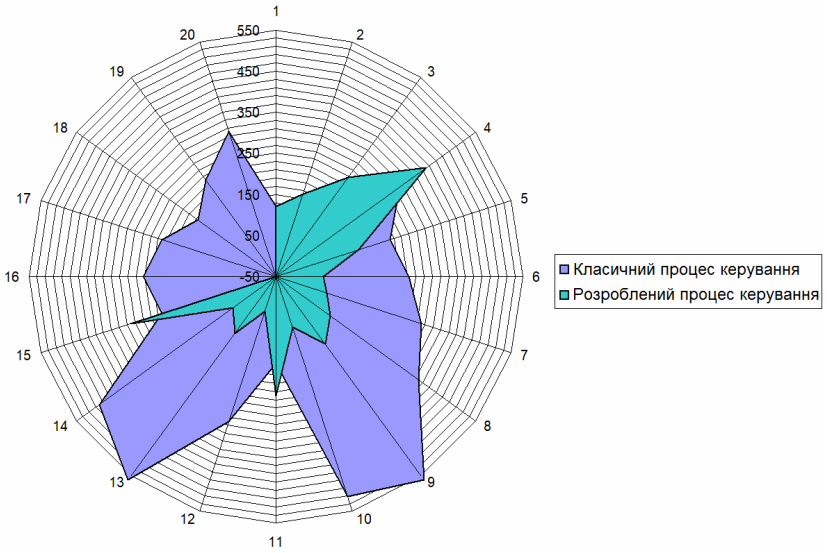


Рисунок 6.31 – Діаграма порівняння витрат обсягу годин при використанні класичного і розробленого процесу управління розробкою «OSCEM»

Також на базі «Системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом» були розроблені наступні рішення для:

– автоматизації керування ТП на виробництві:

1) «Голосове управління роботом РМ-01» (авторське свідоцтво №57666 від 17.12.2014р.) [229];

2) «Програмування та віддалене управління мобільним роботом «Programming robots» (авторське свідоцтво №59439 від 24.04.2015 р.) [230] ;

3) «Автоматизована система проектування технологічного процесу виготовлення акселерометрів «АсСАМ» («АсСАМ») (авторське свідоцтво № 65348 від 16.05.2016 р.) [231, 232];

4) «Модуль автоматизованого проектування технологічних схем складання роботів «Мах-САМ»» (авторське свідоцтво № 74619 від 13.11.2017 р.) [233];

5) «Автоматизація комп'ютерного зору та обробки відеопотоку для мобільних роботів» (авторське свідоцтво № 80306 від 16.07.18 р.) [234] ;

– автоматизовані системи проектування ТП і нормування: «Система нормування «НОРМА» (авторське свідоцтво №57667 від 17.12.2014р.) [221,235];

– автоматизовані системи конструкторської підготовки: «Модуль автоматизованого проектування конструкції роботів «Max-Robotics» (авторське свідоцтво № 74642 від 13.11.2017 р.) [236];

– автоматизовані системи неруйнівного контролю: «Програма для визначення синхронного контролю температурних режимів плат на виробництві «QUAcontrol» (авторське свідоцтво №59980 від 4.06.2015р.) [237];

– система розробки програмного засобу для комп'ютерно-інтегрованої системи технологічної підготовки виробництва: «Автоматизована система проектування програмного забезпечення для корпоративно-інформаційних систем технологічної підготовки виробництва «CAD-Programming Code»

(авторське свідоцтво № 74576 від 09.11.2017 р.) [238];

Другий експеримент, для перевірки адекватності розроблених методів і моделей керування організаційно-технічними об'єктами на базі кібер-фізичних систем, проводився в рамках науково-дослідної роботи за договором №20/80-П від 08.08.2108р. між ТОВ «НВП «Укрінтех» і підприємств ГП «НАЕК «Енергоатом»», ОП «Атоменергомаш» ЗНСОіТ, головною метою якого була модернізація преса гідравлічного ДА2238Б і розробка системи керування на базі кібер-фізичних систем.

Загальний вигляд преса ДА2238Б до модернізації представлений на рисунку 6.32.

Рисунок 6.32 – Загальний вигляд преса ДА2238Б до модернізації

Прес є універсальним обладнанням, основне призначення якого – пресування виробів з пластмас в закритих прес-формах. Може бути використаним для виконання операцій неглибокої витяжки, гнуття і

рихтування металевих виробів. Технічні характеристики преса гідравлічного ДА2238Б:

Розмір столу:

– зліва направо, мм: 1400;

– спереду назад, мм: 1250.

Хід повзуна, мм: 800.

Номінальне зусилля преса, кн.: 6300.

Номінальне зусилля нижнього поршня, кН: 1000.

Хід нижнього поршня, мм: 450.

Швидкість холостого ходу повзуна, мм / с: 125.

Швидкість робочого ходу повзуна, мм / с: 3,5.

Швидкість поворотного ходу повзуна, мм / с : 65.

Швидкість робочого ходу нижнього поршня, мм / с: 15.

Відстань між столом і повзуном, мм: 1600.

Потужність двигуна головного руху, кВт: 30,75.

Габарити верстата ДхШхВ, мм: 4100х2330х5580.

Система управління: оператор.

Маса, кг: 33500.

Рік випуску: 1982.

Система управління преса гідравлічного ДА2238Б представлена на рис. 6.33. Як можна бачити з рисунків 6.32 і 6.33 прес керувався тільки в ручному режимі, контроль за технологічними параметрами виготовлення виробів проводитися оператором у візуальному режимі. Відповідно це не дозволяє контролювати параметри та режими роботи преса гідравлічного ДА2238Б, який в залежності від переходу від одного типу деталей на інший, вимагає тривалого настроювання, що призводить до збільшення часу простою обладнання, а отже знижує ритмічність виробництва і збільшує вартість виробів.

Рисунок 6.33 – Ручна система управління преса гідравлічного ДА2238Б

Тому для вирішення поставленого завдання модернізації преса гідравлічного ДА2238Б, відповідно до запропонованих методів і моделей в 2-3 розділах дисертації, було проведено декомпозицію мети на наступні підцілі:

– розробити схему гідравлічну принципову (фрагмент представлений на рис. 6.34);

– розробити таблицю включень електроапаратів в напівавтоматичному режимі і параметрів гідроустаткування на етапах роботи «Виштовхувач», «Пресування», «Підпресовка»;

– реалізувати 3 режими роботи преса: «Автоматичний», «Напівавтоматичний» і «Ручний»;

– реалізувати 3 режими роботи за швидкістю: «Швидкий режим», «Повільний» і «Режим захисту»;

– реалізувати сенсорне управління пресом з контролем наступних параметрів: мнемосхема (призначена для відображення інформації про

роботу преса, настройки і налагодження) і панель оператора (призначена для відображення: тиску пресування, температуру масла, поточний час, положення верхнього повзуна по датчику переміщення, вихідну позицію повзуна, швидкість опускання, номер деталі зі збереженими настройками ТП).

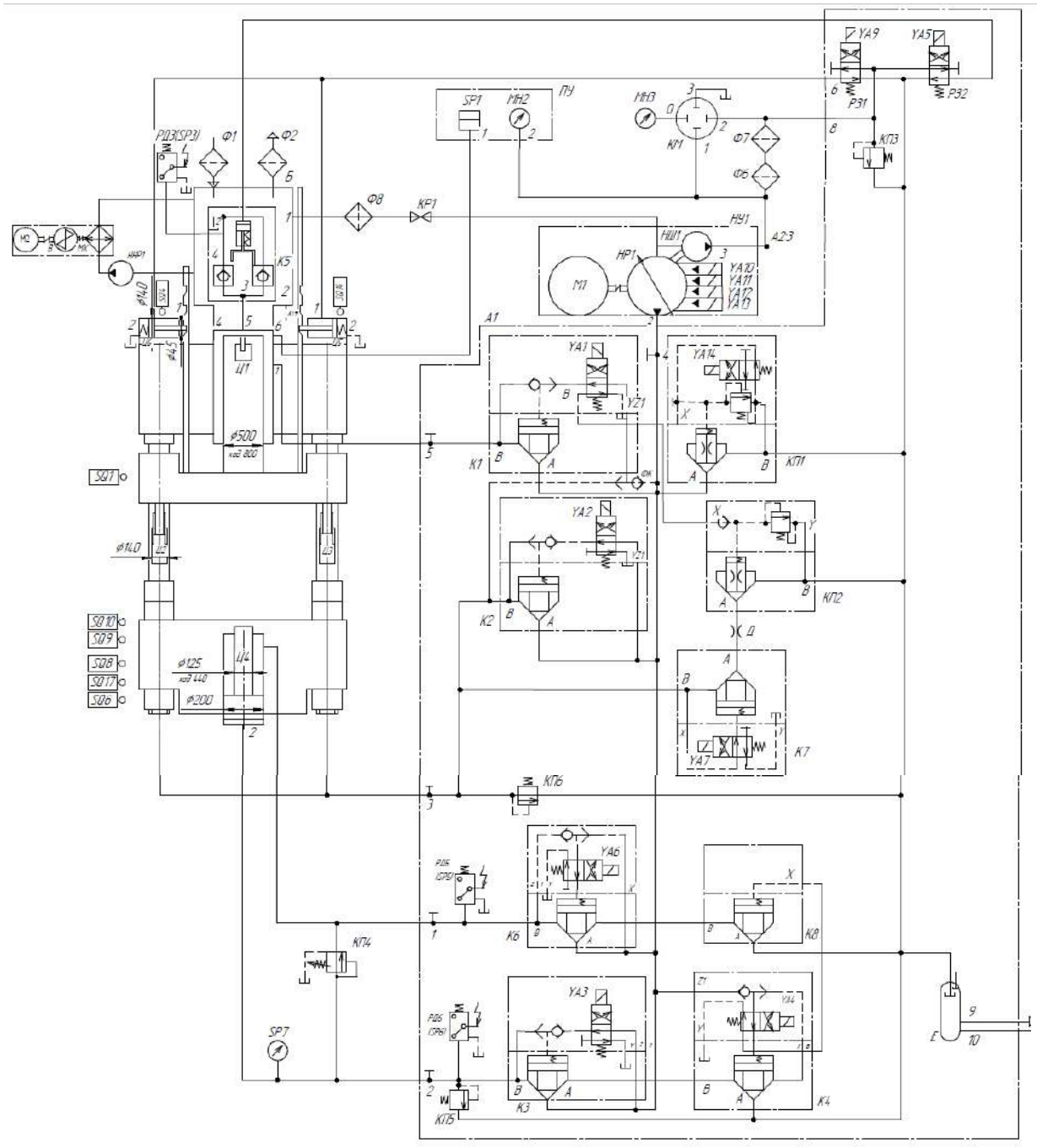


Рисунок 6.34 – Схема гідравлічна принципова преса ДА2238Б

Відповідно до поставлених підцілей реалізації автоматизації управління преса ДА2238Б необхідно розробити таблицю включень

електроапаратів в напівавтоматичному режимі і параметрів гідроустаткування на етапах роботи «Виштовхувач», «Пресування», «Підпресовки». Фрагмент таблиці представлений на рис. 6.35.

Наименование операций		Электромагниты(Y A)															Эл.Двигатель			Командоапараты		Давление Mпа
		1	2	3	4	5	6	7		9	10	11	12	13	14	16	M1		M3	Начало операции	Конец операции	
Исходное положение																	X			SQ1, SQ2		
Отвод фиксаторов						X				X							X			SB	SQ4,SQ14	2.5
Ускоренное смыкание пресса						X		X		X							X			SQ4, SQ14, SP3	SQ2.2	2.5
Подпрессовки	Замедленное смыкание пресса и подъем давления до 10МПа	X			X			X		X	X				X		X			SQ2.2	SP1.1	до10
	Подъем давления свыше10МПа	X			X			X		X		X			X		X			SP1.1	SP1.3	св10
	Выдержка							X		X							X			SP1.3	KT1	св10
	Подпитка	X			X			X		X			X		X		X			SP1.2	SP1.3	св10
	Сброс давления					X				X							X			KT1	SP3	1
	Размыкание замедленное		X			X				X			X		X		X			SP3	SQ2.3	2.6
	Выдержка разомкнутого пресса									X							X			SQ2.3	KT2	2.6
Прессование	Замедленное смыкание пресса и подъем давления до 10МПа	X			X			X		X	X				X		X			KT2,KC	SP1.1	до10
	Подъем давления свыше 10МПа фиксация ползуна	X			X			X		X		X			X		X			SP1.1	SP15,SP16*,SP7*	10.32
	Выдержка							X												SP1.5	KT3	до32
	Подпитка	X			X			X		X			X		X		X			SP1.4	SP1.5	до32
	Пресс1 Сброс давления					X											X			KT3	SP3	1
	Пресс2 Сброс давления															X	X			KT3	SP1.3	до10
	Пресс2 Выдержка							X									X			SP1.3	KT4	св10
	Пресс2 Подпитка	X			X			X					X		X		X			SP1.2	SP1.3	св10
	Пресс2 Сброс давления					X											X			KT4	SP3	1
	Размыкание замедленное		X			X							X		X		X			SP3	KT8	2.6
	Размыкание ускоренное		X			X					X				X		X			KT8	SQ2.1	2.6

Рисунок 6.35 – Фрагмент таблиці включень електроапаратів в напівавтоматичному режимі і параметрів гідроустаткування преса ДА2238Б

За розробленими схемою гідравлічною принциповою (рис. 6.34) і таблицею включень електроапаратів в напівавтоматичному режимі і параметрів гідроустаткування преса ДА2238Б (рис. 6.35), було запропоновано наступне компонування автоматизованого робочого місця (АРМ) оператора управління преса ДА2238Б, яка представлена на рис. 6.36.

АРМ оператора управління преса ДА2238Б складається з 4 основних компонентів: пульт управління містить: *1* – манометр; *2* – дисплей індикації та мнемосхеми преса; *3* – сенсорна панель оператора; *4* – перемикачі та кнопки управління.

Рисунок 6.36 – АРМ оператора керування преса ДА2238Б

Манометр *1* (рис. 6.36) відображає робочий тиск управління гідроклапанами. Дисплей індикації та мнемосхеми преса *2* відображає уявну схему преса, індикацію роботи електророзподільних клапанів, кінцевих вимикачів, поточний стан верхнього повзуна і поршня, кінцевиків дверей, представлений на рисунку 6.37. Сенсорна панель оператора *3* призначена для відображення інформації про роботу преса, настройки і налагодження якої розроблялись з використанням програмного забезпечення «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом», *4* панель перемикачів і кнопок управління пресом ДА2238Б в «Ручному» і «Напівавтоматичному» режимах.

Відповідно до поставлених підцілей і завдань модернізації преса ДА2238Б, було запропоновано реалізувати сенсорне вікно оператора у вигляді 4 програмних форм ($Form_{PO}^{master}$, $Form_{tuning}^{slave}$, $Form_{param\_de}^{slave}$ і $Form_{adjustment}^{slave}$), ґрунтуючись на запропонованому застосуванні інформації з сигнально-кодовою конструкцією.

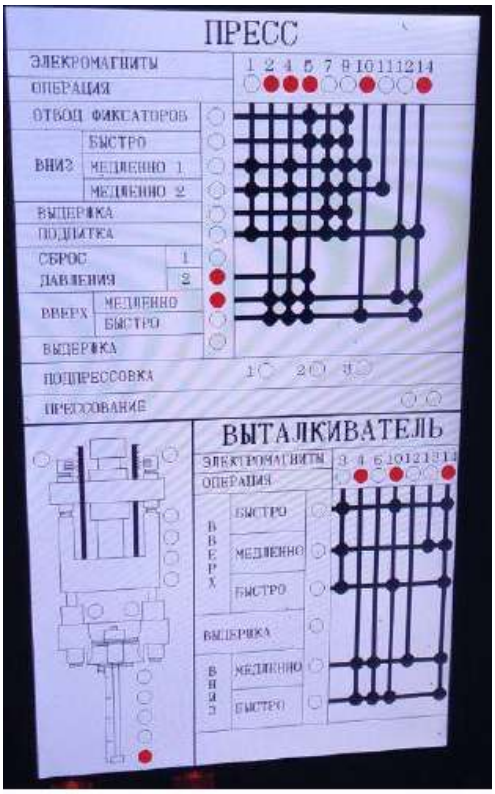


Рисунок 6.37 – Мнемосхема преса ДА2238Б

На наступному етапі розробки сенсорної панелі оператора управління пресом ДА2238Б необхідно визначити розміщення GUI елементів. Для повного уявлення про обсяги необхідної і достатньої інформації оператору, на головній формі $Form_{PO}^{master}$, було прийнято рішення відобразити наступну технологічну інформацію:

– тиск пресування;

– температура масла;

– поточний час і дата;

– положення верхнього повзуна по датчику переміщення;

– вихідну позицію повзуна;

– швидкість опускання повзуна;

– № деталі;

– кнопку переходу в настройки.

Розробка HMI панелі оператора проводилась відповідно розроблених моделей і методів в розділі 4. Варто зауважити, що система управління

пресом ДА2238Б реалізована на базі одноплатного комп'ютера LattePanda 4GB / 64GB, який підтримує Arduino-сумісний співпроцесор ATmega32u4, GPIO контакти для чипу Intel X-Z8300 і ATmega32u4, методи передачі інформації, як провідні 100Mbps Ethernet так і бездротові Wi-Fi, Bluetooth 4.0. Структура модернізованого преса ДА2238Б на базі LattePanda 4/64Gb представлена на рисунку 6.38

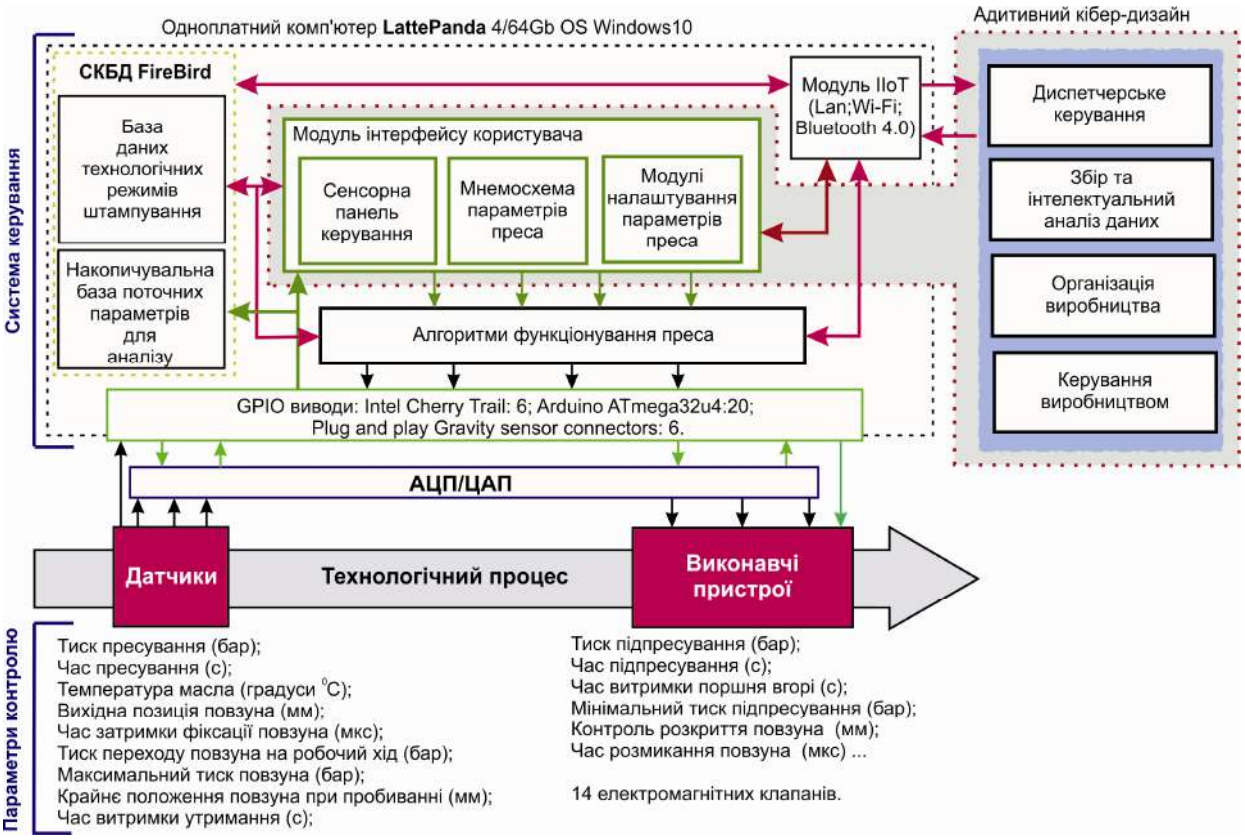


Рисунок 6.38 – Структура модернізованого преса ДА2238Б на базі LattePanda 4/64Gb

Як можна бачити з рисунка 6.38 для модернізації системи управління пресом ДА2238Б, необхідно забезпечити контроль таких параметрів: тиск пресування (бар); час пресування (с); температура масла (^{0}C); вихідна позиція повзуна (мм); час затримки фіксації повзуна (мкс); тиск переходу повзуна на робочий хід (бар); максимальний тиск повзуна (бар); крайнє положення повзуна при пробиванні (мм); час витримки утримання (с); тиск підпресовки (бар); час підпресовки (с); час витримки поршня вгорі (с); мінімальний тиск

підпресовки (бар); контроль розкриття повзуна (мм); час розмикання повзуна (мкс). Для забезпечення роботи преса ДА2238Б використовуються 14 електромагнітних клапани, схема підключення, яких представлена на рис. 6.34, а таблиця включення режимів: «Виштовхувач», «Пресування», «Підпресовка» представлена на рис. 6.35.

Для контролю технологічних параметрів, в даному досліджень були використані датчики:

– датчик температури TER8, який відповідає суворим вимогам гігієнічних стандартів, всі контактуючі частини захищені PEEK-конусом. Точність датчиків TER8 на рівні <0,25°C. Компактне виконання корпусу під вузькі труби;

– датчик тиску MBS 3207, який підтримує інтерфейс CAN open, середовищ з температурою від 0 до 125°C, що повністю відповідає вимогам до температурних режимів роботи преса ДА2238Б, з допустимою температурної компенсацією в діапазоні від 0 до 100°C і діапазон вимірювань від 0-0 , 6 до 0-600 бар;

– лінійний датчик положення WDS-MP/MPW. Серія MPW (водонепроникна) передбачена спеціально для використання у важких умовах навколишнього середовища. Точність + \ – 0,5 мм.

Для розробки кібернетичної складової системи управління пресом ДА2238Б був обраний одноплатний комп'ютер LattePanda 4/64Gb на базі OS Windows 10. Обґрунтуванням цього вибору послужило те, що LattePanda 4/64Gb реалізований на 4-х ядерному процесорі Intel Cherry Trail Z8350 з частотою 1.44ГГц. Об'єм оперативної пам'яті 4Gb DDR3L і з зовнішньою eMMC пам'яттю 64Gb. Так само LattePanda 4/64Gb повністю підтримує: бездротовий інтерфейс Wi-Fi 802.11n 2.4G; Bluetooth 4.0 і Ethernet 100Mbps. На платі реалізовані GPIO входи/виходи, які підтримують: GPIO процесора Intel x 6, GPIO контролера ATmega32U4 x 20, інтерфейсні роз'єми x 6, при цьому розмір плати 88x70 мм і вага 55г.

Розроблений модуль перетворення АЦП/ЦАП дозволив підключити датчики до шини GPIO контролера ATmega32U4 на LattePanda 4/64Gb. На OS Windows 10 було розгорнуто сервер на базі СКБД FireBird, який містить дві бази даних: базу даних технологічних режимів штампування і накопичувальну базу даних поточних параметрів для аналізу, яка дозволяє проводити моніторинг і прогнозування зносу штампа.

Кібернетична система управління пресом ДА2238Б реалізована у вигляді клієнт-серверної архітектури з адитивним кібер-дизайном. На пульті керування пресом ДА2238Б виведені: сенсорна панель управління; мнемосхема параметрів преса і модулів налаштування. Віддалений доступ до управління і моніторингу здійснюється за допомогою реалізації віддаленого підключення до LattePanda 4/64Gb через Wi-Fi мережу із зовнішнім виходом в мережу Internet. При цьому для віддаленого управління і моніторингу параметрами преса ДА2238Б використовується адитивний кібер-дизайн.

На базі запропонованих досліджень був розроблений гнучкий HMI, який дозволив реалізувати доступ в режимі реального часу доступ для: диспетчерського управління, збору та інтелектуального аналізу даних, організації виробництва і управління виробництвом, при цьому відповідність поточної технологічної інформації не має спотворення і достовірна для всіх зарезервованих користувачів. Дане рішення дозволяє адекватно представляти інформацію про стан обладнання і протікання технологічного процесу, оперативного втручання і коригування відхилення і похибок, розрахунок продуктивності і зносу штампа, що дає можливість забезпечити мінімальний простій обладнання при плановому ремонті.Провівши опис структури, на базі запропонованої в 5 розділі мови, були отримані наступні інтерфейси оператора: на рисунку 6.39 представлена конструкція головного вікна управління ($Form_{PO}^{master}$), на рисунку 6.40 представлена конструкція вікна настройки ($Form_{tuning}^{slave}$).

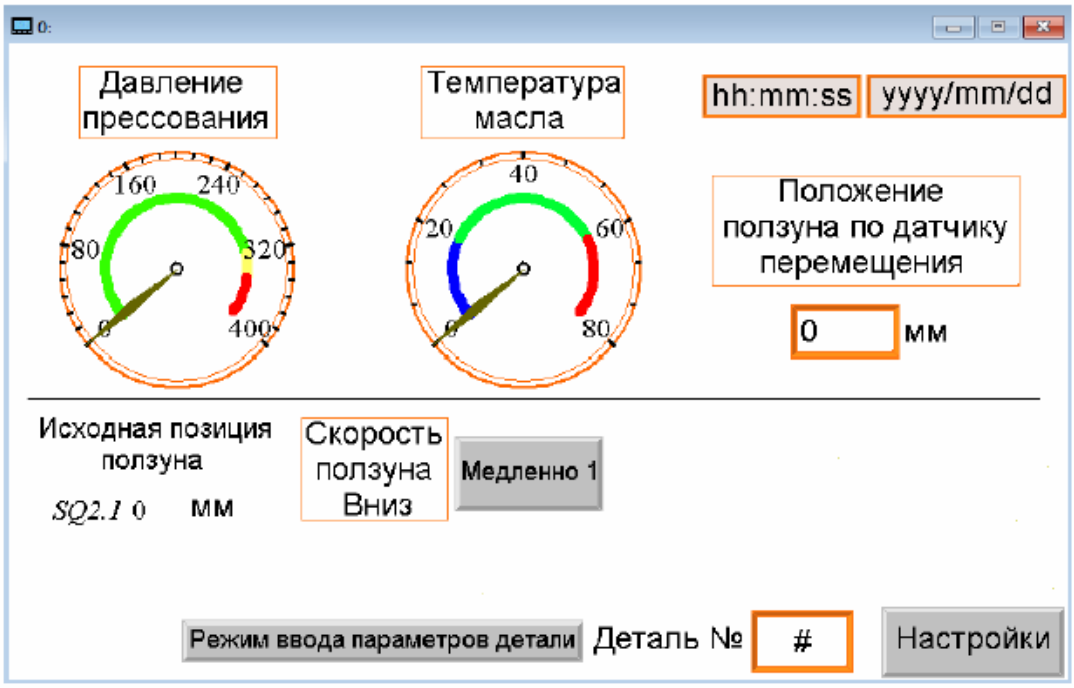


Рисунок 6.39 – Конструкція головного вікна управління ($Form_{PO}^{master}$) пресом ДА2238Б

Вікно ($Form_{tuning}^{slave}$) керування пресом ДА2238Б (рис. 6.40) призначене для налаштування основних параметрів преса. Тут можна змінювати і задати такі технологічні параметри:

– тиск пресування SP1.5, в барах;

– вихідну позицію повзуна SQ2.1 (вибір з чотирьох можливих варіантів);

– перехід на уповільнений хід SQ2.2, в міліметрах (при досягненні даного параметра повзун переходить на уповільнене опускання);

– висоту поршня SA5 (задається вибір від першого до четвертого кінцевого вимикача, при спрацьовуванні якого виштовхувач зупиниться);

– затримку фіксації кінцевих вимикачів поршня, в мілісекундах;

– дату й час;

– вибір номера деталі;

– режим введення параметрів деталі;

– кнопки повернення до Основного вікна і переходу в Налагодження.

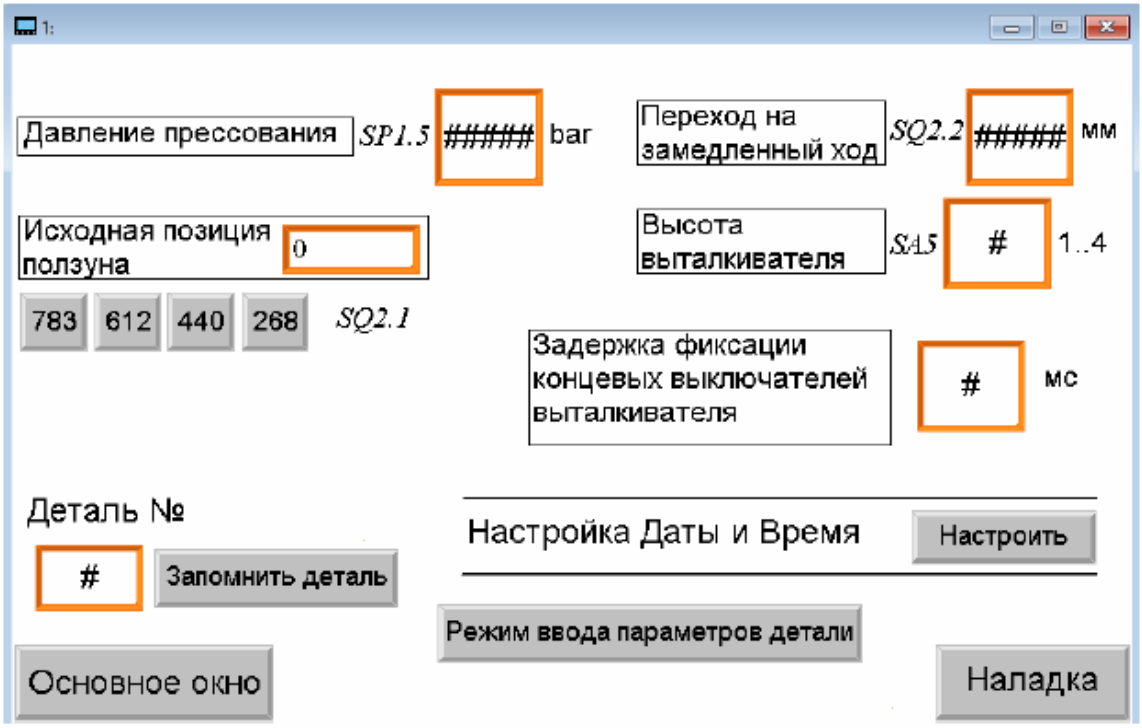


Рисунок 6.40 – Конструкція вікна настройки ($Form_{tuning}^{slave}$)

$Form_{param\_de}^{slave}$ призначена для налагодження параметрів преса ДА2238Б. Для захисту від несанкціонованого доступу замовнику запропонований метод ідентифікації користувача за рахунок установки пароля. Вкладкою можуть користуватися лише сервісні інженери або наладчики з боку замовника.

Конструкція вікна $Form_{param\_de}^{slave}$ (рис. 6.41) містить GUI елементи для реалізації наступних функцій управління параметрами налагодження:

– тиск переходу повзуна на робочий хід SP1.1, в барах (при досягненні зазначеного тиску – повзун з режиму тиску переходить на робочий хід);

– максимальний тиск повзуна SP1.6, в барах (при досягненні заданого значення – залишковий тиск повзуна буде скидатися);

– крайнє положення повзуна при пробиванні SQ2.4, в міліметрах;

– кількість підпресовок SA4, від нуля до трьох;

– витримка при тиску утримання KT4, в секундах;

– час пресування KT3, в секундах;

– перевірка електромагнітних клапанів (дає можливість перевірити гідророзподільник кожного клапана окремо. При натисканні клапан спрацьовує, при повторному натисканні – вимикається);

– вибір деталі;

– режим введення параметрів деталі;

– кнопка «Налаштування» для повернення в попереднє вікно.

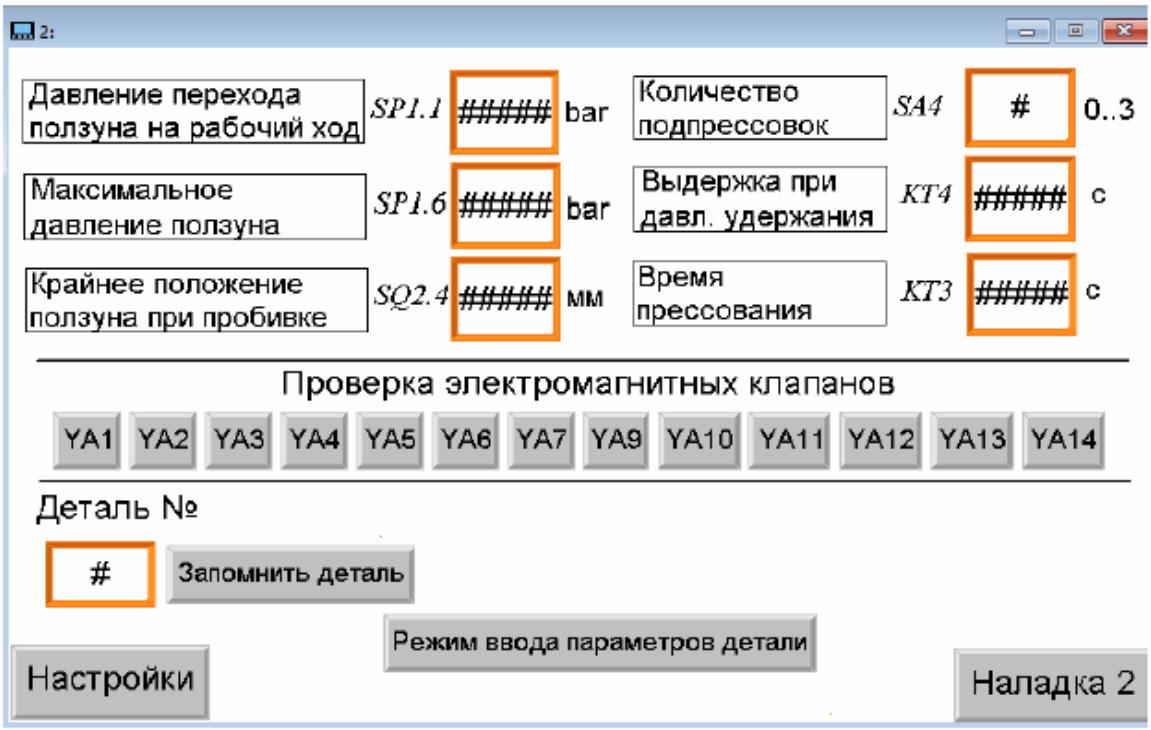


Рисунок 6.41 – Конструкція вікна налагодження ($Form^{slave}_{param\_de}$)

$Form^{slave}_{adjustment}$ призначена для настройки технологічних параметрів підпресовки і реалізує управління наступними параметрами:

– тиск підпресування SP1.3, в барах;

– час підпресування KT1, в секундах;

– час паузи підпресування KT2, в секундах;

– час витримки поршня вгорі KT5, в секундах;

– прискорений хід поршня KT6, в секундах;

– мінімальний тиск підпресування SP1.2, в барах;

– мінімальний тиск пресування SP1.4, в барах;

– контроль розкриття SQ2.3, в міліметрах;

– опускання вниз поршня SA6;

– швидкість розмикання.

Розміщення GUI елементів на HMI формі $Form^{slave}_{adjustment}$ представлене на рисунку 6.42

Рисунок 6.42 – Конструкція вікна налагодження $Form^{slave}_{adjustment}$

Отримані конструкції розроблених форм, на базі «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом», були впровадженні для управління преса ДА2238Б і представлені на рис. 6.39–6.42.

Проводячи аналіз отриманих тимчасових таймингів виконання поставленого завдання з модернізації гідравлічного преса ДА2238Б, на базі розроблених моделей і методів і стандартних підходів (експериментальна розробка системи управління проводилась паралельно), були отримані наступні результати, представлені на рисунку 6.43.

Для зручності візуалізації отриманих даних, весь процес модернізації гідравлічного преса ДА2238Б згрупований в 4 етапи:

– постановка цілей і завдань;

– розробка фізичної складової;

– розробка кібернетичної складової;

– впровадження.

З графіку (рис. 6.43) можна помітити, що застосування розроблених методів і моделей управління організаційно-технічним виробничим об'єктом на базі кібер-фізичних систем, дозволив скоротити витрати часу на етапі розробки фізичної складової, у порівнянні зі стандартним підходом, на 10,71% і на 17,78% на етапі розробки кібернетичної складової.

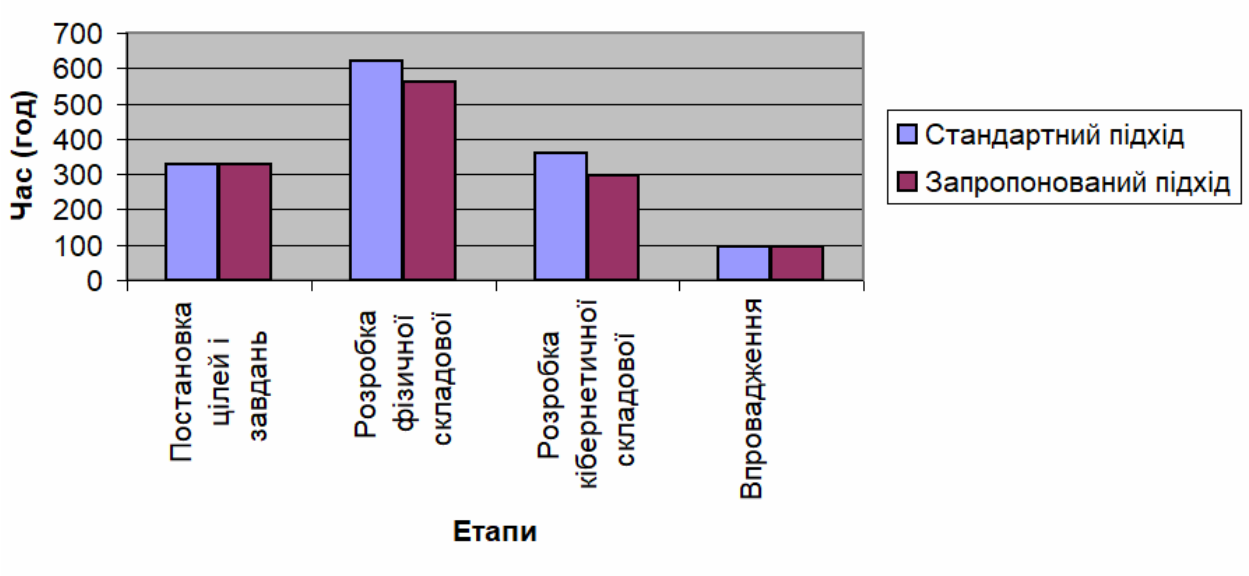


Рисунок 6.43 – Графік витрат часу на модернізацію гідравлічного преса ДА2238Б

Варто зауважити, що результати кібернетичного етапу і частково фізичного, які були реалізовані на базі «Системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом», були впроваджені у виробництво, що підтверджується відповідним актом впровадження ТОВ «НВП«УКРІНТЕХ»» від 23.10.2019 р.

Результати впровадження довели, що запропоновані методи та моделі, дозволили удосконалити процес керування та дозволили підвищити продуктивність на 1,2% та ритмічність 1,8% за місяць.

Висновки до розділу 6

1. Ґрунтуючись на запропонованих моделях, в даній роботі, для апробації та перевірки правильності прийнятих науково-дослідних рішень була розроблена «Система розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом». З огляду на те, що розроблена система дозволяє не тільки автоматизувати процес управління розробкою CPPS з «нуля», а й дає можливість автоматизувати процес управління розробкою кібернетичної складової для модернізації і удосконалення вже існуючих CPPS.

2. Впровадження моделей та методів керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS дозволило:

– об'єднати стратегічну, фізичні та кібернетичні складові керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS в єдиному інформаційному просторі від початку розробки до його впровадження;

– автоматизувати процес перевірки досягнення головної мети (вимог ТЗ) розробки CPPS, який дає можливість вносити зміни і керувати процесом на будь-якому рівні і етапі запропонованої технології;

– автоматизувати процес керування розробки кібернетичної складової (HMI) на базі синтезованих алгоритмів функціонування з використанням GUI елементів об'єктно-орієнтованих мов програмування.

– збільшити гнучкість запропонованої архітектури і технологій автоматизації процесу розробки CPPS, що дозволяє скоротити час розробки кібернетичної складової, шляхом розширення БЗ «*ContainerSolutions*», який містить готовий програмний код.

3. В ході дослідження було проведено ряд порівняльних експериментів, які показали:

– час на розробку «Комплексної системи оперативно-диспетчерського контрольного управління виробництвом підприємства ТОВ «ЗЕО «Сокіл» (OSCEM), на базі «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом», скоротився в 2,5 рази у порівнянні з класичним методом;

– кількість етапів розробки «Комплексної системи оперативно-диспетчерського контрольного управління виробництвом підприємства ТОВ «ЗЕО «Сокіл» (OSCEM) на базі класичного методу на 25% більше ніж у запропонованому;

– обсяг годин витрачених на розробку «Комплексної системи оперативно-диспетчерського контрольного управління виробництвом підприємства ТОВ «ЗЕО «Сокіл» (OSCEM) скоротився в 2,4 рази.

При проведенні модернізації та розробки автоматизованої системи керування гідравлічного преса ДА2238Б були отриманні наступні результати:

– впровадження запропонованих моделей та методів, які реалізовані в «Системі розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом», дозволило суттєво зменшити час розробки кібернетичної складової на 18%, а фізичної складової на 11%;

– запропоновані методи та моделі дозволили удосконалити процес керування, що дало можливість підвищити продуктивність на 1,2% та ритмічність 1,8% на місяць.

Отримані наукові результати можуть бути корисні розробникам систем керування процесами в складних організаційно-технічних виробничих об'єктах на базі CPPS, програмно-технічних модулів у сфері SCADA, ERP, MES систем, відділам АСУ ТП підприємств для модернізації

існуючих CPPS, а також фахівцям в області автоматизації процесів керування розробкою складних CPPS.

Результати наукових і теоретичних досліджень впроваджено в освітній процес, у виробничий процес ряду державних і приватних підприємств, також викладені рішення захищені авторськими свідоцтвами.

Список джерел, які використано у даному розділі, наведено у повному списку використаних джерел [223–238].

ВИСНОВКИ

У дисертаційній роботі, на підставі отриманих результатів, вирішена актуальна науково-прикладна проблема забезпечення ефективної стратегії автоматизації керування складними організаційно-технічними виробничими об'єктами, шляхом реалізації комплексу моделей, методів процесів керування і технології на базі кібер-фізичних виробничих систем.

В результаті проведеного дослідження отримано такі наукові і практичні результати.

1. Проведено критичний аналіз існуючих архітектур, методів та моделей керування процесами в складних організаційно-технічних виробничих об'єктах та виявлено, що найбільш перспективним, в рамках концепції Industry 4.0, є використання кібер-фізичних систем. Встановленно основні протиріччя, що дозволили визначити наукову проблему.

2. Вперше розроблено архітектурно-логічну модель представлення керування процесами в складних організаційно-технічних виробничих об'єктах на базі кібер-фізичних систем, яка базується на науково-обгрунтованих теоріях мультисистем і моносистем та методах формалізованого представлення систем, що дозволило об'єднати стратегічні, фізичні та кібернетичні складові системи у єдиний інформаційний простір.

3. Вперше розроблено логічно узгоджені послідовності взаємопов'язаних методів прийняття рішень на кожному етапі архітектурно-логічної моделі, що дозволило реалізувати технологію «Digital Twins» та самоадаптацію елементів кібер-фізичних виробничих систем керування.

4. Вперше запропоновано технологію розробки кібер-фізичних виробничих систем, яка реалізована на базі теоретико-множинного представлення інформаційних блоків за кожним етапом і рівнем архітектурно-логічної моделі, а також методах їх структуризації, що дозволило реалізувати гнучкість керування процесами в організаційно-

технічних виробничих об'єктах.

5. Удосканалено метод уявлення структурних системних моделей кібер-фізичного керування процесами в організаційно-технічних виробничих об'єктах, що дозволило формалізувати алгоритм функціонування на базі теорії апарату регулярних схем і алгоритмічних алгебр та побудувати структурні і подієві моделі функціонування кібер-фізичних виробничих систем.

6. Удосканалено метод синтезу алгоритмів функціонування кібер-фізичних виробничих систем, що дозволило мінімізувати кількість операторів і спростити структуру системи функціонування організаційно-технічного об'єкту.

7. Удосканалено модель життєвого циклу керування організаційно-технічним об'єктом на базі кібер-фізичних виробничих систем, що дозволило автоматизувати процесс розробки кібернетичної складової на основі синтезованих блоків функціонування.

8. Вперше розроблено модель формалізації кібернетичної складової організаційно-технічного об'єкта у вигляді взаємопов'язаних багаторівневих GUI елементів, що дозволило автоматизувати реалізацію функцій, відповідно до вимог HMI кібер-фізичних виробничих систем.

9. Вперше розроблено математичний опис зв'язків між GUI, як основних елементів HMI, що дозволило реалізувати структурне уявлення кібернетичної складової організаційно-технічного об'єкта за рахунок реалізації подій GUI елементів у вигляді фрагментів програмного коду.

10. Отримала подальший розвиток методологія Константайна, на базі якої запропоновано метод графічного представлення конструкції кібернетичної складової організаційно-технічного об'єкта, що дозволило досягнути редукцію розробки структури.

11. Вперше розроблено синтаксичну і семантичну моделі декларативної мови визначення і опису моделювання кібернетичної складової для керування процесами в складних організаційно-технічних виробничих

об'єктах, на базі кібер-фізичних виробничих систем, за розширеною формою Бекуса-Наура, що дозволило істотно спростити процес керування організаційно-технічним виробничим об'єктом.

12. Практичне значення отриманих теоретичних результатів дисертаційної роботи полягає у розробці методів та моделей керування процесами в складних організаційно-технічних виробничих об’єктах на базі кібер-фізичних систем. Результати дисертаційної роботи впроваджені: у виробничий процес Акціонерного товариства «Мотор Січ» (акт від 15.05.2019р.), Товариства з обмеженою відповідальністю «Науково виробниче підприємство «УКРІНТЕХ»» (акт від 23.10.2019р.); в освітній процес Кременчуцького національного університету імені М. Остроградського (акт від 04.11.2020р.), Харківського національного університету імені В. Н. Каразіна (акт від 29.10.2020р.), Національного університету «Запорізька політехніка» (акт від 23.01.2020р.).

13. Обґрунтованість та достовірність отриманих результатів дисертаційної роботи підтверджуються:

– використанням в роботі теоретично-обгрунтованих та апробованих на практиці методів дослідження;

– впровадженням у виробництво запропонованих методів та моделей автоматизації керування процесами на базі кібер-фізичних виробничих систем у вигляді програмного засобу, які дозволили підвищити продуктивність на 5% та ритмічність 3% (ВАТ «Мотор Січ»), продуктивність на 1,2% та ритмічність 1,8% на місяць (ТОВ «НВП «УКРІНТЕХ»»), що підтверджено відповідними актами впровадження;

– залученням наукової громадськості до апробації наукових результатів на представницьких наукових форумах та їх публікації у фахових наукових виданнях.

14. Наукове значення роботи полягає у подальшому розвитку теорії керування складними організаційно-технічними об'єктами та комплексами на базі кібер-фізичних виробничих систем, в рамках концепцій Industry 4.0 та

Smart Factory.

15. Подальші дослідження рекомендовано продовжити у напрямку узагальнення розроблених методів та моделів кібер-фізічного керування процесами в складних організаційно-технічних об'єктах та комплексах, для вирішення завдання створення безлюдного виробництва, в рамках Industry 4.0 та Smart Factory, для досягення мети «Lean Manufacturing».

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Oztemel, E., Gursev, S. (2020). Literature review of Industry 4.0 and related technologies. *Journal of Intelligent Manufacturing*. Vol. *31*, P.127–182, DOI:10.1007/s10845-018-1433-8.

2. Frank, A. G., Dalenogare, L. S., Ayala, N. F. (2019). Industry 4.0 technologies: Implementation patterns in manufacturing companies. *International Journal of Production Economics*, Vol. *210*(C), P. 15–26, DOI:10.1016/ j.ijpe.2019.01.004.

3. Haseeb, M., Hussain, H. I., Ślusarczyk, B., Jermsittiparsert, K. (2019). Industry 4.0: A solution towards technology challenges of sustainable business performance. *Social Sciences*, Vol. *8*(5), P. 154, DOI:10.3390/ socsci8050154.

4. Alcácer, V., Cruz-Machado, V. (2019). Scanning the industry 4.0: A literature review on technologies for manufacturing systems. *Engineering science and technology, an international journal*, Vol. *22*(3), P. 899–919, DOI: 10.1016/j.jestch.2019.01.006.

5. Lu, Y. (2017). Industry 4.0: A survey on technologies, applications and open research issues. *Journal of industrial information integration*, Vol. *6*, P. 1-10, DOI:10.1016/j.jii.2017.04.005.

6. Haleem, A., & Javaid, M. (2019). Additive manufacturing applications in industry 4.0: a review. *Journal of Industrial Integration and Management*, Vol. *4*(04), P. 1930001, DOI:10.1142/S2424862219300011.

7. Rossit, D. A., Tohmé, F., & Frutos, M. (2019). Industry 4.0: smart scheduling. *International Journal of Production Research*, Vol. *57*(12), P. 3802–3813, DOI:10.1080/00207543.2018.1504248.

8. Culot, G., Nassimbeni, G., Orzes, G., & Sartor, M. (2020). Behind the definition of Industry 4.0: Analysis and open questions. *International Journal of Production Economics*, Vol.*226*(C), DOI:10.1016/j.ijpe.2020.107617.

9. Castelo-Branco, I., Cruz-Jesus, F., Oliveira, T. (2019). Assessing Industry 4.0 readiness in manufacturing: Evidence for the European Union. *Computers in Industry*, Vol. *107*, P. 22–32, DOI:10.1016/ j.compind.2019.01.007.

10. Governement, German Federal. (2014) The new High-Tech Strategy-Innovations for Germany. *Report, Federal Ministry of Education and Research, Berlin* , P.26.

11. Federal Ministry for Economic Affairs and Energy Germany (BMWi). Industrie 4.0 – Checkliste: Kommt Industrie 4.0 für unser Unternehmen in Frage. Berlin. 2017.

12. Хаустова, В. Є., Крамарев, Г. В., Зінченко, В. А. (2019). Інноваційно-технологічне забезпечення модернізації пріоритетних галузей промисловості України. *Бизнес Информ*, (3 (494)), С. 218–228, DOI: 10.32983/2222-4459-2019-3-218-228.

13. Cebeci, U. (2019). The Project Management of Industry 4.0 Strategy for Software Houses. In *Agile Approaches for Successfully Managing and Executing Projects in the Fourth Industrial Revolution* , P. 228–241, DOI:10.4018/978-1-5225-7865-9.ch012.

14. Nowakowski, E., Farwick, M., Trojer, T., Häusler, M., Kessler, J., Breu, R. (2019). An Enterprise Architecture Planning Process for Industry 4.0 Transformations. In *ICEIS*, Vol. *2*, P. 572–579, DOI: 10.5220/0007680005720579.

15. Moeuf, A., Lamouri, S., Pellerin, R., Tamayo-Giraldo, S., Tobon-Valencia, E., Eburdy, R. (2020). Identification of critical success factors, risks and opportunities of Industry 4.0 in SMEs. *International Journal of Production Research*, , Vol. *58*(5), P. 1384–1400, DOI: 10.1080/00207543.2019.1636323.

16. Anderl, R., Picard, A., Wang, Y., Fleischer, J., Dosch, S., Klee, B., Bauer, J. (2015). Guideline Industrie 4.0-Guiding principles for the implementation of Industrie 4.0 in small and medium sized businesses. In *Vdma forum industrie*, Vol. *4*, P. 1–31.

17. Mohammad, U., Cheng, Y. L., Abd Rahman, R., Johar, M. A., Ching, T. K., Roman, D., Kamaruddin, S. (2019). Smart factory reference model for training on Industry 4.0. *Journal of Mechanical Engineering (JMechE)*, Vol *16*(2), P. 129–144, DOI: ir.uitm.edu.my/id/eprint/36433.

18. Both, M., Müller, J., Kämper, B. (2019). Development of Industry 4.0 models and their applicability for BIM. *Revista Romana de Inginerie Civila*, Vol. *10*(3), P. 215–222.

19. Bradac, Z., Marcon, P., Zezulka, F., Arm, J., & Benesl, T. (2019). Digital Twin and AAS in the Industry 4.0 Framework. In *IOP Conference Series: Materials Science and Engineering*, Vol. *618*, No. 1, P. 012001, DOI:10.1088/1757-899X/618/1/012001.

20. Ye, X., & Hong, S. H. (2019). Toward Industry 4.0 components: Insights into and implementation of asset administration shells. *IEEE Industrial Electronics Magazine*, Vol. *13*(1), P. 13–25, DOI:10.1109/MIE.2019.2893397.

21. Євсєєв В.В., Андрусевич А.О., Власенков Д.П. Аналіз концепції Industry 4.0 в технології IIOT. *Технология приборостроения*. 2020, №1. С.64–68.

22. de Lacalle LNL, Posada J. (2019). Special Issue on New Industry 4.0 Advances in Industrial IoT and Visual Computing for Manufacturing Processes. *Applied Sciences*, Vol. *9*(20), P. 4323, DOI:10.3390/app9204323.

23. Madakam, S., & Uchiya, T. (2019). Industrial internet of things (IIoT): principles, processes and protocols. In *The Internet of Things in the Industrial Sector*, P. 35–53, DOI: 10.1007/978-3-030-24892-5_2.

24. Radanliev, P., De Roure, D. C., Nurse, J. R., Montalvo, R. M., & Burnap, P. (2019). Supply Chain Design for the Industrial Internet of Things and the Industry 4.0. *Integration*, Vol. *5*, *Preprints*, P. 2019030123, DOI:10.20944/preprints201903.0123.v1.

25. Osterrieder, P., Budde, L., & Friedli, T. (2020). The smart factory as a key construct of industry 4.0: A systematic literature review. *International Journal of Production Economics*, Vol. *221*, P. 107476, DOI:10.1016/j.ijpe.2019.08.011.

26. Büchi, G., Cugno, M., & Castagnoli, R. (2020). Smart factory performance and Industry 4.0. *Technological Forecasting and Social Change*, Vol. *150*, P. 119790, DOI:10.1016/j.techfore.2019.119790.

27. Kumar, K., Zindani, D., & Davim, J. P. (Eds.). (2019). *Digital Manufacturing and Assembly Systems in Industry 4.0*. CRC Press. P.364. ISBN: 978-1-138-61272-3.

28. Maurer, F., & Schumacher, J. (2019). Evolving towards a smart factory of the future within supply chains: selected cases out of the Alpine space. In *Symposium on Logistics*, P. 197–205.

29. Burns, M., Manganelli, J., Wollman, D., Laurids Boring, R., Gilbert, S., Griffor, E., ... & Smith-Jackson, T. (2018, September). Elaborating the human aspect of the NIST framework for cyber-physical systems. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. *62*, No. 1, P. 450–454, DOI: 10.1177/1541931218621103.

30. Aceto, G., Persico, V., & Pescape, A. (2019). A survey on information and communication technologies for Industry 4.0: state-of-the-art, taxonomies, perspectives, and challenges. *IEEE Communications Surveys & Tutorials*, Vol. *21*(4), P. 3467–3501, DOI:10.1109/COMST.2019.2938259.

31. Faheem, M., Shah, S. B. H., Butt, R. A., Raza, B., Anwar, M., Ashraf, M. W., ... & Gungor, V. C. (2018). Smart grid communication and information technologies in the perspective of Industry 4.0: Opportunities and challenges. *Computer Science Review*, Vol. *30*, P.1–30, DOI: 10.1016/j.cosrev.2018.08.001.

32. Lu, Y. (2017). Industry 4.0: A survey on technologies, applications and open research issues. *Journal of industrial information integration*, Vol. *6*, P. 1–10, DOI:10.1016/j.jii.2017.04.005.

33. Qi, Q., & Tao, F. (2018). Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison. *Ieee Access*, Vol. *6*, P. 3585–3593, DOI:10.1109/ACCESS. 2018.2793265.

34. Kong, X. T., Luo, H., Huang, G. Q., & Yang, X. (2019). Industrial wearable system: the human-centric empowering technology in Industry 4.0. *Journal of Intelligent Manufacturing*, Vol. *30*(8), P. 2853–2869, DOI: 10.1007/s10845-018-1416-9.

35. Tortorella, G. L., Giglio, R., & Van Dun, D. H. (2019). Industry 4.0 adoption as a moderator of the impact of lean production practices on operational performance improvement. *International journal of operations & production management*, Vol. *39*, No. 6/7/8, P. 860–886, DOI:10.1108/IJOPM-01-2019-0005.

36. Cao, Q., Giustozzi, F., Zanni-Merk, C., de Bertrand de Beuvron, F., & Reich, C. (2019). Smart condition monitoring for industry 4.0 manufacturing processes: An ontology-based approach. *Cybernetics and Systems*, Vol. *50*(2), P. 82–96, DOI: 10.1080/01969722.2019.1565118.

37. Givehchi, O., Landsdorf, K., Simoens, P., & Colombo, A. W. (2017). Interoperability for industrial cyber-physical systems: An approach for legacy systems. *IEEE Transactions on Industrial Informatics*, Vol. *13*, Iss. 6 , P. 3370 – 3378, DOI: 10.1109/TII.2017.2740434.

38. Romero, D., Bernus, P., Noran, O., Stahre, J., & Fast-Berglund, Å. (2016, September). The operator 4.0: human cyber-physical systems & adaptive automation towards human-automation symbiosis work systems. In *IFIP international conference on advances in production management systems*, Vol *488*, P. 677–686, DOI: 10.1007/978-3-319-51133-7_80.

39. Leitão, P., Colombo, A. W., & Karnouskos, S. (2016). Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges. *Computers in industry*, Vol. *81*, P. 11–25, DOI:10.1016/j.compind.2015.08.004.

40. Monostori, L., Kádár, B., Bauernhansl, T., Kondoh, S., Kumara, S., Reinhart, G., ... & Ueda, K. (2016). Cyber-physical systems in manufacturing. *Cirp Annals*, Vol. *65*(2), P. 621–641, DOI:10.1016/j.cirp.2016.06.005.

41. Yu, X., & Xue, Y. (2016). Smart grids: A cyber–physical systems perspective. *Proceedings of the IEEE*, Vol. *104*(5), P. 1058–1070, DOI:10.1109/JPROC.2015.2503119.

42. Stefano Zanero.(2017). Cyber-Physical Systems. *Computer*, Volume: *50*, Issue: 4, P: 14 – 16, DOI: 10.1109/MC.2017.105.

43. Muccini, H., Sharaf, M., & Weyns, D. (2016). Self-adaptation for cyber-physical systems: a systematic literature review. In *Proceedings of the 11th international symposium on software engineering for adaptive and self-managing systems* , P. 75–81, DOI:10.1145/2897053.2897069.

44. Iarovyi, S., Mohammed, W. M., Lobov, A., Ferrer, B. R., & Lastra, J. L. M. (2016). Cyber–physical systems for open-knowledge-driven manufacturing execution systems. *Proceedings of the IEEE*, Volume:*104* , Issue: 5, P. 1142 – 1154, DOI:10.1109/JPROC.2015.2509498.

45. Yao, X., Zhou, J., Lin, Y., Li, Y., Yu, H., & Liu, Y. (2019). Smart manufacturing based on cyber-physical systems and beyond. *Journal of Intelligent Manufacturing*, Volume *30*(8), P. 2805–2817 DOI:10.1007/s10845-017-1384-5.

46. Bangemann, T., Riedl, M., Thron, M., & Diedrich, C. (2016). Integration of classical components into industrial cyber–physical systems. *Proceedings of the IEEE*, Volume:*104*, Issue:5, P. 947 – 959, DOI: 10.1109/JPROC.2015.2510981.

47. Jirkovský, V., Obitko, M., & Mařík, V. (2016). Understanding data heterogeneity in the context of cyber-physical systems integration. *IEEE Transactions on Industrial Informatics*, Volume *13*(2), P. 660–667, DOI: 10.1109/TII.2016.2596101.

48. Mezgár, I., & Pedone, G. (2019). Cloud-based manufacturing (CBM) interoperability in Industry 4.0. In *Technological Developments in Industry 4.0 for Business Applications* , IGI Global, P.171–198, DOI: 10.4018/978-1-5225-4936-9.ch008

49. Tao, F., Qi, Q., Wang, L., & Nee, A. Y. C. (2019). Digital twins and cyber–physical systems toward smart manufacturing and industry 4.0: Correlation and comparison. *Engineering*. Volume *5*, Issue 4, P. 653–661 DOI:10.1016/ j.eng.2019.01.014.

50. Li, G., Tan, J., & Chaudhry, S. S. (2019). Industry 4.0 and big data innovations. *Enterprise information systems*, Volume *13*, Issue 2, P. 145–147 DOI:10.1080/17517575.2018.1554190.

51. Lu, Y., & Xu, X. (2019). Cloud-based manufacturing equipment and big data analytics to enable on-demand manufacturing services. *Robotics and Computer-Integrated Manufacturing*, Volume *57*, P. 92–102 DOI:10.1016/ j.rcim.2018.11.006.

52. Silva, J., Gaitán, M., Varela, N., & Lezama, O. B. P. (2019). Engineering teaching: simulation, industry 4.0 and big data. In *International Conference On Computational Vision and Bio Inspired Computing* . P. 226–232 DOI:10.1007/ 978-3-030-37218-7_26.

53. Bogoviz, A., Lobova, S., Karp, M., Vologdin, E., & Alekseev, A. (2019). Diversification of educational services in the conditions of industry 4.0 on the basis of AI. *On the Horizon*. Vol. *27*, No. 3/4, P. 206–212. DOI:10.1108/OTH-06-2019-0031.

54. Inés Sittón-Candanedo, Ricardo S. Alonso, Sara Rodríguez-González, José Alberto García Coria, Fernando De La Prieta (2020). Edge Computing Architectures in Industry 4.0: A General Survey and Comparison. *In 14th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2019). Advances in Intelligent Systems and Computing*, Vol. *950*, P. 121–131, DOI:10.1007/978-3-030-20055-8_12.

55. Alexander Kropp, Robert-Steve Schmoll, Giang T. Nguyen, Frank H. P.(2019) Fitzek Demonstration of a 5G Multi-access Edge Cloud Enabled Smart Sorting Machine for Industry 4.0. *In 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, DOI: 10.1109/CCNC.2019. 8651732.

56. Klingenberg, C., Borges, M. and Antunes Jr, J. (2019). Industry 4.0 as a data-driven paradigm: a systematic literature review on technologies. *Journal of Manufacturing Technology Management*, Vol.*32*, No. 3, P.570–592 DOI:10.1108/JMTM-09-2018-0325.

57. Sundarakani, B., Kamran, R., Maheshwari, P. and Jain, V. (2019). Designing a hybrid cloud for a supply chain network of Industry 4.0: a theoretical framework. *Benchmarking: An International Journal*, Vol. *ahead-of-print*, No. ahead-of-print, DOI:10.1108/BIJ-04-2018-0109.

58. C. Occhiuzzi, S. Amendola, S. Nappi, N. D'Uva, G. Marrocco. (2019) RFID Technology for Industry 4.0: Architectures and Challenges. *In IEEE International Conference on RFID Technology and Applications (RFID-TA).* DOI:10.1109/RFID-TA.2019.8892049.

59. Leal P., Madeira R.N., Romão T. (2019) Model-Driven Framework for Human Machine Interaction Design in Industry 4.0. *Human-Computer Interaction – INTERACT 2019. Lecture Notes in Computer Science*, Vol. *11749*, P. 644–648 DOI:10.1007/978-3-030-29390-1_54.

60. Aitor Ardanza, Aitor Moreno, Álvaro Segura, Mikel de la Cruz, Daniel Aguinaga (2019) Sustainable and flexible industrial human machine interfaces to support adaptable applications in the Industry 4.0 paradigm. *Journal International Journal of Production Research*, Volume *57*, Issue 12, P. 4045–4059 DOI:10.1080/00207543.2019.1572932.

61. Roda-Sanchez L., Olivares T., Garrido-Hidalgo C., Fernández-Caballero A. (2019) Gesture Control Wearables for Human-Machine Interaction in Industry 4.0. *From Bioinspired Systems and Biomedical Applications to Machine Learning. IWINAC 2019. Lecture Notes in Computer Science*, Vol. 11487. P. 99–108 DOI: 10.1007/978-3-030-19651-6_10.

62. Visit Hirankitti. (2019). An Intelligent Agent Framework for SCADA. *In 2019 First International Conference on Digital Data Processing (DDP)*, P. 67–81, DOI:10.1109/DDP.2019.00024.

63. Luo Pan, Wu Zhizheng, Chen Fan (2019) Design of SCADA System for CNC Grinder Workshop Based on SIMATIC NET. *In 2019 IEEE International Conference on Smart Manufacturing, Industrial & Logistics Engineering (SMILE)*. P. 94–115, DOI:10.1109/SMILE45626.2019.8965296.

64. Felipe Orellana, Romina Torres. (2019) From legacy-based factories to smart factories level 2 according to the Industry 4.0. *In Journal International Journal of Computer Integrated Manufacturing*, Volume *32*, Issue 4-5, P. 441–451, DOI:10.1080/0951192X.2019.1609702.

65. Clemens Fallera, Max Höftmanna. (2018). Service-oriented communication model for cyber-physical-production-systems. *Procedia CIRP*, Volume *67*, P. 156–161, DOI:10.1016/j.procir.2017.12.192

66. Francisco Almada-Lobo.(2015). The Industry 4.0 revolution and the future of Manufacturing Execution Systems (MES). *Journal of Innovation Management (JIM)*, Vol.3, Issue 4, P. 16–21, DOI: 10.24840/2183-0606_003. 004_0003.

67. Lei Yue, Linkun Wang, Pengfei Niu, Nan Zheng. (2019). Building a reference model for a Manufacturing Execution System (MES) platform in an Industry 4.0 context. *Journal of Physics: Conference Series*, Volume *1345*, DOI: 10.1088/1742-6596/1345/6/062002.

68. Jiří Vyskočil, Petr Kadera. (2019). Plan Executor MES: Manufacturing Execution System Combined with a Planner for Industry 4.0 Production Systems. *In International Conference on Industrial Applications of Holonic and Multi-Agent Systems (HoloMAS 2019)*, *Lecture Notes in Computer Science*, Vol. *11710*, P. 67–80, DOI:10.1007/978-3-030-27878-6_6.

69. Mantravadi Soujanya, Møller Charles. (2019). An Overview of Next-generation Manufacturing Execution Systems: How important is MES for Industry 4.0?. *Procedia Manufacturing*, Vol. *30*, P. 588–595, DOI:10.1016/ j.promfg.2019.02.083.

70. Mahyar Azarmipour, Haitham Elfaham, Caspar Gries, Ulrich Epple (2019). PLC 4.0: A Control System for Industry 4.0. *In IECON 2019 - 45th Annual*

Conference of the IEEE Industrial Electronics Society, P. 356–408, DOI:10.1109/IECON.2019.8927026.

71. Markus Schäfer, Patrick Moll, Lukas Brocke, Sven Coutandin, Jürgen Fleischer. (2019) Model for Web-Application based Configuration of Modular Production Plants with automated PLC Line Control Code Generation. *Procedia CIRP*, Volume *83*, P. 292–297 DOI:10.1016/j.procir.2019.03.126.

72. . Karabegovic, E. Karabegovic, M. Mahmic, E. Husak. (2019). The Role of Smart Sensors in Production Processes and the Implementation of Industry 4.0. *Журнал інженерних наук*.Т.*6*, № 2. С. 8–13, DOI:10.21272/jes.2019.6(2).b2.

73. Maurizio Galetto, Alessandro Schiavi, Gianfranco Genta, Andrea Prato, Fabrizio Mazzoleni. (2019). Uncertainty evaluation in calibration of low-cost digital MEMS accelerometers for advanced manufacturing applications. *CIRP Annals*, Volume 68, Issue 1, P. 535–538, DOI:10.1016/j.cirp.2019.04.097.

74. Roman Szewczyk, Jiří Krejsa, Michał Nowicki, Anna Ostaszewska-Liżewska.(2030). Mechatronics 2019: Recent Advances Towards Industry 4.0. *Advances in Intelligent Systems and Computing*, Springer Nature Switzerland AG 2020, P. 515, ISBN 978-3-030-29992-7, DOI:10.1007/978-3-030-29993-4.

75. Andreas M. Radke, Minh Trang Dang, Albert Tan. (2020). Using robotic process automation (RPA) to enhance item master data maintenance process. *Scientific Journal of Logistics*, Volime *16* (1), P.129–140 DOI:10.17270/J.LOG.2020.380.

76. LilianaZarco, JörgSiegerta, Thomas Bauernhansl. (2019). Software Model Requirements Applied to a Cyber-Physical Modular Robot in a Production Environment. *Procedia CIRP,* Volume *81*, P. 352–357, DOI:10.1016/j.procir. 2019.03.061.

77. Mehrpouya, M.; Dehghanghadikolaei, A.; Fotovvati, B.; Vosooghnia, A.; Emamian, S.S.; Gisario, A. (2019). The Potential of Additive Manufacturing in the Smart Factory Industrial 4.0: *A Review. Appl. Sci*, Volume *9(18)*, P. 54–68, DOI:10.3390/app9183865.

78. Dilberoglu, U.M.; Gharehpapagh, B.; Yaman, U.; Dolen, M. (2017) The role of additive manufacturing in the era of industry 4.0. *Procedia Manufacturing*, Volume *11*, P. 545–554, DOI:10.1016/j.promfg.2017.07.148.

79. DIN SPEC 91345:2016-04 (E) Reference Architecture Model Industrie 4.0 (RAMI4.0). [Electronic resource]. URL: https://dx.doi.org/10.31030/2436156. (Дата звернення:25.03.2020).

80. Yevsieiev V., Jijavadze O. (2020). Analysis architectural model of Industry 4.0 (RAMI 4.0). *XXIV Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті»,* Т. *2*. С.93–94.

81. Kunath M., Winkler H. (2019) Adaptive Assistenzsysteme zur Entscheidungsunterstützung für die dynamische Auftragsabwicklung: Konzeptionelle Überlegungen und Anwendungsszenarien unter Berücksichtigung des Digitalen Zwillings des Produktionssystems. *Industrie 4.0 und Digitale Transformation*, P. 269–294, DOI:10.1007/978-3-658-24576-4_12.

82. IEC PAS 63088:2017 (E) Smart manufacturing-Reference Architecture Model Industrie 4.0 (RAMI4.0). [Electronic resource]. URL: https://webstore.iec.ch/publication/30082. (Дата звернення:30.04.2020).

83. DIN SPEC 16593-1:2018-04 (E) RM-SA - Reference Model for Industrie 4.0 Service Architectures - Part 1: Basic Concepts of an Interaction-based Architecture. [Electronic resource]. URL:https://dx.doi.org/10.31030/2838942. (Дата звернення:30.04.2020).

84. PD IEC/TS 62832-1:2016 Industrial-process measurement, control and automation. Digital factory framework. General principles. [Electronic resource]. URL: https://standards.globalspec.com/std/10063259/iec-ts-62832-1. (Дата звернення: 30.04.2020).

85. IEC 62264-1:2013 Enterprise-control system integration — Part 1: Models and terminology. [Electronic resource]. URL: https://www.iso.org/ru/standard/57308.html. (Дата звернення: 30.04.2020).

86. IEC 62541-100:2015 OPC Unified Architecture - Part 100: Device Interface. [Electronic resource]. URL: https://webstore.iec.ch/publication/21987. (Дата звернення: 30.04.2020).

87. Yang Liu, Yu Peng, Bailing Wang, Sirui Yao, Zihe Liu. (2107) Review on cyber-physical systems. *In IEEE/CAA Journal of Automatica Sinica*, Vol. *4*, Issue 1, P. 27–40, DOI:10.1109/JAS.2017.7510349.

88. Fei Tao, Qinglin Qia, Lihui Wang, A.Y.C.Nee. (2019) Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison. *Engineering*, Volume *5*. Issue 4. P. 653–661, DOI: 10.1016/j.eng.2019.01.014.

89. Zhou Ji, Li Peigen, Zhou Yanhong ,Wang Baicun, Zang Jiyuan, Meng Liu. (2018). Toward New-Generation Intelligent Manufacturing. *Engineering,* Volume *4*, Issue 1, P.11–20, DOI:10.1016/j.eng.2018.01.002.

90. Ilge Akkaya, Patricia Derler, Shuhei Emoto, Edward A. Lee. (2016) Systems Engineering for Industrial Cyber–Physical Systems Using Aspects. *Proceedings of the IEEE*, Volume *104*, Issue 5, P. 997–1012, DOI: 10.1109/ JPROC.2015.2512265.

91. Paulo Leitão, Stamatis Karnouskos, Luis Ribeiro, Jay Lee, Thomas Strasser, Armando W. Colombo. (2016). Smart Agents in Industrial Cyber–Physical Systems. *Proceedings of the IEEE*, Volume *104*, Issue 5, P. 1086–1101, DOI: 10.1109/JPROC.2016.2521931.

92. P.Hehenberger, B.Vogel-Heuser, D.Bradley, B.Eynard, T.Tomiyama, S.Achichef. (2016). Design, modelling, simulation and integration of cyber physical systems: Methods and applications. *Computers in Industry,* Volume *82*, P. 273–289, DOI:10.1016/j.compind.2016.05.006.

93. Ezio Bartocci, Jyotirmoy Deshmukh, Alexandre Donzé, Georgios Fainekos, Oded Maler, Dejan Ničković, Sriram Sankaranarayanan. (2018). Specification-based monitoring of cyber-physical systems: a survey on theory, tools and applications. *Lectures on Runtime Verification. Lecture Notes in Computer Science*, Vol. 10457, P.135–175, DOI: 10.1007/978-3-319-75632-5_5.

94. Li Da Xu, Lian Duan. (2019). Big data for cyber physical systems in industry 4.0: a survey. *Journal Enterprise Information Systems*, Volume *13*, Issue 2, P.148–169, DOI:10.1080/17517575.2018.1442934.

95. BorjaBordel, RamónAlcarria, TomásRobles, Diego Martín. (2017). Cyber–physical systems: Extending pervasive sensing from control theory to the Internet of Things. *Pervasive and Mobile Computing*, Volume *40*, P. 156–184, DOI:10.1016/j.pmcj.2017.06.011.

96. Yevsieiev V., Bronnikov A. (2020). Comlexity development analysis of cyber-physical production systems for smart manufacturing. *The X th International scientific and practical conference «Trends in the development of modern scientific thought»*, P.699–703. DOI:10.46299/ISG.2020.II.X.

97. Why Germany is the place to be for Industrie 4.0. [Electronic resource]. URL: https://www.gtai.de/gtai-en/invest/industries/industrie-4-0. (Дата звернення: 30.01.2020)

98. The next horizon for industrial manufacturing: Adopting disruptive digital technologies in making and delivering. [Electronic resource]. URL: https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/the-next-horizon-for-industrial-manufacturing. (Дата звернення: 30.01.2020)

99. Industry 4.0: The Future of Productivity and Growth in Manufacturing. [Electronic resource]. URL: https://www.bcg.com/publications/2015/engineered_products_project_business_industry_4_future_productivity_growth_manufacturing_industries.aspx. (Дата звернення: 30.01.2020)

100. Pericles Loucopoulos, Evangelia Kavakli, Natalia Chechina. (2019). Requirements Engineering for Cyber Physical Production Systems. *Advanced Information Systems Engineering. CAiSE 2019. Lecture Notes in Computer Science*, Vol. *11483*, P. 276–291, DOI: 10.1007/978-3-030-21290-2_18.

101. O. Cardin. (2019). Classification of cyber-physical production systems applications: Proposition of an analysis framework. *Computers in Industry*, Volume *104*, P. 11–21, DOI:10.1016/j.compind.2018.10.002.

102. S. Vijayakumar, N. Dhasarathan, P. Devabalan, C. Jehan. (2019). Advancement and Design of Robotic Manipulator Control Structures on Cyber Physical Production System. *Journal of Computational and Theoretical Nanoscience*, Volume *16*, Number 2, P.659–663(5) DOI:10.1166/jctn.2019.7786.

103. Hermann Meissner, Jan C. Auricha. (2019). Implications of cyber-physical production systems on integrated process planning and scheduling. *Procedia Manufacturing*, Volume *28*, P. 167–173 DOI:10.1016/j.promfg.2018.12.027.

104. Fazel Ansari, Robert Glawar, Tanja Nemeth. (2019). PriMa: a prescriptive maintenance model for cyber-physical production systems. *Journal International Journal of Computer Integrated Manufacturing*, Volume *32*, Issue 4, P. 482–503, DOI:10.1080/0951192X.2019.1571236.

105. Hyoung Seok Kang, Ju Yeon Lee, Sang Do Noh. (2019) A dynamic processing methodology of manufacturing data for the automated throughput analysis in cyber-physical production environment. *Journals Concurrent Engineering* Volume *27*, Issue 2, P. 155–169, DOI:10.1177/1063293X19842264.

106. A V Gurjanov, D A Zakoldaev, A V Shukalov, I O Zharinov. (2019). Formation principles of digital twins of Cyber-Physical Systems in the smart factories of Industry 4.0. *IOP Conf. Series: Materials Science and Engineering,* Volume *483,* P.301–313, DOI:10.1088/1757-899X/483/1/012070.

107. Industrial Agents: Emerging Applications of Software Agents in Industry. *Edited by:* Paulo Leitão, Stamatis Karnouskos, 2015, Elsevier. Inc, P. 447, ISBN: 978-0-12-800341-1.

108. Ribeiro L, Hochwallner M (2018) On the design complexity of cyber-physical production systems. *Complexity,* P. 1–13, DOI: 10.1155/2018/4632195.

109. Kai Li, Tao Zhou, Bo-hai Liu, Hui Li. (2018). A multi-agent system for sharing distributed manufacturing resources. *Expert Systems with Applications,* Volume *99,* Pages 32–43, DOI: 10.1016/j.eswa.2018.01.027.

110. Haoues M, Sellami A, Ben-Abdallah H, Cheikhi L. (2017). A guideline for software architecture selection based on ISO 25010 quality related characteristics. *Int J Syst Assur Eng Manag*, Volume *8*, P.886–909, DOI: 10.1007/s13198-016-0546-8.

111. Farid AM, Ribeiro L. (2015). An axiomatic design of a multiagent reconfigurable mechatronic system architecture. *IEEE Trans Ind Informatics*, Volume *11*, P. 1142–1155, DOI:10.1109/TII.2015.2470528.

112. Cruz SLA, Mayer F, Schütz D, Vogel-Heuser B (2018) Platform independent multi-agent system for robust networks of production systems. *IFAC-PapersOnLine*, Volume *51* P.1261–1268, DOI:10.1016/j.ifacol.2018.08.359.

113. Lüder A, Calá A, Zawisza J, Rosendahl R. (2017). Design pattern for agent based production system control—a survey. *In: 13th IEEE conference on automation science and engineering (CASE)*. P. 717–722, DOI: 10.1109/ COASE.2017.8256187

114. Rehberger S, Spreiter L, Vogel-Heuser B. (2017). An agent-based approach for dependable planning of production sequences in automated production systems. *At-Automatisierungstechnik*, Volume *65*(11), P.766–778, DOI: 10.1515/auto-2017-0040.

115. L. Ribeiro, M. Hochwallner. (2018). On the design complexity of cyber-physical production systems. *Complexity*. P. 1–13, DOI:10.1155/2018/ 4632195.

116. Cruz SLA, Vogel-Heuser B. (2017). Comparison of agent oriented software methodologies to apply in cyber physical production systems. *In: 15th international conference on industrial informatics*, *INDIN. IEEE*, P. 65–71, DOI:10.1109/INDIN.2017.8104748.

117. Lüder A, Schleipen M, Schmidt N, et al. (2018). One step towards an industry 4.0 component. *In 13th IEEE conference on automation science and engineering, CASE*, P.1268–1273, DOI: 10.1109/COASE.2017.8256275.

118. ANSI/ISA-95.00.01-2010 (IEC 62264-1 Mod) Enterprise-Control System Integration - Part 1: Models and Terminology. [Electronic resource]. URL: https://www.isa.org/products/ansi-isa-95-00-01-2010-iec-62264-1-mod-enterprise. (Дата звернення: 02.05.2020).

119. The structure of the administration shell: trilateral perspective from France, Italy and Germany. [Electronic resource]. URL: https://www.de.digital/DIGITAL/Redaktion/EN/Publikation/the-structure-of-the-administration-shell.pdf?__blob=publicationFile&v=3. (Дата звернення: 02.05.2020).

120. L. Ribeiro. (2017). Cyber-physical production systems' design challenges. *In 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, P. 1189–1194, DOI: 10.1109/ISIE.2017.8001414.

121. Yevsieiev V., Bronnikov A. (2020) Analysis of the multi-agent systems application to solve the problem of cyberphysical production systems development. *The IV th International scientific and practical conference «Integration of scientific bases into practice»*. P.459–462, DOI:10.46299/ISG.2020.IV.

122. Yevsieiev V., Bronnikov A. (2020), Structural model of a cyber-physical production system based on multi-agent systems analysis. *Матеріали VII Міжнародної науково-технічної Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами»*, С.312–313.

123. Yevsieiev V., Bronnikov A. (2020). Analysis of the cyber-physical production systems implementation impact to achieve the goals of lean production. *The IIth International scientific and practical conference «Development of scientific and practical approaches in the era of globalization»,* P.221–226, DOI:10.46299/ISG.2020.II.II.

124. Syed Imran Shafiq, Edward Szczerbicki, Cesar Saninc. (2019). Proposition of the methodology for Data Acquisition, Analysis and Visualization in support of Industry 4.0. *Procedia Computer Science*, Volume *159*, P. 1976–1985, DOI:10.1016/j.procs.2019.09.370.

125. Sebastian Thiede, Artem Turetskyy, Arno Kwade, Sami Kara, Christoph Herrmann. (2019). Data mining in battery production chains towards multi-criterial quality prediction. *CIRP Annals*, Volume *68*, Issue 1, P. 463–466, DOI:10.1016/j.cirp.2019.04.066.

126. Guejong Jo, Su-Hwan Jang, Jongpil Jeong. (2019). Design and Implementation of CPPS and Edge Computing Architecture based on OPC UA Server. *Procedia Computer Science*, Volume *155*, P. 97–104, DOI:10.1016/j.procs.2019.08.017.

127. Malhotra J., Iqbal F., Sahu A.K., Jha S. (2019) A Cyber-Physical System Architecture for Smart Manufacturing. *Advances in Forming, Machining and Automation. Lecture Notes on Multidisciplinary Industrial Engineering*, P. 637–647, DOI:10.1007/978-981-32-9417-2_53.

128. Christine Schulze, Sebastian Thiede, Bastian Thiede, Denis Kurle, Stefan Blume, Christoph Herrmann. (2019). Cooling tower management in manufacturing companies: A cyber-physical system approach. *Journal of Cleaner Production*, Volume *211*, P. 428–441, DOI:10.1016/j.jclepro.2018.11.184.

129. Kashif Mahmood, Tatjana Karaulova, Tauno Otto, Eduard Shevtshenko. (2019). Development of cyber-physical production systems based on modelling technologies. *Proceedings of the Estonian Academy of Sciences*, Volume *68*(4), P.348–355, DOI:10.3176/proc.2019.4.02.

130. Sergei Kaganski, Jüri Majak, Kristo Karjust, Silver Toompalu. (2017). Implementation of key performance indicators selection model as part of the Enterprise Analysis Model. *Procedia CIRP*, Volume *63*, P. 283−288 DOI:10.1016/j.procir.2017.03.143.

131. O.Mörth, C.Emmanouilidis, N.Hafner, M.Schadler. (2020). Cyber-Physical Systems for Performance Monitoring in Production Intralogistics. *Computers & Industrial Engineering*, Volume *142*, P. 328−341, DOI:10.1016/j.cie.2020.106333.

132. R. van de Sand, S. Schulz, J. Reiff-Stephan. (2019). Smart Process Communication for Small and Medium-Sized Enterprises. *Proceedings of the I-ESA Conferences*, Vol. 9, P. 411−420, DOI:10.1007/978-3-030-13693-2_34.

133. Hammer M. (2019) Digitization Perspective: Impact of Digital Technologies in Manufacturing. *In: Management Approach for Resource-Productive Operations. Industrial Management*, P. 27−68, DOI:10.1007/978-3-658-22939-9_3.

134. Chiara Cimini, Fabiana Pirola, Roberto Pinto, Sergio Cavalieri. (2020). A human-in-the-loop manufacturing control architecture for the next generation of production systems. *Journal of Manufacturing Systems*, Volume *54*, P. 258−271, DOI:10.1016/j.jmsy.2020.01.002.

135. Tobias Wagner, Christoph Herrmann, Sebastian Thiede. (2017). Industry 4.0 Impacts on Lean Production Systems. *Procedia CIRP*, Volume 63, P. 125−131, DOI:10.1016/j.procir.2017.02.041.

136. Tortorella, G., Miorando, R., Meiriño, M. and Sawhney, R. (2019). "Managing practitioners' experience and generational differences for adopting lean production principles". *The TQM Journal*, Vol. *31* No. 5, P. 758−771, DOI:10.1108/TQM-02-2019-0041.

137. Rainer Stark, Carina Fresemann, Kai Lindow. (2019). Development and operation of Digital Twins for technical systems and services. *CIRP Annals*, Volume *68*, Issue 1, P. 129−132, DOI:10.1016/j.cirp.2019.04.024.

138. H.S. Kang, J.Y. Lee, S. Choi, H. Kim, J.H. Park, J.Y. Son, et al. (2016). Smart manufacturing: past research, present findings, and future directions. *Int J Precis Eng Manuf-Green Technol*, Volume *3*, P. 111−128, DOI:10.1007/s40684-016-0015-5.

139. L.D. Xu, E.L. Xu, L. Li Industry 4.0: state of the art and future trends Int J Prod Res, 56 (2018), pp. 2941−2962, (10.1080/00207543.2018.1444806)

140. S. Wang, J. Wan, D. Li, C. Zhang. (2016). Implementing Smart Factory of Industrie 4.0: An Outlook. *International Journal of Distributed Sensor Networks,* Volume *12,* Issue 1, P. 634−648, DOI:10.1155/2016/3159805.

141. A. Syberfeldt, M. Holm, O. Danielsson, L. Wang, R.L. (2016). Brewster Support systems on the industrial shop-floors of the future – operators' perspective on augmented reality. *Procedia Cirp*, Volume *44*, P. 108−113, DOI:10.1016/j.procir.2016.02.017.

142. Weber, C., Königsberger, J., Kassner, L., Mitschang, B. (2017). M2DDM – A maturity model for data-driven manufacturing, *Procedia CIRP*, Vol. *63*, P. 173-178, DOI:10.1016/j.procir.2017.03.309.

143. M. Resman, M. Pipan, M. Šimic, N. Herakovič. (2019). A new architecture model for smart manufacturing: A performance analysis and comparison with the RAMI 4.0 reference model. *Advances in Production Engineering & Management,* Volume *14*, Number 2, P.153–165, DOI:10.14743/apem2019.2.318.

144. Yevsieiev V., Jijavadze O. (2020). Analysis architectural model of Industry 4.0 (RAMI 4.0). *XXIV Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті»,* Т. *2*, С. 93–94.

145. Yübo Wang, Thilo Towara, and Reiner Anderl. (2017). Topological Approach for Mapping Technologies in Reference Architectural Model Industrie 4.0 (RAMI 4.0). *Proceedings of the World Congress on Engineering and Computer Science 2017, WCECS 2017*, Vol. *2*, P.982–990.

146. Zezulka, F., Marcon, P., Vesely, I., Sajdl, O. (2016). Industry 4.0 – An introduction in the phenomenon, *IFACPapersOnLine*, Vol. *49*, No. 25, P. 8–12, DOI:10.1016/j.ifacol.2016.12.002.

147. Dimitris Mourtzis, Antonis Gargallis, Vasilios Zogopoulos. (2019). Modelling of Customer Oriented Applications in Product Lifecycle using RAMI 4.0. *Procedia Manufacturing,* Volume *28*, P. 31–36, DOI: 10.1016/j.promfg. 2018.12.006

148. M.A. Pisching, M.A.O. Pessoa, F.Junqueira, D.J. dos Santos Filho, P.E. Miyagi. (2018). An architecture based on RAMI 4.0 to discover equipment to process operations required by products, *Computers & Industrial Engineering*, Vol. *125*, P. 574–591, DOI:10.1016/j.cie.2017.12.029.

149. Elder Hernández, Pedro Senna, Daniela Silva, Rui Rebelo, Ana C. Barros, César Toscano.(2019). Implementing RAMI4. 0 in Production-A Multi-case Study. *International Conference of Progress in Digital and Physical Manufacturing, ProDPM 2019: Progress in Digital and Physical Manufacturing*, P. 49–56, DOI:10.1007/978-3-030-29041-2_6.

150. Alexandros Bousdekis, Katerina Lepenioti, Dimitrios Ntalaperas, Danai Vergeti, Dimitris Apostolou, Vasilis Boursinos. (2019). A RAMI 4.0 View of Predictive Maintenance: Software Architecture, Platform and Case Study in Steel Industry. *Advanced Information Systems Engineering Workshops. CAiSE 2019. Lecture Notes in Business Information Processing*, Vol. *349*, P. 95–106, DOI: 10.1007/978-3-030-20948-3.

151. Václav Kaczmarczyk, Tomáš Benešl, Zdeněk Bradáč, Petr Fiedler, Zuzana Kaczmarczyková. (2019). SkuBATCH-System for control of technological processes. *IFAC-PapersOnLine*, Volume *52*, Issue 27, P. 477–483, DOI:10.1016/j.ifacol.2019.12.709.

152. Llamuca J.D., Garcia C.A., Naranjo J.E., Rosero C., Alvarez-M E., Garcia M.V. (2020). Integrating ISA-95 and IEC-61499 for Distributed Control System Monitoring. *Information and Communication Technologies of Ecuador (TIC.EC). TICEC 2019. Advances in Intelligent Systems and Computing*, Vol. *1099*, P. 66–80, DOI:10.1007/978-3-030-35740-5_5.

153. Bernhard Wally, Jirí Vyskočil, Petr Novák, Christian Huemer, Radek Šindelar, Petr Kadera, Alexandra Mazak. (2019). Manuel Wimmer Flexible Production Systems: Automated Generation of Operations Plans Based on ISA-95 and PDDL. *IEEE Robotics and Automation Letters*, Volume *4*, Issue 4, P. 4062–4069, DOI: 10.1109/LRA.2019.2929991.

154. Rafal Cupek, Adam Ziebinski, Marek Drewniak, Marcin Fojcik. (2019). Knowledge integration via the fusion of the data models used in automotive production systems. *Journal Enterprise Information Systems*, Volume *13*, Issue 7-8, P. 1094–1119, DOI:10.1080/17517575.2018.1489563.

155. Leon F. McGinnis.(2019). Formalizing ISA-95 Level 3 Control with Smart Manufacturing System Models. *National Institute of Standards and Technology*, P. 86, DOI:10.6028/NIST.GCR.19-022.

156. Byeong Woo Jeon, Jumyung Um, Soo Cheol Yoon &Suh Suk-Hwan. (2017). An architecture design for smart manufacturing execution system. *Computer-Aided Design and Applications*, Volume *14*, Issue 4, P.472-485, DOI: 10.1080/16864360.2016.1257189.

157. Jehn-Ruey Jiang. (2108). An improved cyber-physical systems architecture for Industry 4.0 smart factories. *Advances in Mechanical Engineering*, Vol. *10*(6), P.1–15, DOI: 10.1177/1687814018784192.

158. Lee J., Bagheri B., Kao H.-A. (2015). A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manufacturing Letters,* Vol. *3*, P.18–23, DOI:10.1016/j.mfglet.2014.12.001.

159. Usharani Hareesh Govindarajan, Amy J.C. Trappey, Gopal Kumar. (2018). Latent Dirichlet Allocation modeling for CPS patent topic discovery. *Proceedings of the 2018 International Conference on Industrial Enterprise and System Engineering (IcoIESE 2018), Atlantis Highlights in Engineering (AHE),* Volume *2*, P.31–36, DOI:10.2991/icoiese-18.2019.6.

160. Martin Frické, The Knowledge. (2019). Pyramid: the DIKW Hierarchy. *Ko knowledge organization*, Seite 33–46, ISSN online: 0943-7444, DOI:10.5771/0943-7444-2019-1-33.

161. D. Nell, E.H. Mathews, P. Maré. (2019). Industry 4.0 Roll-out Strategy for Dynamic Mine Heat Load Management. *South African Journal of Industrial Engineering*, Vol.30, P. 145–165, DOI:10.7166/30-3-2232.

162. Yucong Duan, Zhihui Lu, Zhangbing Zhou, Xiaobing Sun, Jie Wu.(2019). Data Privacy Protection for Edge Computing of Smart City in a DIKW Architecture. *Engineering Applications of Artificial Intelligence*, Volume *81*, P. 323–335, DOI:10.1016/j.engappai.2019.03.002.

163. H. Gao, Y. Duan, L. Shao, et al. (2019). Transformation-based processing of typed resources for multimedia sources in the IoT environment. *Wireless Netw*, P.92–113, DOI:10.1007/s11276-019-02200-6.

164. Petar Radanliev, David De Roure, Razvan Nicolescu, Michael Huth. (2019). A reference architecture for integrating the Industrial Internet of Things in the Industry 4.0. *Computers and Society,* P. 12–25, DOI: 10.13140/ RG.2.2.26854.47686.

165. Yevsieiev V., Bronnikov A. (2020). Analysis of architectural models for representing the integration of cyber-physical production systems hierarchical levels. *Manufacturing & Mechatronic Systems 2020: Proceedings of IVth International Conference (M&MS 2020)*, P.17–19.

166. Andreas Schumacher, Tanja Nemetha, Wilfried Sihna. (2018) Roadmapping towards industrial digitalization based on an Industry 4.0 maturity model for manufacturing enterprises. *12th CIRP Conference on Intelligent Computation in Manufacturing Engineering*, Volume *79*, P. 409–414, DOI:10.1016/j.procir.2019.02.110.

167. Jung K, Kulvatunyou B, Choi S, Brundage MP. (2016). An overview of a smart manufacturing system readiness assessment. *IFIP - Advances in Information and Communication Technology*, Vol. 488, P.705-712, DOI:10.1007/978-3-319-51133-7_83.

168. Liebrecht C, Burgin J, Benterbusch J, Kiefer C, Lanza G. (2016). Shopfloor-getriebene Einführung von Industrie 4.0. *Werkstattstechnik*, Vol. *106*(7/8), P. 539–543. ISSN: 1436-5006.

169. Schmitz S. (2016). Industrie-4.0-Reifegradindex zur Standortbestimmung der Unternehmen. *Unternehmen der Zukunft - Zeitschrift für Betriebsorganisation und Unternehmensentwicklung,* Vol.17(1), P.31–33.

170. De Carolis A, Macchi M, Negri E, Terzi S. (2016). A Maturity Model for Assessing the Digital Readiness of Manufacturing Companies. *Advances in Production Management Systems. The Path to Intelligent, Collaborative and Sustainable Manufacturing*, P.13–20, DOI: 10.1007/978-3-319-66923-6_2.

171. Nemeth T, Ansari F, Sihn W, Haslhofer B, Schindler A.(2018). PriMa-X: A Reference Model for Realizing Prescriptive Maintenance and Assessing its Maturity Enhanced by Machine Learning. *Procedia CIRP*, Vol.*72*, P.1039–1044, DOI:10.1016/j.procir.2018.03.280.

172. Weber C, Königsberger J, Kassner L, Mitschang B. (2017). M2DDM–a maturity model for data-driven manufacturing. *Procedia CIRP*, Vol.*63*, P.173–178, DOI:10.1016/j.procir.2017.03.309.

173. Leyh C., Schäffer T., Bley K., Forstenhäusler S. (2016). Assessing the IT and Software Landscapes of Industry 4.0-Enterprises: The Maturity Model SIMMI 4.0. *Lecture Notes in Business Information Processing*, Vol *277*, P.103–119, DOI:10.1007/978-3-319-53076-5_6.

174. M. D. Godlevskyi, A. A. Goloskokova, A. A. Chipizhenko. (2017). Medium-term planning information technology for quality improvment of the software development process based on the CMMI model. *Вісник Нац. техн. ун-ту "ХПІ" : зб. наук. пр. Сер.: Системний аналіз, управління та інформаційні технології*. № 51 (1272). С. 32-37.

175. M.Colli, U.Berger, M.Bockholt, O.Madsen, C.Møller, B. Vejrum Wæhrens. (2019). A maturity assessment approach for conceiving context-specific roadmaps in the Industry 4.0 era. *Annual Reviews in Control,* Volume *48*, P. 165–177, DOI:10.1016/j.arcontrol.2019.06.001.

176. Albert Albers, Bartosz Gladysz, Tobias Pinner, Viktoriia Butenko, Tobias Stürmlinger. (2016) Procedure for defining the system of objectives in the initial phase of an Industry 4.0 project focusing on intelligent quality control systems. *Procedia CIRP*, Vol. 52, P. 262–267, DOI:10.1016/j.procir.2016.07.067.

177. Yevsieiev V., Bronnikov A. (2020). Analysis of the CMMI model application for solving the tasks of CPPS control processes automation development. *The IV th International scientific and practical conference «Actual Trends of Modern Scientific Research»,* P.128–132.

178. ISO/IEC/IEEE 12207:2017 Systems and software engineering — Software life cycle processes. [Electronic resource]. URL: https://www.iso.org/standard/63712.html. (Дата звернення: 15.06.2020).

179. Невлюдов И.Ш., Евсеев В.В., Бортникова В.О. (2102). Актуальность создания систем автоматизированного проектирования технического задания на разработку программных продуктов для сложных корпоративных информационных систем технологической подготовки производства. *XII Международная научная конференция «Физические процессы и поля технических и биологических объектов». Материалы конференции. КНУ имени Михаила Остроградського,* С.152–153.

180. Невлюдов И.Ш., Андрусевич А.А., Евсеев В.В. (2010). Анализ жизненного цикла разработки программного обеспечения для корпоративных ин-формационных систем. *Восточно-европейский журнал передовых технологий*, Вып.*6/8*(48), С.25–27.

181. Невлюдов И.Ш., Евсеев В.В., Бортникова В.О. (2011). Модели жизненного цикла программного обеспечения при разработке корпоративных информационных систем технологической подготовки производства. *Вестник Нац. техн. ун-та "ХПИ" : сб. науч. тр. Темат. вып.:Новые решения в современных технологіях,* № *2*, С. 94–101.

182. Yevsieiev V., Bronnikov A. (2020). Information systems development methodologies application analysis for cyber-physical production systems development. *III International scientific-practical conference "Theory, science and practice",* P. 398–401, DOI: 10.46299/ISG.2020.II.III.

183. B. Clark and R. Madachy. (2015). Software Cost Estimation Metrics Manual for Defense Systems, Software Metrics Inc. *Haymarket*, VA, 2015. P.253. ISBN 978-0-9904787-0-6.

184. Евсеев В. В. (2011). Применение программных метрик кода на раннем этапе жизненного цикла программного обеспечения. *Восточно-европейский журнал передовых технологий*. Вып. № *1/2* (49). С.19–21.

185. Невлюдов І.Ш., Євсєєв В.В., Милютина С.С., Бортникова В.О. (2012). Анализ моделей расчета трудоемкости программного продукта при разработке КИС ТПП. *Восточно-европейский журнал передовых технологий*. Вып.*6*. №2(60), С.21–24.

186. Невлюдов И.Ш., Евсеев В.В., Бортникова В.О. (2011). Анализ применимости математических моделей COCOMO при разработке современных корпоративно - информационных систем технологической подготовки производства. *XI Международная научная конференция «Физические процессы и поля технических и биологических объектов». Материалы конференции. КНУ имени Михаила Остроградского.* С. 147–148.

187. Евсеев В.В., Бортникова В.О. (2011). Анализ программных метрик при проектировании информационно-компьютерных систем технологии производства. *15 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке». Сб. материалов форума.* С.152–153.

188. Евсеев В.В., Бортникова В.О. (2012). Анализ языков высокого уровня программирования применяемых для разработки корпоративно-информационых систем технологической подготовки производства. *16 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке». Сб. материалов форума.* Том № *2*. С.142–143.

189. Евсеев В.В., Бортникова В.О. (2013). Параметрическая модель декомпозиции структурированного языка SQL для решения задач расчета трудоемкости. *17 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке». Сб. материалов форума.* Том № *2*. С. 119–120.

190. Теорія систем керування: підручник / В.І. Корнієнко, О.Ю. Гусєв, О.В. Герасіна, В.П. Щокін; М-во освіти і науки України, Нац. гірн. ун-т. – Дніпро: НГУ, 2017. – 497 с. ISBN 978-966-350-650-0

191. Цветков В.Я. Основы теории сложных систем. 1-е изд. ЛАНЬ. 2019. С.152. ISBN 978-5-8114-3509-8.

192. ISO 10303-235:2019 Industrial automation systems and integration — Product data representation and exchange — Part 235: Application protocol: Engineering properties and materials information. [Electronic resource]. URL:https://www.iso.org/ru/ standard/74409.html. (Дата звернення: 5.08.2020).

193. ISO 18828-3:2017 Industrial automation systems and integration — Standardized procedures for production systems engineering — Part 3: Information flows in production planning processes. [Electronic resource]. URL:https://www.iso.org/standard/ 68533.html. (Дата звернення: 5.08.2020).

194. ISO 18828-5:2019 Industrial automation systems and integration — Standardized procedures for production systems engineering — Part 5: Manufacturing change management. [Electronic resource]. URL: https://www.iso.org/standard/ 69231.html. (Дата звернення: 5.08.2020).

195. Nevliudov I., Yevsieiev V., Maksymova S., Filippenko I. (2020). Development of an architectural-logical model to automate the management of the process of creating complex cyber-physical industrial systems. *Eastern-European Journal of Enterprise Technologies*. Vol *4*. No 3(106). C.44–52. DOI: 10.15587/1729- 4061.2020.210761.

196. Євсєєв В.В., Максимова С.С.(2020). Технологія процесу керування розробкою кібер-фізичних виробничих систем. *Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки».* Том *31*(70). № 6. С.57–63.

197. Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л. Алгебра. Языки. Программирования. –К. «Наукова думка».1974. С. 328.

198. Горлушкина Н.Н. (2016). Системный анализ и моделирование информационных процессов и систем. *СПб: Университет ИТМО*, С.120.

199. Nevliudov I., Yevsieiev V., Omarov M., Bronnikov A., Liashenko V. (2020). Method of Algorithms for Cyber-Physical Production Systems Functioning Synthesis. *International Journal of Emerging Trends in Engineering Research (IJETER)*, Volume *8*. No.10. PP. 7465–7473. DOI:10.30534/ijeter/2020/ 1278102020.

200. Pedro Daniel Urbina Coronado, Roby Lynn, Wafa Louhichi, Mahmoud Parto, Ethan Wescoat&Thomas Kurfess. (2018) Part data integration in the Shop Floor Digital Twin: Mobile and cloud technologies to enable a manufacturing execution system. *Journal of Manufacturing Systems*. Volume *48*, Part C, P.25–33. DOI: 10.1016/j.jmsy.2018.02.002.

201. Yevsieiev V., Miliutina S., Kollesnyk K. (2015). Software development Life Cycle Model . *Warsaw University of techology, Instiyte of Design Fundamenals XXIII Polish-Ukrainin conference CAD in MACHINERY DESIGN (CADMD 2015)*, P.19–20.

202. Невлюдов И.Ш., Евсеев В.В., Бортникова В.О. (2013). Разработка модели жизненного цикла проектирования корпоративных информационных систем технологической подготовки. *Международная научно-практическая конференция «Математическое моделирование процессов в экономике и управлении инновационными проектами (ММП-2013)»*, С. 139–140.

203. Євсєєв В.В., Демська А.І. (2017). Розробка моделі життєвого циклу розробки програмних продукту та програмних модулів для КІС ТПВ. *Технология приборостроения.* №*1*. С.12–16.

204. Невлюдов И.Ш., Евсеев В.В., Бортникова В.О. (2013). Математическая модель расчета трудоемкости и стоимости программного продукта. *Proceedings XXIII International Conference "New Leading technologies in Machine Building"*. P.24.

205. Невлюдов И.Ш., Евсеев В.В., Бортникова В.О. (2011) Информационная модель автоматизированной системы проектирования корпоративно-информационных систем технологической подготовки производства на ранней стадии разработки технического задания. *Перша Всеукраїнська науково-практична конференція «Актуальні проблеми створення електронних засобів промислових автоматизованих систем»*. С. 18–21.

206. Евсеев В.В., Бортникова В.О. (2013). Параметрическая модель декомпозиции структурированного языка SQL. *Сучасні проблеми радіотехніки та телекомунікацій «РТ–2013»: матеріали 9-ої міжнар. молодіжної на-ук.-техн. конф.* С.328.

207. Невлюдов И.Ш., Евсеев В.В., Милютина С.С. (2015). Разработка графов принадлежности элементов языков программирования высокого уровня. *Материалы 4-й Международной научно-технической конференции «Информационные системы и технологи ИСТ-2015».* С. 88–89.

208. Невлюдов И.Ш., Евсеев В.В., Милютина С.С. (2015). Параметрическая модель области видимости памяти для языков объектно-ориентированного программирования. *XIV Міжнародна науково-технічна конференція "Фізичні процеси та поля технічних і біологічних об'єктів". Материалы конференции. КНУ имени Михаила Остроградского*. С. 120.

209. Невлюдов І.Ш., Євсєєв В.В., Милютина С.С., Бортникова В.О. (2012). Разработка графа параметрической зависимости для КИС ТПП на базе языков высокого уровня программирования. *Вестник Нац. техн. ун-та "ХПИ":сб. науч. тр. Темат. вып.: Новые решения в современных технологиях.* № *66* (972). С. 67–73.

210. Nevlyudov I., Yevsieiev V., Miliutina S. (2014). Program Project Development Life Cycle Model. *Комп'ютерні системи проектування теорія і практика*. № *808*. С. 26–30.

211. Nevlyudov I., Yevsieiev V., Miliutina S., Kolesnyk K. (2015). High – Level Programming Language Decomposition Parametric Model. *Machine Dynamics Research, Warsaw University of Technology*. Vol. *39*. No 1. P.81–91.

212. Nevlyudov I., Yevsieiev V., Miliutina S., Kollesnyk K. (2017). Object semantic model for life cycle model "Jamp". *CAD in Machinery Design. Implementation and Educational Issues. 25 Proceedings of Polish- Ukrainian Conference (CADMD'2017).* P. 31–32.

213. Yevsieiev V. (2018). Visual objects interaction mathematical presentation to solve the problem of software design automation for computer information systems of technological production preparation. *Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація»*. № *1*(31). С. 24–31

214. Yevsieiev V. (2018). Conceptual scheme and basic concepts graphic representation of software and modules visual elements description in CIS TPP design automation problem solution. *Наукові нотатки. Міжвузівський збірник (за галузями знань «Технічні науки»).* Випуск *61*. С. 40–47.

215. Yevsieiev V. (2018). Visual components formal description development for the automated design of software products and modules for computer-integrated production technological preparation systems. *Вчені записки таврійського національного університету імені В.І. Вернадського Серія: Технічні науки.* Том *29* (68) № 1 Частина 1. С.143–147, DOI: 10.31474/2075-4272-2018-1-31-24-31.

216. Невлюдов И.Ш., Андрусевич А.А., Евсеев В.В., Милютина С.С., Замирец Я.О. (2014). Формализация объектно-орентированных языков програмирования. *Технология приборостроения.* №*3*. С.11–17.

217. Плотникова З.В., Евсеев В.В. (2009). Метод нисходящего анализа с прогнозируемым выбором альтернатив для констекстно – зависимых граматик. *Восточно-европейский журнал передовых технологий*. Вып. *4/11*(40). С.11–13.

119. The structure of the administration shell: trilateral perspective from France, Italy and Germany. [Electronic resource]. URL: https://www.de.digital/DIGITAL/Redaktion/EN/Publikation/the-structure-of-the-administration-shell.pdf?__blob=publicationFile&v=3. (Дата звернення: 02.05.2020).

218. ISO/IEC 27001:2013. Information technology -Security techniques - Information security management systems – Requirements. [Electronic resource]. URL:https://www.iso.org/standard/54534.html. (Дата звернення: 02.09.2019).

219. Невлюдов І., Євсєєв В., Демська А. (2018). Розробка синтаксичної та семантичної моделі мови визначення і опису даних предметної області. *Manufacturing & Mechatronic Systems 2018: Proceedings of IIst International Conference (M&MS 2018)*. P. 48–53.

220. В.А. Морозова, В.И. Паутов.(2017). Представление знаний в экспертных системах. *Екатеренбург.Изд.-во Урал. Ун-та.*. С – 120. ISBN 978-5-7996-2037-0.

221. Євсєєв В.В., Невлюдов І.Ш., Мілютіна С.С. Автоматизизована система нормування «НОРМА». *Свідоцтво про реєстрацію авторського права на твір №57667 від 17.12.2014.*

222. Nevliudov I., Yevsieiev V., J. H. Baker, Ahmad M. A., Lyashenko V. (2021). Development of a cyber design modeling declarative language for cyber physical production systems. *Journal of Mathematical and Computational Science.* No.*1*. P.520–542, DOI:10.28919/jmcs/5152

223. Осипов Д. (2014) Delphi. Программирование для Windows, OS X, iOS и Android. *БХВ-Петербург.*С.464.

224. Джон Шарп. (2018). Microsoft Visual C#. *Питер Пресс*. С.848. ISBN: 978-5-496-02372-6.

225. Симонов Денис. Руководство по языку SQL СУБД Firebird 3.0:Firebird 3.0.4. [Electronic resource]. URL:https://firebirdsql.org /file/documentation/ reference_manuals/firebird-language-reference-30-rus.pdf. (Дата звернення: 02.09.2018).

226. Yevsieiev V., Bronnikov A. (2020). Development of databases interconnection “essences” information model for cyber-physical production systems additive cyber design creation automation. *Збірник наукових праць національного університету кораблебудування ім. адмірала Макарова*. №*3*(481). P. 56–62. DOI: 10.15589/znp2020.3(481).7.

227. Yevsieiev V.(2017). Program code automated system development at early stage of software life cycle. *Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація»*. №1(30). С.69–77. DOI: 10.31474/2075-4272-2018-1-31-24-31.

228. Nevliudov,I., Yevsieiev, V., Demska,N., Novoselov, S. (2020). Development of a software module for operational dispatch control of production based on cyber-physical control systems. *Innovative Technologies and Scientific Solutions for Industries*. No.*4*(14), P.155–168. DOI:10.30837/ITSSI.2020.14.155.

229. Мілютіна С.С., Невлюдов І.Ш., Євсєєв В.В. Модуль для голосового управління роботом РМ-01. *Свідоцтво про реєстрацію авторського права на твір №57666 від 17.12.2014.*

230. Гурін Л.А., Невлюдов І.Ш., Євсєєв В.В. Програма для програмування та віддаленого управління мобільним роботом «Programming robots». *Свідоцтво про реєстрацію авторського права на твір №59439 від 24.04.2015.*

231. Невлюдов І.Ш., Євсєєв В.В., Бортникова В.О. Розробка програмного модуля для автоматизованого проектування технологічного процесу виготовлення мікроелектромеханічних акселерометрів. *Системи управління, навігації та зв’язку. Збірник наукових прац*. (2015). Випуск 3(35). С. 107–112.

232. Бортнікова В.О., Невлюдов І.Ш., Євсєєв В.В. Комп’ютерна програма «Автоматизована система проектування технологічного процесу виготовлення акселерометрів «АсCAM» («АсCAM»). *Свідоцтво про реєстрацію авторського права на твір № 65348 від 16.05.2016.*

233. Голіков М.О., Невлюдов І.Ш., Євсєєв В.В., Функендорф А.О. «Модуль автоматизованого проектування технологічних схем складання роботів «Max-CAM»». *Свідоцтво про реєстрацію авторського права на твір № 74619 від 13.11.2017.*

234. Невлюдов І.Ш., Євсєєв В.В., Бортнікова В.О., Чала О.О. «Автоматизація комп'ютероного зору та обробки відеопотоку для мобільних роботів». *Свідоцтво про реєстрацію авторського права на твір № 80306 від 16.07.18.*

235. Невлюдов И.Ш., Евсеев В.В., Андрусевич А.А., Усков С.С. (2010). Некоторые аспекты разработки баз данных и внедрения САПР «Норма». *Восточно-европейский журнал передовых технологий*, Вып. 6/8(48). С. 21–24.

236. Голіков М.О., Невлюдов І.Ш., Євсєєв В.В., Функендорф А.О. «Модуль автоматизованого проектування конструкції роботів «Max - Robotics»». *Свідоцтво про реєстрацію авторського права на твір № 74642 від 13.11.2017.*

237. Горячевська Д.В., Невлюдов І.Ш., Євсєєв В.В., Горячевська І.В. Комп'ютерна програма «Програма для визначення синхронного контролю температурних режимів плат на виробництві «QUAcontrol»». *Свідоцтво про реєстрацію авторського права на твір №59980 від 4.06.2015.*

238. Євсєєв В.В**.** «Автоматизована система проектування програмного забезпечення для корпоративно-інформаційних систем технологічної підготовки виробництва «CAD-Programming Code»». *Свідоцтво про реєстрацію авторського права на твір № 74576 від 09.11.2017.*

ДОДАТОК А

Список публікацій здобувача за темою дисертації та відомості про апробацію результатів дисертації

А.1 Список публікацій здобувача за темою дисертації

Наукові праці, в яких опубліковані основні наукові результати дисертації:

1. **Yevsieiev V.** Visual components formal description development for the automated design of software products and modules for computer-integrated production technological preparation systems. *Вчені записки таврійського національного університету імені В.І. Вернадського Серія: Технічні науки.* 2018. Том 29 (68) № 1 Частина 1. С.143–147. DOI: 10.31474/2075-4272-2018-1-31-24-31.

2. **Евсеев В. В.** Применение программных метрик кода на раннем этапе жизненного цикла программного обеспечения. *Восточно-европейский журнал передовых технологий*. 2011. Вып. № 1/2 (49). С.19–21.

3. **Yevsieiev V.** Conceptual scheme and basic concepts graphic representation of software and modules visual elements description in CIS TPP design automation problem solution. *Наукові нотатки. Міжвузівський збірник (за галузями знань «Технічні науки»).* 2018. Випуск 61. С. 40–47.

4. **Yevsieiev V.** Visual objects interaction mathematical presentation to solve the problem of software design automation for computer information systems of technological production preparation. *Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація»* 2018. № 1(31). С. 24–31.

5. **Yevsieiev V.** Program code automated system development at early stage of software life cycle. *Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація»*. 2017. №1(30). С.69–77. DOI: 10.31474/2075-4272-2018-1-31-24-31.

6. Невлюдов І.Ш., **Євсєєв В.В.**, Демська А.І. Розробка моделі життєвого циклу розробки програмних продукту та програмних модулів для КІС ТПВ. *Технология приборостроения.* 2017. №1. С.12–16.

7. Nevlyudov I., **Yevsieiev V.**, Miliutina S., Kolesnyk K. High – Level Programming Language Decomposition Parametric Model. *Machine Dynamics Research, Warsaw University of Technology*. 2015. Vol. 39. No 1. P.81–91.

8. Nevlyudov I., **Yevsieiev V.**, Miliutina S. Program Project Development Life Cycle Model. *Комп'ютерні системи проектування теорія і практика*. 2014. № 808. С. 26–30.

9. Невлюдов И.Ш., Андрусевич А.А., **Евсеев В.В.**, Милютина С.С., Замирец Я.О. Формализация объектно-орентированных языков програмирования. *Технология приборостроения*. 2014. №3. С.11–17.

10. Невлюдов І.Ш., **Євсєєв В.В.**, Милютина С.С., Бортникова В.О. Анализ моделей расчета трудоемкости программного продукта при разработке КИС ТПП. *Восточно-европейский журнал передовых технологий*. 2012. Вып.6. №2(60). С.21–24.

11. Невлюдов И.Ш., Андрусевич А.А., **Евсеев В.В.** Анализ жизненного цикла разработки программного обеспечения для корпоративных информационных систем. *Восточно-европейский журнал передовых технологий*. 2010. Вып.6/8(48). С.25–27.

12. **Євсєєв В.В.**, Андрусевич А.О., Власенков Д.П. Аналіз концепції Industry 4.0 в технології IIOT. *Технология приборостроения*. 2020, №1. С.64–68.

13. Плотникова З.В., **Евсеев В.В.** Метод нисходящего анализа с прогнозируемым выбором альтернатив для констекстно – зависимых граматик. *Восточно-европейский журнал передовых технологий*. 2009. Вып. 4/11(40). С.11–13.

14. Nevliudov I., **Yevsieiev V.**, Maksymova S., Filippenko I. Development of an architectural-logical model to automate the management of the process of creating complex cyber-physical industrial systems. *Eastern-European Journal of*

Enterprise Technologies. Vol 4. No 3(106). C.44–52. DOI: 10.15587/1729-4061.2020.210761.

15. **Євсєєв В.В.**, Максимова С.С. Технологія процесу керування розробкою кібер-фізичних виробничих систем. *Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки».* 2020. Том 31(70). № 6, С.57–63.

16. Nevliudov I., **Yevsieiev V.**, Omarov M., Bronnikov A., Liashenko V.. Method of Algorithms for Cyber-Physical Production Systems Functioning Synthesis. *International Journal of Emerging Trends in Engineering Research (IJETER)*, 2020. Volume 8. No.10. PP.7465–7473. DOI:10.30534/ijeter/2020/1278102020.

17. Nevliudov I., **Yevsieiev V.**, J. H. Baker, Ahmad M. A., Lyashenko V. Development of a cyber design modeling declarative language for cyber physical production systems. *Journal of Mathematical and Computational Science*. 2021. No.1. PP.520–542. DOI:10.28919/jmcs/5152.

18. Nevliudov,I., **Yevsieiev, V.**, Demska,N., Novoselov, S. Development of a software module for operational dispatch control of production based on cyber-physical control systems. *Innovative Technologies and Scientific Solutions for Industries*. 2020. No.4(14), P.155–168. DOI:10.30837/ITSSI.2020.14.155.

19. Невлюдов І.Ш., **Євсєєв В.В.**, Бортникова В.О. Розробка програмного модуля для автоматизованого проектування технологічного процесу виготовлення мікроелектромеханічних акселерометрів. *Системи управління, навігації та зв'язку. Збірник наукових прац*. (2015). Випуск 3(35). С 107–112.

20. Невлюдов І.Ш., **Євсєєв В.В.**, Милютина С.С., Бортникова В.О. Разработка графа параметрической зависимости для КИС ТПП на базе языков высокого уровня программирования. *Вестник Нац. техн. ун-та "ХПИ":сб. науч. тр. Темат. вып.: Новые решения в современных технологиях.* 2012. № 66 (972). С. 67–73.

21. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Модели жизненного

цикла программного обеспечения при разработке корпоративных информационных систем технологической подготовки производства. *Вестник Нац. техн. ун-та "ХПИ" : сб. науч. тр. Темат. вып.:Новые решения в современных технологиях. – Харьков : НТУ "ХПИ"*. 2011. № 2. С. 94–101.

22. **Yevsieiev V.**, Bronnikov A. Development of databases interconnection “essences” information model for cyber-physical production systems additive cyber design creation automation. *Збірник наукових праць національного університету кораблебудування ім. адмірала Макарова*. 2020. №3(481) . PP. 56–62. DOI: 10.15589/znp2020.3(481).7.

А.2 Свідоцтво про реєстрацію авторського права за темою дисертації:

1. Мілютіна С.С., Невлюдов І.Ш., **Євсєєв В.В.** Модуль для голосового управління роботом РМ-01. *Свідоцтво про реєстрацію авторського права на твір №57666 від 17.12.2014.*

2. **Євсєєв В.В.**, Невлюдов І.Ш., Мілютіна С.С. Автоматизизована система нормування «НОРМА». *Свідоцтво про реєстрацію авторського права на твір №57667 від 17.12.2014.*

3. Гурін Л.А., Невлюдов І.Ш., **Євсєєв В.В.** Програма для програмування та віддаленого управління мобільним роботом «Programming robots». *Свідоцтво про реєстрацію авторського права на твір №59439 від 24.04.2015.*

4. Горячевська Д.В., Невлюдов І.Ш., **Євсєєв В.В.**, Горячевська І.В. Комп’ютерна програма «Програма для визначення синхронного контролю температурних режимів плат на виробництві «QUAcontrol»». *Свідоцтво про реєстрацію авторського права на твір №59980 від 4.06.2015.*

5. Бортнікова В.О., Невлюдов І.Ш., **Євсєєв В.В.** Комп’ютерна програма «Автоматизована система проектування технологічного процесу виготовлення акселерометрів «АсСАМ» («АсСАМ»). *Свідоцтво про реєстрацію авторського права на твір № 65348 від 16.05.2016.*

6. Голіков М.О., Невлюдов І.Ш., **Євсєєв В.В.**, Функендорф А.О. «Модуль автоматизованого проектування конструкції роботів «Max - Robotics»». *Свідоцтво про реєстрацію авторського права на твір № 74642 від 13.11.2017.*

7. Голіков М.О., Невлюдов І.Ш., **Євсєєв В.В.**, Функендорф А.О. «Модуль автоматизованого проектування технологічних схем складання роботів «Max-CAM»». *Свідоцтво про реєстрацію авторського права на твір № 74619 від 13.11.2017.*

8. **Євсєєв В.В.** «Автоматизована система проектування програмного забезпечення для корпоративно-інформаційних систем технологічної підготовки виробництва «CAD-Programming Code»». *Свідоцтво про реєстрацію авторського права на твір № 74576 від 09.11.2017.*

9. Невлюдов І.Ш., **Євсєєв В.В.**, Бортнікова В.О., Чала О.О. «Автоматизація комп'ютероного зору та обробки відеопотоку для мобільних роботів». *Свідоцтво про реєстрацію авторського права на твір № 80306 від 16.07.18.*

А.3 Наукові праці, які засвідчують апробацію матеріалів дисертації:

1. Невлюдов І., **Євсєєв В.**, Демська А. Розробка синтаксичної та семантичної моделі мови визначення і опису даних предметної області. *Manufacturing & Mechatronic Systems 2018: Proceedings of IIst International Conference (M&MS 2018)*. (Kharkiv, 25–26 October 2018) P. 48–53.

2. **Yevsieiev V.**, Bronnikov A. Analysis of architectural models for representing the integration of cyber-physical production systems hierarchical levels. *Manufacturing & Mechatronic Systems 2020: Proceedings of IVth International Conference (M&MS 2020)*. (Kharkiv, 22–23 October 2020). P:17–19.

3. **Yevsieiev V.**, Miliutina S., Kollesnyk K. Software development Life Cycle Model . *Warsaw University of techology, Instiyte of Design Fundamenals*

XXIII Polish-Ukrainin conference CAD in MACHINERY DESIGN (CADMD 2015). (Polish, Bochnia, 9–10 October 2015). P.19–20.

4. Nevlyudov I., **Yevsieiev V.**, Miliutina S. Structured Language SQL Parametric Model Development. *CAD in Machinery Design. Implementation and Educational Issues. Proceedings of Ukrainian-Polish Conference (CADMD'2016)*. (Lviv, 21–22 October 2016). P. 49–50.

5. Nevlyudov I., **Yevsieiev V.**, Miliutina S., Kollesnyk K. Object semantic model for life cycle model "Jamp". *CAD in Machinery Design. Implementation and Educational Issues. 25 Proceedings of Polish- Ukrainian Conference (CADMD'2017).* (Polish, Bielsko Biala, 20–21 October 2017). P. 31–32.

6. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Анализ применимости математических моделей COCOMO при разработке современных корпоративно - информационных систем технологической подготовки производства. *XI Международная научная конференция «Физические процессы и поля технических и биологических объектов»*. (Кременчук, 4–6 листопада 2011). Материалы конференции. КНУ имени Михаила Остроградского. С. 147–148.

7. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Актуальность создания систем автоматизированного проектирования технического задания на разработку программных продуктов для сложных корпоративных информационных систем технологической подготовки производства. *XII Международная научная конференция «Физические процессы и поля технических и биологических объектов»*. (Кременчук, 2–4 листопада 2012). Материалы конференции. КНУ имени Михаила Остроградского. С. 152–153.

8. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С. Параметрическая модель области видимости памяти для языков объектно-ориентированного программирования. *XIV Міжнародна науково-технічна конференція "Фізичні процеси та поля технічних і біологічних об'єктів"*. (Кременчук, 6–8 листопада 2015). Материалы конференции. КНУ имени Михаила Остроградского. С. 120.

9. Невлюдов И.Ш., **Евсеев В.В.**, Милютина С.С. Разработка графов принадлежности элементов языков программирования высокого уровня. *Материалы 4-й Международной научно-технической конференции «Информационные системы и технологи ИСТ-2015».* (Харьков, 21–27 сентября 2015). С. 88–89.

10. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Разработка модели жизненного цикла проектирования корпоративных информационных систем технологической подготовки. *Международная научно-практическая конференция «Математическое моделирование процессов в экономике и управлении инновационными проектами (ММП-2013)»*. (Алушта, 9–15 сентября 2013). ХНУРЭ. С. 139–140.

11. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Математическая модель расчета трудоемкости и стоимости программного продукта. *Proceedings XXIII International Conference "New Leading technologies in Machine Building"*. (Rybachie, 2–8 september 2013). P.24.

12. **Евсеев В.В.**, Бортникова В.О. Параметрическая модель декомпозиции структурированного языка SQL. *Сучасні проблеми радіотехніки та телекомунікацій «РТ–2013»: матеріали 9-ої міжнар. молодіжної на-ук.-техн. конф.* (Севастополь, 22–26 квітня 2013). СевНТУ. С.328.

13. **Евсеев В.В.**, Бортникова В.О. Анализ программных метрик при проектировании информационно-компьютерных систем технологии производства. *15 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке».* (Харьков, 18–20 апреля 2011). Сб. материалов форума. ХНУРЭ. С.152–153.

14. **Евсеев В.В.**, Бортникова В.О. Анализ языков высокого уровня программирования применяемых для разработки корпоративно-информационых систем технологической подготовки производства. *16 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке»*. (Харьков, 17–19 апреля 2012). Сб. материалов форума. Том № 2.

ХНУРЭ. С.142–143.

15. **Евсеев В.В.**, Бортникова В.О. Параметрическая модель декомпозиции структурированного языка SQL для решения задач расчета трудоемкости. *17 международный молодежный форум «Радиоэлектроника и молодежь в XXI веке»*. (Харьков, 22–24 апреля 2013). Сб. материалов форума. Том № 2. ХНУРЭ. С. 119–120.

16. **Yevsieiev V.**, Jijavadze O. Analysis architectural model of Industry 4.0 (RAMI 4.0). *XXIV Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті».* (Харків, 7–9 квітня 2020). Зб. матеріалів форуму. Т. 2. С.93–94.

17. Невлюдов И.Ш., **Евсеев В.В.**, Бортникова В.О. Информационная модель автоматизированной системы проектирования корпоративно-информационных систем технологической подготовки производства на ранней стадии разработки технического задания. *Перша Всеукраїнська науково-практична конференція «Актуальні проблеми створення електронних засобів промислових автоматизованих систем»* (м. Сєвєродонецьк, 25–26 жовтня 2011). С. 18–21.

18. **Yevsieiev V.**, Bronnikov A. Analysis of the cyber-physical production systems implementation impact to achieve the goals of lean production. *The IIth International scientific and practical conference «Development of scientific and practical approaches in the era of globalization»* (USA, Boston, 28–30 September. 2020). P.221–226. DOI:10.46299/ISG.2020.II.II.

19. **Yevsieiev V.**, Bronnikov A. Analysis of the CMMI model application for solving the tasks of CPPS control processes automation development. *The IV th International scientific and practical conference «Actual Trends of Modern Scientific Research»* (Germany, Munich, 11–13 October 2020). P.128–132.

20. **Yevsieiev V.**, Bronnikov A. Information systems development methodologies application analysis for cyber-physical production systems development. *III International scientific-practical conference "Theory, science and practice"* (Japan, Tokyo, 5–8 October 2020). P. 398–401. DOI:

10.46299/ISG.2020.II.III.

21. **Yevsieiev V.**, Bronnikov A. Analysis of the multi-agent systems application to solve the problem of cyberphysical production systems development. *The IV th International scientific and practical conference «Integration of scientific bases into practice»*. (Sweden, Stockholm, 12–16 October 2020). P.459 – 462. DOI:10.46299/ISG.2020.IV.

22. **Yevsieiev V.**, Bronnikov A. Structural model of a cyber-physical production system based on multi-agent systems analysis. *Матеріали VII Міжнародної науково-технічної Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами»*, (Київ, 26 листопада 2020). С.312–313.

23. **Yevsieiev V.**, Bronnikov A. Comlexity development analysis of cyber-physical production systems for smart manufacturing. *The X th International scientific and practical conference «Trends in the development of modern scientific thought»* (Canada, Vancouver, 23–26 Nov. 2020). P.699–703. DOI:10.46299/ISG.2020.II.X.

ДОДАТОК Б

Акти про впровадження результатів дисертаційної роботи

61064 м. Харків, вул. Воложанівська, 43 кв. 7
Телефон: +38 (057) 768-09-02
E-mail: info@ukrintech.com.ua

UKRINTECH
UKRAINIAN INNOVATIVE TECHNOLOGIES

UA153348510000000002600811191
Поточний рахунок в АТ КБ «ПРИВАТБАНК»:
UA333515330000026003052222090

Код ЄДРПОУ 37461045, ІПН № 374610420350

ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ
«НАУКОВО-ВИРОБНИЧЕ ПІДПРИЄМСТВО «УКРІНТЕХ»

Витяг з реєстру платників ПДВ №1420354500038

АКТ

про впровадження у виробництво результатів дисертаційної роботи
на здобуття вченого ступеня доктора технічних наук

Комісія у складі:

Голова: Демченко С. В. – директора Товариства з обмеженою відповідальністю «Науково-виробниче підприємство «УКРІНТЕХ»»;

Члени комісії:

Вікторова О. В. – керівник випробувальної лабораторії ТОВ «НВП «УКРІНТЕХ»»;

Боцман О. С. – провідний інженер-електронник ТОВ «НВП «УКРІНТЕХ»»;

склала даний Акт про те, що наукові результати:

- архітектурно-логічна модель і методи декомпозиції кібер-фізичного керування процесами в складних організаційно-технічних об'єктах;

- взаємопов'язані методи процесів керування організаційно-технічним об'єктом, як логічно узгоджених послідовностей прийняття рішень, що формуються на базі апаратних засобів на фізичному рівні;

– програмне забезпечення «Система розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом».

були використані при виконанні договору № 20/80-П від 08.08.2018р. «Ремонт преса гідравлічного ДА2238Б» з ГП «НАЕК «Енергоатом», ОП «Атоменергомаш» ЗНСОіТ.

Результати впровадження довели, що запропоновані методи та моделі дозволили удосконалити процес керування, що дало можливість підвищити продуктивність на 1,2% та ритмічність 1,8% на місяць.

Комісія підтверджує працездатність запропонованих моделей та методів, які реалізовані в «Система розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом», а також той факт, що їх впровадження дозволило суттєво зменшити час розробки кібернетичної складової на 18%, а фізичної складової на 11%.

Економічний ефект від впровадження не розраховувався у зв'язку з науковими призначеннями результатів.

Акт складено для представлення в спеціалізовану вчену раду та не є основою для виплати винагороди за впровадження та інших авторських винагород.

ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ «НАУКОВО ВИРОБНИЧЕ ПІДПРИЄМСТВО «УКРІНТЕХ» №37461045 місто Харків

Голова комісії _____ С.В. Демченко

Члени комісії _____ О.В. Вікторова

_____ О.С. Боцман

«23» 10 2019

АКТ

про впровадження у виробництво результатів дисертаційної роботи
на здобуття вченого ступеня доктора технічних наук
Євсєєва Владислава В'ячеславовича

Комісія у складі голови: замісника начальника управління обчислювальної техніки, інформатики та зв'язку – начальник відділу системного забезпечення Харитонова О.Б.; та членів комісії: начальника бюро системного програмного забезпечення та управління технологічними процесам Баштового В.С. та провідного інженера-програміста бюро системного програмного забезпечення та управління технологічними процесами Живіло А.А. даним актом засвідчуємо, що основні наукові положення розроблені Євсєєвим В.В. та викладені в його докторській дисертації, а саме:

- архітектурно-логічна модель і методи декомпозиції кібер-фізичного керування процесами в складних організаційно-технічних об'єктах;
- технології безперервного процесу керування організаційно-технічним об'єктом;
- модель формалізації кібернетичної складової організаційно-технічного об'єкта, яка дозволяє представити HMI у вигляді багаторівневої абстракції існування взаємопов'язаних GUI елементів;
- синтаксична і семантична моделі декларативної мови визначення і маніпулювання даними;

які були реалізовані у програмному засобі «Системи розробки кібернетичної складової для автоматизації процесів керування організаційно-технічним виробничим об'єктом», що був впроваджений у виробництво та дозволив одержані наступні результати: підвищення продуктивності на 5% та ритмічності 3%.

Економічний ефект від впровадження не розраховувався у зв'язку з науковими призначеннями результатів.

Акт складено для представлення в спеціалізовану вчену раду та не є основою для виплати винагороди за впровадження та інших авторських винагород.

Голова комісії _____ Харитонов О.Б.

Члени комісії _____ Баштовой В.С.

_____ Живіло А.А.

«15» 05 2019

«ЗАТВЕРДЖУЮ»
Проректор з наукової роботи та
міжнародної діяльності
Національного університету
«Запорізька політехніка»
_____ В.В. Наумик
«01» 01 2020 р.

АКТ

з впровадження результатів дисертаційної роботи
Євсєєва Владислава В'ячеславовича

Комісія у складі голови: декана факультету радіоелектроніки та телекомунікацій – к.т.н., доц. Кабака Владислава Семеновича та членів комісії: завідувача кафедри «Інформаційні технології електронних засобів» – д.т.н., доц. Шило Галина Миколаївни, доцента кафедри «Інформаційні технології електронних засобів» – к.т.н., доц. Фарафонова Олексія Юрійовича, доцента кафедри «Інформаційні технології електронних засобів» – к.т.н., доц. Фурманової Наталії Іванівни, склали акт про провадження результатів дисертаційної роботи Євсєєва В.В. у навчальний процес Національного університету «Запорізька політехніка».

Склад впровадження:

– синтаксична і семантична моделі декларативної мови визначення і маніпулювання даними, яка дозволяє автоматизувати процес розробки кібернетичної складової для керування складними організаційно-технічними виробничим об'єктом;

– програмне забезпечення «Система розробки кібернетичної складової для автоматизації процесів курування організаційно-технічним виробничим об'єктом».

Комісія встановила, що результати дисертаційної роботи Євсєєва В.В. були впроваджені у навчальний процес кафедри «Інформаційні технології електронних засобів» Національного університету «Запорізька політехніка» при проведенні лекційних та лабораторних занять з дисципліни «Системи управління технологічними процесами (SCADA системи)» та «Математичне моделювання та системний аналіз» для спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології.

Голова комісії _____ В.С. Кабак
Члени комісії _____ Г.М. Шило
_____ О.Ю. Фарафонов
_____ Н.І. Фурманова

«Затверджую»
Проректор з науково-педагогічної та
методичної роботи Кременчуцького
національного університету
ім. М. Остроградського
проф. Костін В.В.
_____ 2020 р.

АКТ
з впровадження результатів дисертаційної роботи
Євсєєва Владислава В'ячеславовича

Комісія у складі голови: декана факультету електроніки та комп'ютерної інженерії – к.т.н., доц. Мосьпан Владислава Олександровича та членів комісії: завідувача кафедри електронних апаратів – к.т.н., доц. Фомовської Олени Владиславівни, доцента кафедри електронних апаратів – к.т.н., доц. Юрка Олексія Олексійовича, склали акт про провадження результатів дисертаційної роботи Євсєєва В. В. на тему «Методи та моделі кібер-фізичного керування процесами в організаційно-технічних виробничих об'єктах», подану на здобуття наукового ступеню доктора технічних наук за спеціальністю 05.13.07 – автоматизація процесів керування у навчальний процес Кременчуцького національного університету ім. М. Остроградського.

Склад впровадження:

– технології процесу управління організаційно-технічним об'єктом в режимі реального часу, які дають можливість отримати алгоритми функціонування кожного рівня автоматизованої системи керування виробництвом та дозволяють уявити структуру у вигляді узгоджених інформаційних блоків;

– програмне забезпечення для автоматизації процесів керування організаційно-технічним виробничим об'єктом.

Комісія встановила, що результати дисертаційної роботи Євсєєва В. В. були впроваджені у навчальний процес кафедри електронних апаратів Кременчуцького національного університету імені М. Остроградського при проведенні лекцій, у методичних вказівках щодо виконання лабораторних та практичних занять з курсів «Основи технології електронних пристроїв», «Основи САПР електронних пристроїв», «Автоматизація обробки інформації, методи та апаратура» для спеціальності 171 Електроніка.

Голова комісії _____ В. О. Мосьпан

Члени комісії _____ О. В. Фомовська

_____ О. О. Юрко

«ЗАТВЕРДЖУЮ»
Директор навчально-наукового інституту
комп'ютерної фізики та енергетики
Харківського національного університету
імені В.Н. Каразіна
_____ к.т.н. Гарячевська І.В.
«__» _____ 2020 р.

АКТ
з впровадження результатів дисертаційної роботи
Євсєєва Владислава В'ячеславовича

Комісія у складі: заступник директора з наукової роботи ННІ КФЕ – к.ф-м.н., доц. Марущенко Ілля Миколайович, заступник директора з навчальної роботи ННІ КФЕ – к.ф-м.н., доц. Лісіна Ольга Юлієвна, завідувача кафедри Інформаційних технологій в фізико-енергетичних системах – к.ф-м.н., доц. Сухов Руслан Володимирович, склали акт про провадження результатів дисертаційної роботи Євсєєва В.В. у навчальний процес навчально-наукового інституту комп'ютерної фізики та енергетики Харківського національного університету імені В. Н. Каразіна.

Склад впровадження:

– метод уявлення структурних системних моделей кібер-фізичного керування, який дозволяє формалізувати інформаційні блоки алгоритму функціонування в системні моделі;

– модель життєвого циклу процесу управління організаційно-технічним об'єктом;

– програмне забезпечення для автоматизації процесів керування організаційно-технічним виробничим об'єктом.

Комісія встановила, що результати дисертаційної робот Євсєєва В.В. були впроваджені у навчальний процес кафедри інформаційних технологій в фізико-енергетичних системах навчально-наукового інституту комп'ютерної фізики та енергетики Харківського національного університету імені В. Н. Каразіна при проведенні лекційних, лабораторних та практичних занять з курсів Моделювання процесів перетворення енергії для студентів 4 курсу бакалаврату та Практикум з програмування в фізичному експерименті для студентів 1 курсу магістратури спеціальності 105 Прикладна фізика та наноматеріали.

Голова комісії _____ І.М. Марущенко

Члени комісії _____ О.Ю. Лісіна
_____ Р.В. Сухов

Додаток В

Параметрична модель формалізації та граф приналежності параметрів «Container Solution»

$$L =< Gs, C, Sc, Bt, Con, Ex, Des, Op, Fun, \Pr > , \quad \text{(B.1)}$$

де *Gs* – загальний синтаксис;

C – константи;

Sc – область видимості;

Bt – типи даних;

Con – перетворення;

Ex – вирази;

Des – опис;

Op – оператори;

Fun – функції;

Pr – предпроцесори.

В рамках даних досліджень загальний синтаксис можна представити у вигляді параметричної моделі:

$$\mathrm{Gs} = < Com, Ind, key >, \quad \text{(B.2)}$$

де *Com* – коментарі, що задаються початковим символом /* і закриваючим символом */, або символом // – якщо коментар задається і закінчується на тій же строчці;

Ind – ідентифікатори, послідовність букв и цифр довільної довжини, з урахуванням того, що перший символ обов'язково повинен бути буквою чи знаком «_» і початкова буква ідентифікатора є різною в різних регістрах.

key – ключові слова, зарезервовані ідентифікатори, які можна використовувати у вигляді ключових слів і не як по-іншому.

Існує кілька видів констант: цілі константи, явно задані довгі константи і символьні константи. Уявімо константи у вигляді такої параметричної моделі:

$$C =< Ic, Elc, Cc, Fps, Ec, Cdu, Li >, \qquad \text{(В.3)}$$

де Ic – цілі константи;

Elc – явно задані довгі константи;

Cc – символьні константи;

Fps – константи з плаваючою точкою;

Ec – перелічувальні константи; імена, описані як перелічувальні;

Cdu – константи, описані користувачем; об’єкт любого типу, який може бути визначений як такий, що має постійне значення по всій області видимості його імені;

Li – строки, послідовність символів, укладена в подвійні лапки “...”.

Цілі константи (Ic) – складаються з послідовності цифр і діляться на восьмеричні, десяткові і шістнадцяткові.

Параметрична модель цілих констант:

$$Ic =< ic_8, ic_{10}, ic_{16} > , \qquad \text{(В.4)}$$

де ic_8 – восьмерична константа, починається з 0 (за винятком цифр 8 та 9);

ic_{10} – десятерична константа, починається з 0 до 9;

ic_{16} – шістнадцятерична константа, починається з 0x, включає в себе цифри від 0 до 9 і букви від A до F.

Явно задані довгі константи (Elc) – десяткова, восьмерична або шістнадцятерична константа, за якою безпосередньо стоїть l або L, вважається довгою константою.

Параметрична модель явно заданих констант представлена у формулі:

$$Elc =< elc_8, elc_{10}, elc_{16} > \quad \text{(В.5)}$$

де els_8 – явно задана восьмерична константа;

els_{10} – явно задана десяткова константа;

els_{16} – явно задана шістнадцятерична константа.

Символьні константи (Cc) – складаються із символу, укладеного в одиничні лапки – ’x’.

Константи х плаваючою точкою (Fpc) – складаються із цілої частини, десяткової точки, мантиси, символу e чи E та цілого показника ступеня. Параметрична модель константи з плаваючою точкою представлена в (В.6).

$$fpc =< ip, dp, ma, se, exT >, \quad \text{(В.6)}$$

де ip – ціла частина;

dp – десяткова точка ;

ma – мантиса;

se – символ e чи E;

exT – цілий показник ступеню.

Існує чотири види областей видимості: локальна, файл, програма і клас. Параметрична модель області видимості представлена в (В.7):

$$\text{Sc} =< loc, f, prog, cl >, \quad \text{(В.7)}$$

де *loc* – локальна; ім’я, описане локально в цьому блоці, може використовуватися тільки в ньому, після місця опису і в охоплених блоках;

f – файл; ім’я, описане поза любого блока чи класу, може використовуватись в файлі, де воно описано, після місця опису;

prog – програма; ім’я, описане в файлі, може використовуватись в будь-якому іншому файлі проекту шляхом декларації зовнішнього опису цього імені;

cl – клас; ім’я члена класу локальне для його класу і може використовуватися тільки у функції-члені цього класу, після застосованої до об'єкта його класу операції «.» (точка) чи після застосованої до покажчика на об'єкт його класу операції - >.

Класи пам’яті (Cl) – визначають область видимості змінної, а також як довга змінна, що знаходиться в пам’яті. Є два описових класів пам’яті: автоматичний і статичний. Параметрична модель для класів пам’яті представлена нижче.

$$Cl =< au, st >, \tag{В.8}$$

де *au* – автоматичний клас пам’яті; автоматичні об’єкти локальні для кожного виклику блока і скидається по виходу з нього;

st – статичний клас пам’яті; статичні об’єкти існують і зберігають своє значення протягом виконання всієї програми.

Параметрична модель типів даних представлена в (В.9):

$$Bt =< ch,in,fl,db,vd,\ dt > \quad , \tag{В.9}$$

де *ch* – базовий тип даних char;

in – базовий тип даних int;

fl – базовий тип даних float;

db – базовий тип даних double;

vd – тип даних void, визначає пусту множину значень.

Також, крім базових типів існують похідні типи (Bt), сконструйовані з основних типів. Перетворювачі (Con) – певні операції, які призводять до зміни значення операнда від одного типу до іншого.

$$Con =< sw, fd, fli, poi, uns, acon, conp, conl >, \qquad \text{(B.10)}$$

де *sw* – перетворення символів або цілого;

fd – перетворення між числами одинарної і подвійної точності;

fli – перетворення плаваючих значень в цілий тип і перетворення цілочисельного значення в плаваючий тип;

poi – перетворення цілого типу і покажчика;

uns – перетворення цілого беззнакового;

acon – арифметичне перетворення;

conp – перетворення покажчиків;

conl – перетворення посилань.

Вираз *Ex* – конструкція мови програмування, результатом обчислення якої є «істина» або «брехня». Його параметрична модель:

$$Ex =< bex, ao, uo, mo, oitpclm, so, ro, eo, ba, beo, booo, la, lo, co, as, como, oo > ,$$

де *bex* – основні вирази;

ao – адитивні операції;

uo – унарні операції;

mo – мультиплікативні операції;

oitpclm – операції непрямого звернення через покажчик на член класу;

so – операції зсуву;

ro – операція відносин (порівняння);

eo – операція рівності;

ba – операція побітова «І» &;

beo – операція побітова виключає «АБО» ^;

booo – операція побітова, що включає «АБО» |;

la – операція логічне «І» &&;

lo – операція логічна «АБО» ||;

co – умовна операція;

as – операція присвоювання;

como – операція кома;

oo – перевантажені операції.

Основні вирази (*bex*) – включають в себе операції найвищого пріоритету: прямого звернення до члена класу . , непрямого звернення до члена класу →, специфікація доступу ::, індексування [ ] і виклику функції ( ). Вони групуються зліва направо.

Параметрична модель основних виразів представлена нижче:

$$bex =< drclm, irclm, sps, ind, func >, \quad \text{(В.11)}$$

де *drclm* – пряме звернення до члена класу;

irclm – непряме звернення до члена класу;

spa – специфікації доступу;

ind – індексування;

func – виклик функції.

Унарна операція (*uo*) – це операція над одним операндом. Параметрична модель унарних операцій представлена нижче:

$$uo =< ind, pto, oinc, ore, lon, avoo, prikipsk, prediposd, siz, nw, del > \text{(В.12)},$$

де *ind* – унарна операція * (непряме звернення);

pto – унарна операція &(покажчик на об’єкт);

oinc – унарна операція +;

ore – унарна операція;

lon – операція логічного заперечення;

avoo – операція додаткового значення операнди до одиниці.

prikipsk – прединкремент чи постінкремент ++;

prediposd – прединкремент чи постінкремент --;

siz – розмір;

nw – створення динамічного об’єкта;

del – видалення динамічного об’єкта.

Мультиплікативні операції (*mo*) – операції *, / і %, виконують звичайні арифметичні перетворення. Параметрична модель мультиплікативної операції представлена нижче:

$$mo =< mom, \bmod, \bmod m >, \qquad \text{(В.12)}$$

де *mom* – бінарна операція, що визначає множення;

mod – бінарна операція, що визначає ділення;

$\bmod m$ – бінарна операція ділення по модулю.

Параметрична модель операцій непрямого звернення через вказівник на член класу представлена нижче:

$$oitpclm =< prcochy, ccochy >,$$

де *prcochy* – операція прямого непрямого звернення через вказівник;

ccochy – операція непрямого звернення через вказівник.

Адитивні операції + і – (*ao*) -, виконують звичайні арифметичні перетворення. Кожна операція має деякі додаткові можливості, пов'язані з типами.

Параметрична модель адитивних операцій представлена нижче:

$$ao =< bpl, bmi >,$$

де bpl – операція бінарного плюса;

bmi – операція бінарного мінуса.

Операції зсуву << і >> (so), обидві виконують звичайні арифметичні перетворення над своїми операндами. Параметрична модель мультиплікативних операцій представлена нижче.

$$so =< sol, sop >, \quad \text{(В.13)}$$

де sol – операція зсуву вліво;

sop – операція зсуву вправо.

Операції співвідношень (ro) – це операції порівняння. Параметрична модель операцій відношень представлена у (А.14).

$$ro =< omch, obch, ochr, obrr >, \quad \text{(В.14)}$$

де $omch$ – операція менше чим;

$obch$ – операція менше чи дорівнює;

$ochr$ –операція більше чим;

$obrr$ –операція більше чи дорівнює.

Операція рівності (eo) == і != аналогічні операціям порівняння, за винятком їх низького пріоритету. Вказівник може порівнюватись з нулем.

Параметрична модель операцій рівності представлена в (В.15).

$$eo =< eor, eonr > , \quad \text{(В.15)}$$

де *eor* – операція дорівнює;

eonr – операція не дорівнює.

Існує безліч операцій привласнення (*as*). Параметрична модель операцій привласнення представлена нижче:

$$as =< asr, aspr, asmr, asdr, asprr, asbr, asrmr, asir, astr, asvilr, asviprr >,$$

де *asr* – операція простого привласнення;

aspr – операція привласнення з множенням;

asmr – операція привласнення з діленням;

asdr – операція привласнення з діленням по модулю;

asprr –операція привласнення з сумою;

asbr – операція привласнення з різницею ;

asrmr – операція привласнення з побітовим «И»;

asir – операція привласнення з побітовим, що не містить «ИЛИ»;

astr – операція привласнення з побітовим, що містить «ИЛИ»;

asvilr – операція привласнення з лівим зсувом;

asviprr – операція привласнення з правим зсувом.

Опис (*Des*) використовують для визначення інтерпретації, що задається кожному ідентифікатору. Представимо опис у вигляді параметричної моделі (В.16).

$$Des =< scisp, tsp, ldes, descl, ini, nt, tdef, destl, desa >, \qquad (B.16)$$

де *sclsp* – специфікатори класу пам’яті;

tsp – специфікатори типу;

ldes – описувачі;

descl – опис класів;

ini – ініціалізація;

nt – ім’я типів;

$tdef$ – визначення типу;

$destl$ – опис перерахування;

$desa$ – опис asm.

Специфікатори класу пам’яті ($sclsp$) – опис, що використовує специфікатори auto, static, register, extern служать визначенням, так як викликають резервування відповідного об’єму пам’яті. Якщо опис extern не є визначенням, то десь ще повинно бути визначення для даних ідентифікаторів.

Параметрична модель операторів представлена в (В.17):

$$sclsp =< au, st, ex, re >, \qquad \text{(В.17)}$$

де au – специфікатор auto, використовується тільки в оголошеннях змінних з локальним контекстом;

st – специфікатор static, може використовуватись в оголошеннях функцій та змінних з контекстом файлу та локальним контекстом для визначенням внутрішнього типа компоновки;

ex – специфікатор extern, може бути використаним в оголошеннях функцій та змінних з контекстом файлу і з локальним контекстом для визначення зовнішнього типу компоновки;

re – специфікатор register, допустимий тільки в оголошеннях локальних змінних та параметрів функцій.

Специфікатори типу (tsp) змінюють значення базового типа, для того щоб він більш точно відповідав своєму призначенню в програмі.

Параметрична модель специфікаторів типів представлена в (В.18):

$$tsp =< ptsp, stsp >, \qquad \text{(В.18)}$$

де *ptsp* – простий специфікатор типу;

stsp – складний специфікатор типу.

Параметрична модель опису класів (*descl*) представлена нижче:

$$decl =< stch, funch, prcl, vifun, konct, pre, destr, fr, funo, stru, uni, bp, vlcl >,$$

де *stch* – статичні члени;

gunch – функції-члени;

prcl – похідні класи;

vifun – віртуальні функції;

konct – конструктори;

pre – перетворення;

destr – деструктори;

snch – видимість імен членів;

fr – друзі;

funo – функції-операції;

stru – структури;

uin – об’єднання;

bp – бітові поля;

vlcl – вкладені класи.

Оператори (op) – будь-який вираз, що закінчується точкою з комою (;). Параметрична модель операторів представлена в (В.19):

$$op =< oex, comst, tro, lst, cst, flw, osw, og, exo, rst, deso >, \qquad (B.19)$$

де *oex* – оператори-вирази;

comst – складовий оператор (блок);

tro – оператор переходу;

lst – позначений оператор;

cst – умовний оператор;

flw – оператор циклу while;

fldw – оператор циклу do-while;

fif – оператор циклу for;

osw – оператор-перемикач switch;

og – оператор розриву break;

exo – оператор продовження continue;

rst – оператор повернення return;

deso – оператор опису.

Один оператор може займати одну або більше строк. Два або більша кількість операторів можуть бути розташованими на одній строчці. Оператори, що керують порядком виконання (if, if-else, switch, while, do-while, for), можуть бути вкладені один в інший.

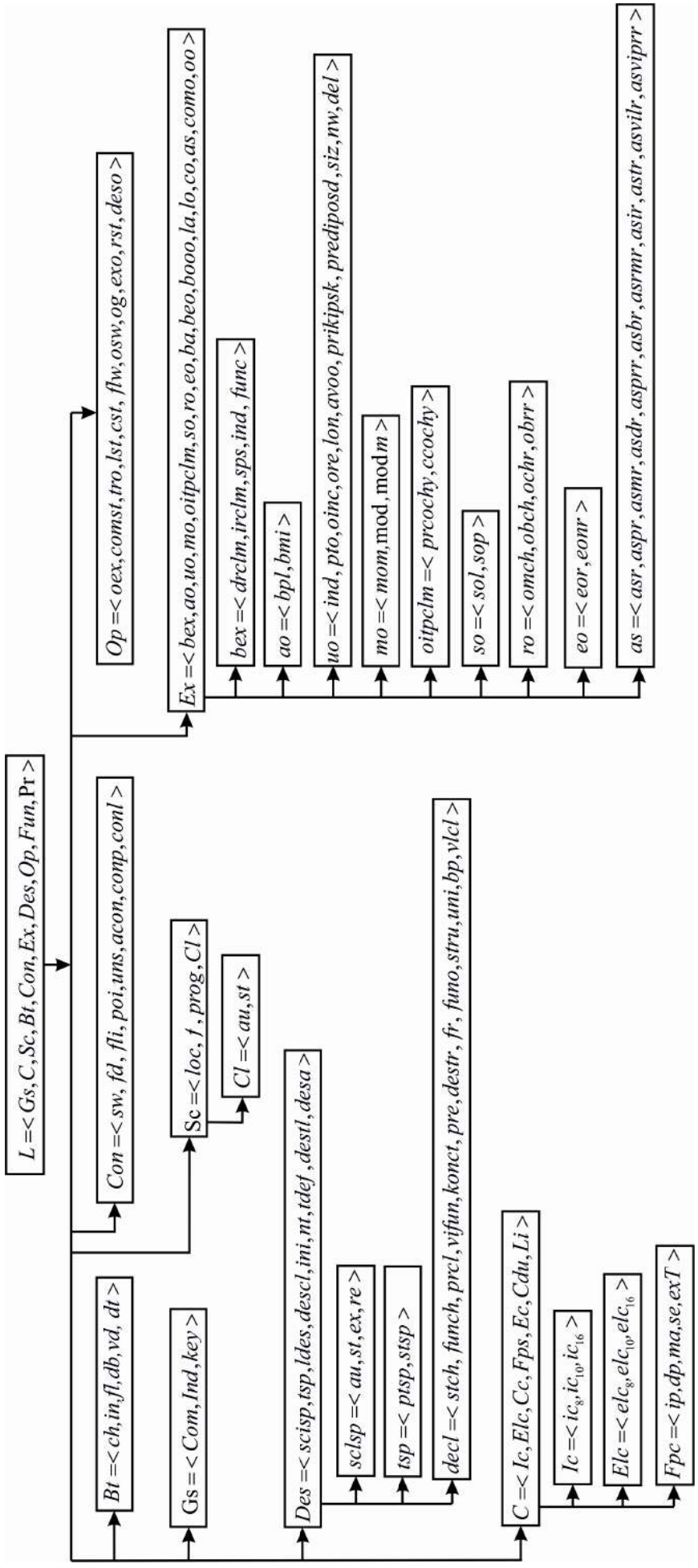


Рисунок В.1 - Граф приналежності параметрів у параметричній моделі

Додаток Г

Фрагмент реалізації «Container Solution»

Таблица Г.1 – Фрагмент реалізацій Lingustic Variable

	Lingustic Variable	Container Solution	
		cod (Delphi XE3)	cod (Microsoft Visual Studio 2019)
1	2	3	4
Реалізація HMI на базі подій GUI елементів			

1	form_display	if Form_name.ShowModal = mrOk then
		begin
		Form_name.Visible:=true;
		end;
			private void Button1_Click(Object sender, EventArgs e
			)
			{
			Form1 myForm = new Form1();
			myForm.Show();
			}

2	panel_show	Panel_name.visible:=true;	Panelname.Visible = true;
3	dialog_window	if MessageDlg('your text',mtConfirmation, [mbYes,mbNo],0)=mrYes then ShowMessage('your text') else ShowMessage('your text');	var result = MessageBox.Show(‘message’, ‘caption’, MessageBoxButtons.YesNo,MessageBoxIcon.Question); if (result == DialogResult.Yes { MessageBox.Show (‘message’)} else MessageBox.Show (‘message’);
4	switching_ panels	Panel_name1.Visible:=True; Panel_name2.Visible:=False;	Panelname.Visible = true; Panelname.Visible = false;
5	message_box	ShowMessage('your text');	MessageBox.Show (‘message’)
6	button_message _box	MessageBox(Form_name.Handle, 'your text',' Error', MB_ICONERROR + MB_OK);	var result = MessageBox.Show(‘message’, ‘Error, MessageBoxButtons.YesNo, MessageBoxIcon.Error);
7	close	Close;	Form.Close();

8	run_file	ShellExecute(form_name.Handle,nil,'help.hlp',nil,nil,SW_SHOW);	ShellExecute shellExecute = new ShellExecute();
			shellExecute.Path = file;
			shellExecute.Execute();

Продовження таблиці Г.1

1	2	3	4

9	open_file_memo	if OpenTextFileDialog_name.Execute then
		Memo_name.Lines.LoadFromFile(OpenTextFileDialog1.FileName);
			if (openFileDialog1.ShowDialog() ==
			DialogResult.Cancel)
			return;
			string filename = openFileDialog1.FileName;
			string fileText =
			System.IO.File.ReadAllText(filename);
			textBox1.Text = fileText;

10	start_timer	Timer_name.Enabled := true;	myTimer.Enabled = true;

11	filling_progress_
	bar
		if countTime < 20 then
		begin
		label_name.Caption := loadingMessage[countTime];
		inc(countTime);
		ProgressBar_name.Position:=countTime;
		Application.ProcessMessages;
		end
		else
		begin
		countTime:=0;
		end;
			if (countTime < 20)
			{label_name.Caption = loadingMessage[countTime];}
			countTime++;
			ProgressBar_name.Positin = countTime;
			Application.DoEvents();}
			else countTime =0;

12	delete_message	if Key = 46 then
		if (Application.MessageBox('Your text?',' Delete',
		MB_OKCANCEL+MB_ICONEXCLAMATION) = mrOk)
		then
		(Sender as TDBGrid).DataSource.DataSet.Delete;
			if (MessageBox.Show("Do you want to delete this row
			?", "Delete", MessageBoxButtons.YesNo) ==
			DialogResult.Yes)
			{
			dataGridView1.Rows.RemoveAt(dataGridView1.Selec
			tedRows[0].Index);
			sAdapter.Update(sTable);}

Продовження таблиці Г.1

1	2	3	4

13	choice_combobox	var
		s:string;
		begin
		s:=ComboBox_vibor1.Items[ComboBox_vibor1.ItemIndex];
		if s=' condition text 1'
		then
		begin
		// execution of the selected condition 1
		end ;
		if s=' condition text 2'
		then
		begin
		// execution of the selected condition 2
		end
			int selectedIndex = comboBox1.SelectedIndex;
			Object selectedItem = comboBox1.SelectedItem;
			if (comboBox1.SelectedIndex ==1) // execution of the selected condition 1
			else if(comboBox1.SelectedIndex ==2) // execution of the selected condition 2
			....

14	upload_picture	if OpenDialog_name.Execute then
		OpenDialog_name.DefaultExt:=GraphicExtension(TBitmap);
		Image_name.Picture.LoadFromFile(OpenDialog_name.FileName);
			var dialog = new OpenFileDialog();
			dialog.Title = "Open Image";
			dialog.Filter = "bmp files (*.bmp)|*.bmp";
			if (dialog.ShowDialog() == DialogResult.OK)
			{
			var PictureBox1 = new PictureBox;
			PictureBox1.Image(dialog.FileName);
			}
			dialog.Dispose();

Продовження таблиці Г.1

1	2	3	4

15	tabcontrol	if tabcontrol_name.Tabindex=0 then
		Panel_name1.Visible:=true
		else
		Panel1.Visible:=false;
		if tabcontrol_name.Tabindex=1 then
		Panel_name2.Visible:=true
		else
		Panel_name2.Visible:=false;
			if (TabControl1.SelectedIndex = 0){
			panelName1.Visible = true;}
			else panelName1.Visible = false;
			if (TabControl1.SelectedIndex = 1){
			panelName2.Visible = true;}
			else panelName2.Visible = false;

16	tree	if TreeView1.Selected <> nil then
		begin
		TreeView1.Items.AddChild(TreeView1.Selected,
		Edit1.Text);
		TreeView1.Selected.Expanded := True;
		end
		else
		TreeView1.Items.Add(nil, Edit1.Text);
		end;
			if (mySelectedNode != null &&
			mySelectedNode.Parent != null)
			{
			treeView1.SelectedNode = mySelectedNode;
			treeView1.LabelEdit = true;
			treeView1.SelectedNode.Expand();
			if(!mySelectedNode.IsEditing)
			{
			mySelectedNode.BeginEdit();
			}

Реалізація роботи з БД			

17	join db	if IBDatabase_ BD_name.Connected then
		IBDatabase_ BD_name.Close;
		p:= ExtractFilePath(Application.ExeName);
		p:= p + 'bd\BD_name.GDB';
		if not EMBEDDED then
		p:= 'localhost:' + p;
		IBDatabase_ name.DatabaseName := p;
		IBDatabase_ name.Open;
			string connString = @"Data
			Source="+datasource+";Initial Catalog="
			+database+";Persist Security
			Info=True;User
			ID="+username+";Password="+password;
			
			SqlConnection conn = new
			SqlConnection(connString);
			
			return conn;

Продовження таблиці Г.1

1	2	3	4

18	Open db	IBDatabase_ name.Open;
		IBTransaction_ name.StartTransaction;
			SqlConnection conn;
			conn = new SqlConnection(constr);
			conn.Open();

19	Table_open	IBDataSet_BD_name.Open;	DataSet dataSet = new DataSet("Suppliers"); adapter.Fill(dataSet);
20	request_select_ all	Query_name.SQL.Clear; Query_name.SQL.Add('select * from table_1, table_2'); Query_name.SQL.Add('where table_1.id= table_2.id'); Query_name.Active:=true;	SqlCommand cmd = new SqlCommand("SELECT * FROM table_1,table_2 WHERE table_1.id= table_2.id'", connection)) using (SqlDataAdapter adapter = new SqlDataAdapter(cmd)) { adapter.Fill(table);}
22	update_table	DBGrid_name.Refresh;	datagridview1.Refresh();

23	add_new_entry	if
		messageDlg('Add?',mtConfirmation,[mbYes,mbNo],0)=mrYe
		s
		then if (Edit_name.Text<>'')then
		IBDataSet_name.Open;
		IBDataSet_name.Append;
		IBDataSet_name['bd_field']:=Edit_name.Text;
		IBDataSet_name.Post;
		Edit_name.Text:='';
			var result = MessageBox.Show(“Add?”, “Add new record”,
			MessageBoxButtons.YesNo,
			MessageBoxIcon.Question);
			if (result == DialogResult.Yes)
			{ DataSet dataSet = new DataSet("Suppliers");
			DataTable dt = new DataTable("Table");
			dt.Columns.Add(new DataColumn("id",typeof(int)));
			dt.Columns.Add(new DataColumn("bd_field",
			typeof(string)));
			DataRow dr = dt.NewRow();
			dr["id"] = 123;
			dr["bd_field"] = Edit_name.Text;
			dt.Rows.Add(dr);
			ds.Tables.Add(dt);
			Edit_name.Text = “”;}

Продовження таблиці Г.1

1	2	3	4

24	delete_entry	if messageDlg('
		Delete?',mtConfirmation,[mbYes,mbNo],0)=mrYes
		then
		IBDataSet_name.Edit;
		IBDataSet_name.Delete;
		DBGrid_name.Refresh;
			var result = MessageBox.Show(“Delete?”, “Delete entry”,
			MessageBoxButtons.YesNo,
			MessageBoxIcon.Question);
			if (result == DialogResult.Yes)
			{ DataSet dataSet = new DataSet("Suppliers");
			ds.Tables[0].Rows[rowindex].Delete();
			ds.AcceptChanges();}

25	edit_entry	IBDataSet_name.Edit;
		if MessageDlg(' Edit?',mtConfirmation,
		[mbYes,mbNo],0)=mrYes
		IBDataSet_name.Edit;
			var result = MessageBox.Show(“Edit?”, “Edit entry”,
			MessageBoxButtons.YesNo,
			MessageBoxIcon.Question);
			if (result == DialogResult.Yes)
			{ foreach(DataRow dr in table.Rows)
			{
			if(dr["Product_id"] == 123)
			{
			dr["Product_name"] = "one";
			}
			}}

26	add_field_memo	IBDataSet_name.Edit;
		IBDataSet_name[' field'] := Memo1.Lines.Text;
		IBDataSet_name.Post;
			DataSet ds = new DataSet("Suppliers");
			DataTable dt = new DataTable("Table");
			DataRow dr in dt.Rows
			dr["Product_name"] = TextBox1.Text;
			ds.AcceptChanges();

Продовження таблиці Г.1

1	2	3	4
27	Select_window	if MessageDlg('your text',mtConfirmation, [mbYes,mbNo],0)=mrYes then ShowMessage('your text') else ShowMessage('your text');	var result = MessageBox.Show(“your text”, “Caption”, MessageBoxButtons.YesNo, MessageBoxIcon.Question); if (result == DialogResult.Yes) {string message = "Message"; MessageBox.Show(message);} else { string message = "Message"; MessageBox.Show(message);}

28	message_add
	_bd
		if messageDlg('your text',mtConfirmation,[mbYes,mbNo],0)=
		mrYes
		then if (Edit_add_name1.Text<>'')or
		(Edit_add_name2.Text<>'') then
		begin
		IBDataSet_name.Open;
		IBDataSet_name.Open;
		IBDataSet_name.Append;
		IBDataSet_name['field name]:=Edit_add_name1.Text;
		IBDataSet_ name ['field name ']:= Edit_add_name2.Text;
		IBDataSet_name;
		end
		else
		IBDataSet_name.Edit;
			var result = MessageBox.Show(“your text”, “Caption”,
			MessageBoxButtons.YesNo,
			MessageBoxIcon.Question);
			if (result == DialogResult.Yes)
			{
			DataTable table = new DataTable();
			DataColumn column;
			DataRow row;
			DataView view;
			row = table.NewRow();
			row["field name 1"] = TextBox1.Text;
			row["field name 1"] = TextBox2.Text;
			table.Rows.Add(row);
			view = new DataView(table);
			dataGrid1.DataSource = view;
			}

29	refresh_add	IBDataSet_name.Refresh;	ds.AcceptChanges();

Продовження таблиці Г.1

1	2	3	4
30	transaction_save	Form_name.IBTransaction_name.CommitRetaining; Close;	SqlConnection conn; ... conn.Close();

31	append_info
	_bd
		IBDataSet_name1.Append;
		IBDataSet_name1['name_field1']:=IBDataSet_name2['name_field2'];
		IBDataSet_name1.Post;
			DataRow row;
			row = table.NewRow();
			row["field name 1"] = TextBox1.Text;
			row["field name 1"] = TextBox2.Text;
			table.Rows.Add(row);

32	search_edit	var
		fre:boolean;
		begin
		if Edit_name.Text<>''then
		fre:=table_name.Locate('field',string(Edit_name.Text),
		[loCaseInsensitive,loPartialKey]);
		if fre<>true then
		MessageDlg('Your text',mtError,[mbOK],0)
		else
		form_name.Visible:=true
			bool fre;
			if (TextBox1.Text != “”) {
			foreach(DataRow dr in table.Rows)
			{
			if(dr["field”] == TextBox1.Text)
			{
			String s = dr["field”];
			s = s.ToLower();
			dr["field _name"] = s;
			}
			}
			if (fre)
			{string message = "Message";
			MessageBox.Show(message);}
			}
			else {
			base.OnVisibleChanged(e);
			this.Visible = true;
			}

Продовження таблиці Г.1

1	2	3	4

33	window_SQL	Query1.close;
		Query1.SQL.Clear;
		If Memo1.Lines[0] <> '' then
		Query1.SQL.Add(Memo1.Text)
		else
		begin
		messageDlg('No SQL was entered', mtError, [mbOK], 0);
		exit;
		end;
		try
		Query1.Open;
		except
		One: EDatabaseError do …..
		messageDlg(e.message, mtError,
		[mbOK],0);
		end;
		end
			conn.Close();
			string queryString = “”;
			if (TextBox1.Text == “”){
			MessageBox.Show(“No SQL was entered”);
			}
			else {
			string queryString = TextBox1.Text;
			DataSet dataSet = new DataSet();
			var connection = new SqlConnection("TODO:put connection string here");
			connection.Open();
			using(connection)
			{
			SqlDataAdapter adapter = new SqlDataAdapter(queryString, connection);
			adapter.Fill(dataSet);
			}
			}

34	access_check	Clear;
		Add('SERVER NAME=C:IBLOCALEXAMPLESEMPLOYEE.GDB');
		Add('SCHEMA CACHE=8');
		Add('OPEN MODE=READ/WRITE');
		Add('SQLPASSTHRU MODE=SHARED NOAUTOCOMMIT');
		Add('USER NAME=' + edit1.text);
		Add('PASSWORD=' + edit2.text);
			DataSet dataSet = new DataSet();
			String CON_ADDRESS = TextBox1.Text;
			String USER_NAME = TextBox2.Text;
			String PASSWORD = TextBox3.Text;
			var connection = new SqlConnection ADDRESS + USER + PASSWORD);
			connection.Open();
			using(connection)
			{
			}

Продовження таблиці Г.1

1	2	3	4
35	dropp_IBQuery	DataModule_zak.IBQuery_zak.Active:=False; DataModule_zak.IBQuery_zak.Active:=True;	connection.Close(); connection.Open();
36	foreign_key	DataSet['id_tabl1'] :=IBDataSet_add_operac['id_table1'];	dr_1["id_tabl1"] = dr_add_operac ["id_tabl1"];