УДК

КП

№ держреєстрації 0100V001722

Інв. №

Міністерство освіти України
Запорізький державний університет
(ЗДУ)
69063, м. Запоріжжя, вул. Жуковського, 66; тел. (0612)-64-45-46, факс (0612)-62-71-61, телекс 12-74-47 ОРИОН

ЗАТВЕРДЖУЮ
проректор з наукової роботи
та міжнародного співробітництва,
д-р. техн. наук, проф.

_____В. З. Грищак

ЗВІТ

ПРО НАУКОВО - ДОСЛІДНУ РОБОТУ

РОЗРОБКА МАТЕМАТИЧНИХ МОДЕЛЕЙ ТА МЕТОДІВ ОПИСУ І ВЗАЄМОДІЇ ЕЛЕМЕНТІВ ЄДИНОГО ІНФОРМАЦІЙНОГО ПРОСТОРУ В ІНТЕГРОВАНІЙ МЕРЕЖІ ВУЗІВ НА БАЗІ ПРИНЦИПІВ ДІАКОПТИКИ І АРХІТЕКТУР ТИПУ МАЙСТЕР - АГЕНТ

(заключний)

Керівник НДР
зав. каф. ММІТ,
д-р. техн. наук., проф. В.О. Толок

2001

Рукопис закінчено 15.11.2001.

ПЕРЕЛІК АВТОРІВ

Керівник НДР
зав. кафедрою
д-р. техн. наук., професор — В.О. Толок
(Вступ, розділи 1, 2, 3, 4, 5, 6, 7, висновки)

Відповідальний виконавець НДР
ст. наук. співр.
канд. техн. наук, .доцент — С.Ю. Борю
(Вступ, розділи 1, 2, 3, 4, 5, 6, 7, висновки)

ст. наук. співр.
канд. фіз. - мат. наук, доцент — В.А. Єрмолаєв
(Вступ, розділи 1, 2, 3, 4, 5, 6,7, висновки)

наук. співр. — Н. Г. Кеберле
(розділи 1,2,3,4,5,6,7)

мол. наук. співр. — С. Л. Плаксін
(розділи 2,3,4,5,6,7)

Реферат

Звіт НДР: 172 с., 116 джерел, 2 таблиць, 54 рисунка.

Об'єкт дослідження – інтегровані комп'ютерні мережі і системи, інтегровані розподілені інформаційні системи, інтелектуальні візуальні інтерфейси доступу до компонентів єдиного інформаційного простору, формальні моделі і методи побудови функціональних компонентів єдиного інформаційного простору, мультіагентські системи, архітектури типу CORBA і майстер - агент, динамічні співтовариства інтелектуальних інформаційних агентів, середовище інтернет і інтранет.

Мета роботи - розробка математичних моделей і методів опису і взаємодії єдиного інформаційного простору в інтегрованій мережі вузу на базі принципів диакоптики і архітектур типу майстер - агент.

Методи дослідження - математичне моделювання, загальна теорія систем.

Отримані результати і новизна – розроблені формальні принципи і методи взаємодії моделей функціональних об'єктів єдиного інформаційного простору.

Область застосування - наукові, навчальні і дослідницькі роботи.

МАКРОМОДЕЛЮВАННЯ, ІНТЕГРОВАНІ КОМП'ЮТЕРНІ МЕРЕЖІ, ІНТЕГРОВАНІ РОЗПОДІЛЕНІ ІНФОРМАЦІЙНІ СИСТЕМИ, МАТЕМАТИЧНІ МОДЕЛІ, ЄДИНИЙ ІНФОРМАЦІЙНИЙ ПРОСТІР, ДИНАМІЧНІ МУЛЬТИАГЕНТСЬКІ МОДЕЛІ, АРХІТЕКТУРИ МАЙСТЕР-АГЕНТ.

Зміст

ВСТУП

В даний час дослідники приділяють пильну увагу проблемам і методам, зв'язаним з моделюванням віртуальних і реальних підприємств [1]. Підвищена увага до цих проблем порозуміється як ростом процесів розподілу і віртуализації виробничих і бізнес процесів (гарним прикладом може служити стрімкий розвиток Інтернет - магазинів, інших засобів надання реальних послуг віртуальними методами, наприклад, електронної комерції й ін.), так і необхідністю додання існуючим моделям автоматизованого керування реальними підприємствами відсутніх у них динамізму. Як легко бачити, проблема в такій постановці відбиває в більшому ступені точку зору стороннього спостерігача або контрагента, взаємодіючого у віртуальним підприємством ззовні. Іншою не менш важливою проблемою є задача створення комфортабельного середовища мешкання особам, що діють усередині моделируемой динамічної системи - суб'єктам, що виконують визначені функції. Ця внутрішня точка зору на проблему має своєю головною метою усунення семантичних розривів між зрозумілими користувачу способами представлення інформації і формулювання задач і елементами інформаційних систем, що побудовані на базі формальних методів і розуміють формальним образом сформульовані впливи.

У сучасній бібліографії з проблем моделювання віртуальних підприємств можна знайти досить цікаві ідеї і методи створення подібних середовищ. Одним з напрямків, наприклад, є використання логіки одночасно виконуваних транзакций, що лежить в основі Системи Керування Віртуальними Підприємствами (СКВП) для моделювання і формальних висновків взаємодій у моделі віртуального підприємства [2]. Цей спосіб моделювання орієнтований на створення формального апарата опису і висновку взаємодій у тимчасових співтовариствах об'єктів, що утворять динамічну систему (віртуальне підприємство) для досягнення спільної мети. Заслуговують на увагу й інші методи функціонального моделювання реальних підприємств, побудовані на

базі апарата, що базується на поняттях ролі, політики і взаємодії. Одним з важливих прикладів реалізації такого підходу є ICRF [3]. Інтенсивно розвивається і напрямок, зв'язаний з побудовою формалізмів для інформаційних потоків. Ці формалізми моделюють взаємодії і взаємини в співтовариствах функціональних елементів віртуальних підприємств на базі парадигми агентів, що кооперуються. Прикладом побудови такого формалізму може служити [4].

У даної науково - дослідницькій роботі парадигма віртуального інформаційного простору є способом моделювання реального або віртуального підприємства за допомогою представлення цієї складної динамічної системи в термінах інтегрованої інформаційної системи [5], архітектура якої базується на стандарті CORBA [6]. Таке віртуальне інтегроване інформаційне середовище підприємства в роботі названо Єдиним Інформаційним Простором (ЄІП).

Для моделювання реальних процесів і функцій, що виконуються усередині підприємства, застосовуються моделі проекцій і їхніх функціональних компонентів ЄІП. Це дозволяє нам на додаток до функціональних моделей враховувати організаційну і геометричну топологію моделируемого підприємства і прозорим образом інтегрувати вже працюючі на підприємстві Інформаційні Системи (ИС). Другою відмінною рисою дослідження є спроба врахувати топологію функціональної проекції, що змінюється. Для побудови адаптивних моделей функціональних блоків, що еволюціонують, цього віртуального інформаційного простору застосовуються: диакоптичний підхід [7]; концепція взаємодіючих інтелектуальних агентів [8]; методи семантичного аналізу даних [9]; елементи теорії кінцевих автоматів [10].

Даний звіт містить результати виконання робіт зі створення Методики декомпозиції ЄІП на проекції, розробці моделей проекцій на базі архітектури ЄІП і розробці формальних моделей для реалізації функціональних компонентів та їх взаємодії у різних проекціях ЄІП.

Основна ідея, що лежить у фундаменті даного дослідження з побудови формальних моделей і методів взаємодії у межах Єдиного Інформаційного Простору (ЄІП) [1], була запропонована в ICDT-моделі Стратегій Бізнесу у Інтернет (Internet Business Strategies) [11]. Концепція ЄІП відрізняється від подібної ідеї Віртуального Інформаційного Простору в моделі ICDT, яка визначає простий канал для відображення і доступу до інформації. В рамках нашого дослідження під ЄІП розуміється віртуальний посередник, організований на верхньому рівні багатошарової ІС, що поєднує ієрархію розподілених, неоднорідних, взаємодіючих функціональних компонент (відділів) і розподілені неоднорідні інформаційні ресурси (ІС локального використання). Люди використовують ЄІП як модель Віртуальної Установи (ВУ) і спілкуються з нею за допомогою уніфікованого візуального Інтранет-інтерфейсу (УВІІ) [5]. Концепції ЄІП і УВІІ близькі за змістом до відомих підходів до розробки Населених Інформаційних Просторів [12]. В ЄІП мешкають активні функціональні компоненти (мультіагентні системи (МАС) та агенти - учасники), які займають відповідні організаційні клітинки на різних рівнях. З організаційної точки зору ці компоненти є віртуальними бізнес-об’єктами, що виконують бізнес-процеси як потоки робіт, в термінах, наприклад, середовища підприємства [13].

Особливістю пропонованого формального підходу є спроба промоделювати реальні функціональні компоненти ВНЗ і реальні процеси, що виконуються у ВНЗ, як бізнес-процеси в середовище ЄІП. Бізнес-процеси в свою чергу моделюються як процеси інформаційного обміну між різними типами користувачів-людей та різноманітними активними функціональними системами/компонентами, представленими як МАС/агенти, що мають відповідні ролі і розподілені в Інтернет. Середовища, архітектури і їх реалізації для моделювання бізнес-процесів і управління в рамках ВУ сьогодні набувають значного поширення та дослідження (наприклад, див. [4-5]). Але, та різноманітність процесів, з якою ми зустрічаємось в реальному житті, показує, що дуже важко моделювати їх більш-менш статичними засобами, наприклад,

такими, що пропонуються в ROOM, моделях ролей OOFRam [14], середовищі, заснованому на CTL [2], ICRF [3]. Парадигма використання агентів, на нашу думку, пропонує вихід із цього світу наперед заданих потоків робіт і специфікацій ролей. Даний підхід використовує метафору динамічних суспільств агентів, для того, щоб використовувати кращі засоби для моделювання внутрішньої динаміки Предметної Області. Цей підхід близький до підходу, використаного в середовищі RETSINA [15] для адаптивної взаємодії між командами агентів, що дозволяє вирішувати завдання по прийняттю рішень і управлінням інформацією. В рамках запропонованого підходу агенти є учасниками різних статичних МАС, які представляють постійно існуючі підрозділи ВУ. Підрозділи спілкуються між собою за допомогою так званих агентів-посередників (Proxy Agents), що діють як виконавці, але мають деякі зовнішні функції та додаткові шляхи комунікації. В свою чергу, посередники формують МАС ВУ більш високого рівня. На нижньому рівні кожний агент-учасник МАС підрозділу може бути розширений в підпорядковану МАС з тією ж архітектурою. Оскільки ці моделі підрозділів представляють функціональні відділи ВУ, вони були розроблені так, щоб "вміти" виконувати належні завдання стосовні своєї ролі. До таких завдань належать завдання отримання інформації, інтеграції, обміну і посередництва. Ролі агентів [16] є більш-менш статичні, оскільки агенти здатні виконати задані набори атомарних робіт. З другого боку, здатності агентів змінюються з часом, як змінюються їх накоплений досвід та обмеження, в яких вони працюють. Більш того, агенти в рамках МАС динамічно формують коаліції для спільного виконання тієї чи іншої роботи. Запропонований підхід використовує діакоптичне середовище МАС [16], модель виконання завдання суспільством агентів [17]. Розробка дизайну для користувачів - людей базується на концепції УВІІ [18].

Легко бачити, що проблема моделювання віртуальної або реальної установи засобами ЄІП, в якому мешкають МАС, має два аспекти. З одного боку, це проблема налагодження семантичних, і психологічних зв’язків між

виконавцями-людинами та їх штучними дублерами, а також між агентами. З другого боку, це проблема операційній інтероперабельності між неоднорідними розподіленими компонентами - тобто комунікації, спільної роботи і координації між цими учасниками. Проблема ускладнюється, якщо прийняти до уваги аспект еволюції штучних акторів (так само, як відповідних інформаційних ресурсів, представлених у різних моделях даних). Цій звіт розглядає також і питання семантичної інтероперабельності (представлення онтологій, спільне використання знань, тощо).

Даний звіт структурований наступним образом: розділ 1 розглядує підходи і концепції до побудови формальних моделей об'єктів і агентів для проекцій ЄІП, що є використаними впродовж виконання роботи; розділ 2 присвячений побудові архітектури розподіленої Інформаційної Системи (ІС) організації для представлення проекцій ЄІП; вкладом розділу 3 є розробка базових моделей і теоретичного апарату представлення активних функціональних об'єктів ЄІП; завдання розділу 4 – це презентація розроблених моделей і апарату взаємодії функціональних об'єктів ЄІП; розділ 5 присвячений розробленим моделям і методам формування агентами динамічних коаліцій для спільного використання завдань в ЄІП; вкладом розділу 6 є вирішення проблем семантичної інтероперабельності в суспільствах і коаліціях взаємодіючих агентів ЄІП; практичні питання використання розроблених рішень для моделювання і управління в галузях планування, дистанційного навчання, побудови інтелектуальних програмних систем для е-комерції розглядуються в заключном розділі 7.

Підставою для виконання даної науково - дослідної роботи є наказ ЗДУ № 60 від 25.02.1999 р.

1 Підхід до побудови формальних моделей об'єктів і агентів для проекцій єдиного інформаційного простору

У сучасній бібліографії з проблем моделювання віртуальних підприємств приводяться досить цікаві моделі і методи створення інтегрованих інформаційних середовищ. Одним з напрямків є використання логіки одночасно виконуються трансакцій, що лежить в основі Системи Керування Віртуальними Підприємствами (СУВП) для моделювання і формальних висновків взаємодій у моделі віртуального підприємства [2]. Цей спосіб моделювання орієнтований на створення формального апарата опису і висновку взаємодій у тимчасових співтовариствах об'єктів, що утворять динамічну систему (віртуальне підприємство) для досягнення якоїсь мети. Заслуговують на увагу й інші методи функціонального моделювання реальних підприємств, побудовані на базі апарата, що базується на поняттях ролі, політики і взаємодії. Одним з важливих прикладів реалізації такого підходу є ICRF [3]. Інтенсивно розвивається і напрямок, зв'язане з побудовою формалізмів для інформаційних потоків моделюючі взаємодії і взаємини в співтовариствах функціональних елементів віртуальних підприємств на базі парадигми агентів, що кооперуються. Прикладом побудови такого формалізму може служити [4].

У даної роботі для моделювання реальних процесів і функцій у ЄІП пропонуються адаптивні моделі функціональних компонентів, що еволюціонують, що представляють собою співтовариства інтелектуальних інформаційних агентів. Моделі вказаних співтовариств будуються на базі диакоптичного підходу [7], концепції взаємодіючих інтелектуальних агентів [8], методів семантичного аналізу даних [9] і елементів теорії кінцевих автоматів [10].

1.1 Методика декомпозиції єдиного інформаційного простору на проекції

ЄІП у контексті даної НДР розуміється як інтегроване віртуальне інформаційне середовище, задачею якого є надання користувачу природних для

нього візуальних і інтелектуальних інтерфейсів з інтегрованою інформаційною системою підприємства (у розглянутому прикладі - університету) (див. Рис. 1.1).

З методичної точки зору декомпозиція такого віртуального інформаційного простору на більш прості складові, котрі в даній роботі визначені як проекції, повинне бути підлеглої задоволенню двох аспектів: інтерфєйсного і функціонального. Також важливо, перш ніж приступити до власне декомпозиції ЄІП, визначитися з поняттями базових компонентів такої розбивки.

У дослідженні поняття ЄІП підприємства/установи визначається на базі загальноприйнятого поняття Віртуального Інформаційного Простору (ВІП). ЄІП визначається як віртуальний простір в зв’язку з тим фактом, що воно містить у собі різноманітні розподілені програмні компоненти: існуючі інформаційні системи, бази даних і знань, клієнти, сервери, співтовариства інтелектуальних програмних агентів і надає їхні ресурси одноманітно за допомогою Уніфікованого Візуального Інтранет Інтерфейсу (УВІІ) [18,19] на базі механізмів інтероперабельности різнорідних інформаційних систем. ЄІП, у такий спосіб є віртуальним гомогенним інформаційним середовищем, орієнтованої на кінцевого непідготовленого користувача.

ЄІП визначається як інформаційний простір за методикою його побудови як багаторівневої інформаційної системи й оскільки його основною функцією є надання користувачу інформації у відповідь на його запити. Типи інформації і відповідні типи запитів є критеріями декомпозиції ЄІП на його проекції.

ЄІП визначається як простір у зв'язку з наявністю в його складі базисних компонентів - проекцій, обумовлених типами інформації і типами елементів УВІІ: топологічної проекції, організаційній проекції і функціональній проекції - Рис. 1.1. Навігація, функціональне наповнення і взаємозв'язки між проекціями ЄІП визначаються також за допомогою УВІІ.

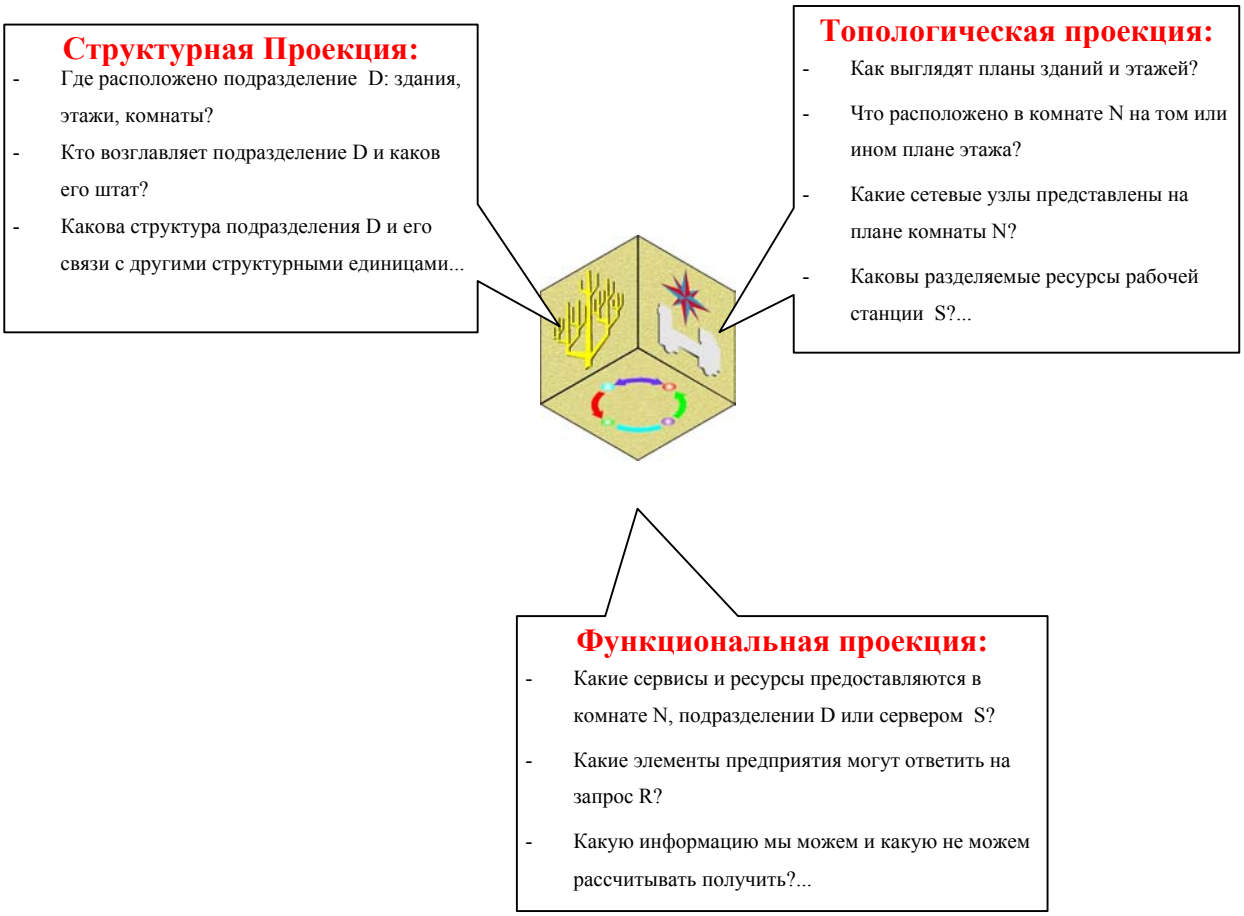


Рисунок 1.1 – Три проекції ЄІП.

1.1.1 Навігація і топологічна проекція ЄІП

Навігаційний або інтерфейснії аспект, як було відзначено, є ключовим у методиці декомпозиції ЄІП на проекції. Навігація по топологічній проекції ЄІП відбувається в такий спосіб.

Верхній рівень топологічної проекції - це рівень кампуса університету (Рис. 1.2.). Цей рівень містить два інформаційних фрейми: контекстно - чуттєвий план і текстовий опис об'єктів, представлених на плані. Слід зазначити, що інформація текстового опису коротко представляє уміст відповідного рівня організаційної проекції ЄІП. Навігація на цьому рівні топологічної проекції досить проста завдяки використанню природно виглядають форм і пропорцій представлених об'єктів. Шляхом вибору будинку користувач попадає на більш низький рівень топологічної проекції - рівень будинку і попадає на план його першого поверху - Рис. 1.3.

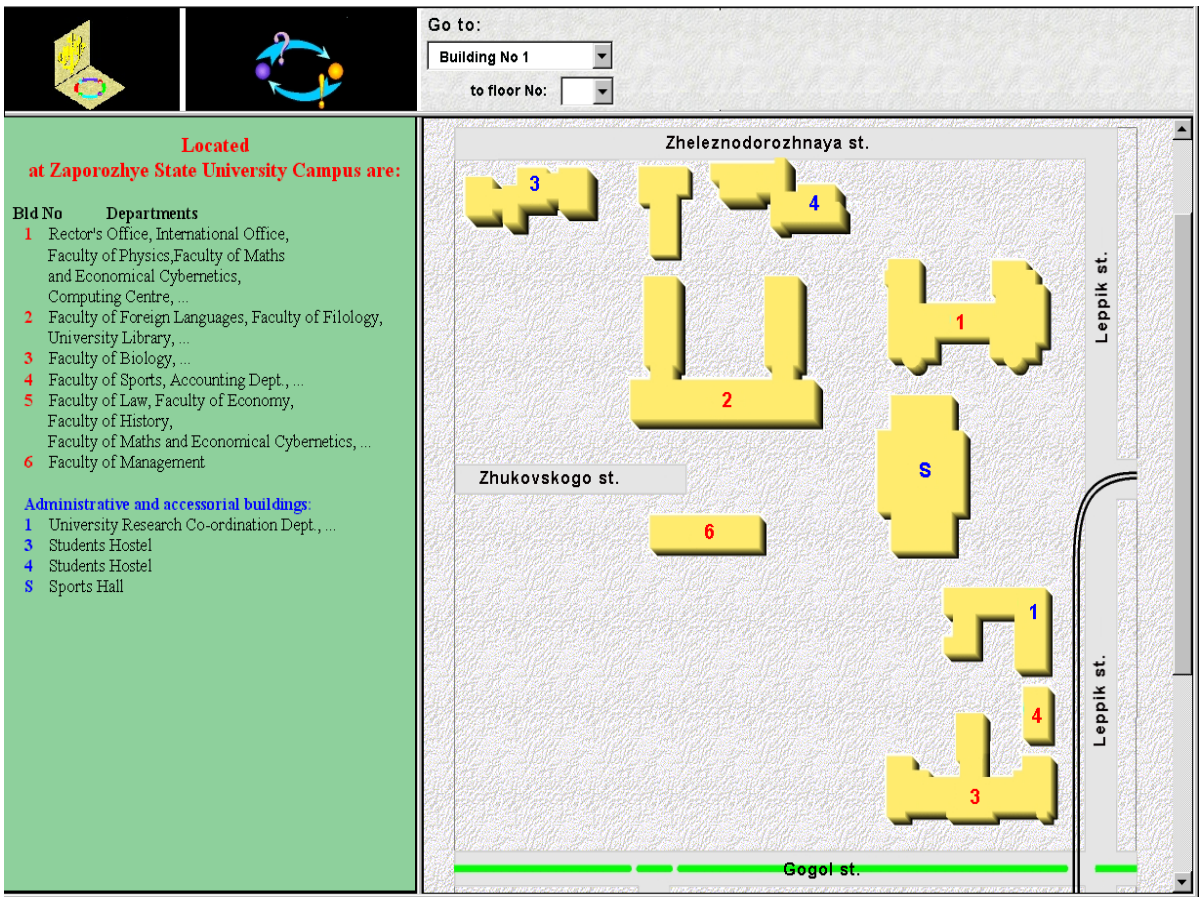


Рисунок 1.2 – Верхній рівень топологічної проекції.

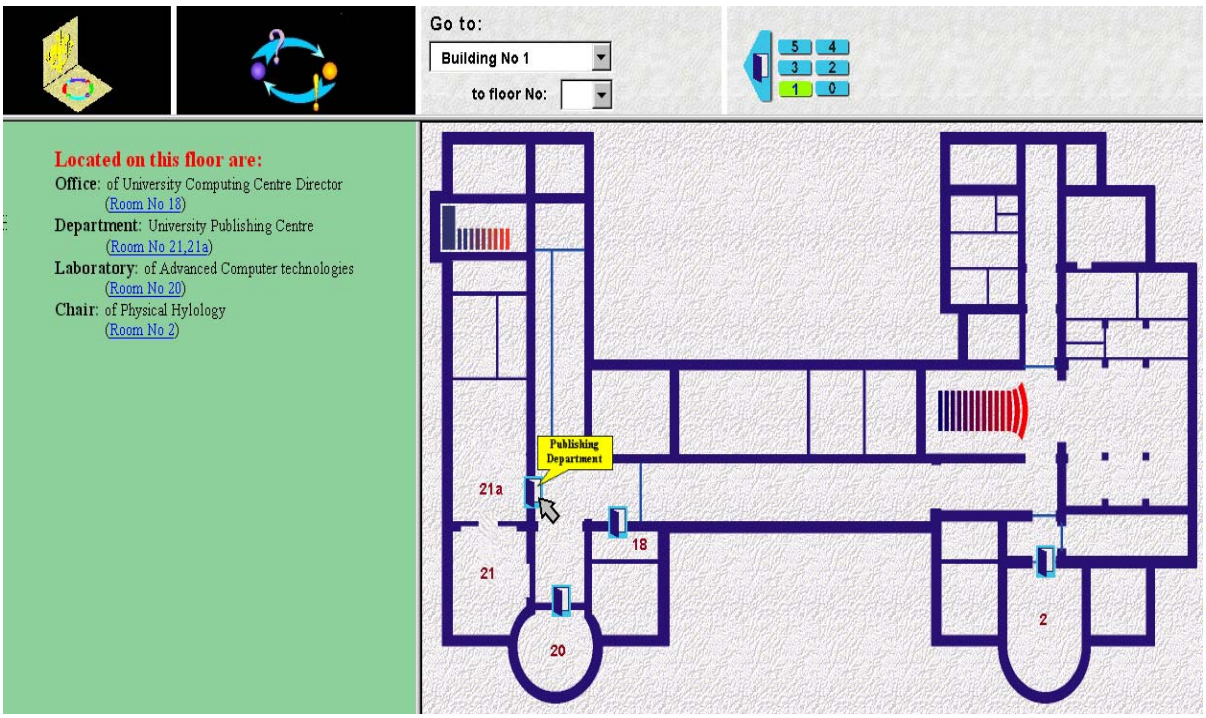


Рисунок 1.3 – Рівень поверху будинку

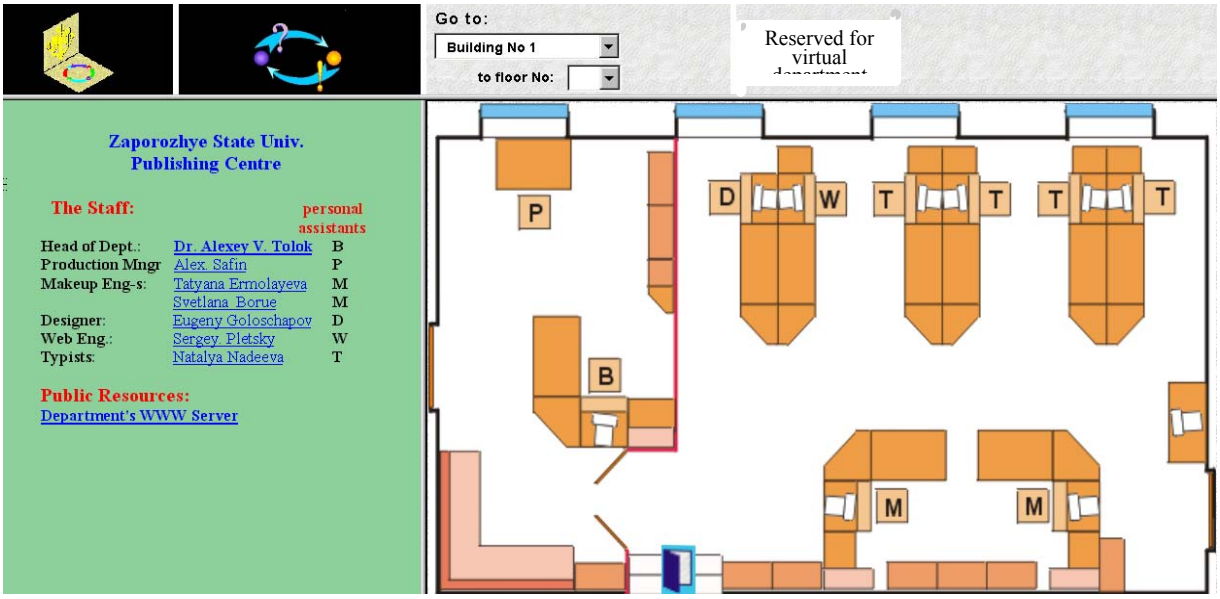


Рисунок 1.4 – Рівень кімнати, офіса

Переміщення між планами поверхів здійснюється за допомогою елементів навігаційного інтерфейсу "сходи" і "ліфт". Для переходу в інший будинок призначений ще один елемент інтерфейсу - "шляхопровід" (кнопка GoTo). У дослідженні передбачається, що в більшості випадків користувач знає, де розташовується необхідний йому ресурс, людина або його агент-асистент. Тому, його задачею є виконання переходу у визначену кімнату визначеного поверху необхідного будинку. Кімнати, відкриті для входу конкретного користувача позначені на плані поверху піктограмою "відкриті двері". Текстовий фрейм цього рівня містить опис доступних кімнат і посилання на об'єкти організаційної проекції відповідного рівня - підрозділу.

При вході в кімнату користувач спускається ще на один рівень топологічної проекції - рівень кімнати, офісу, підрозділу - Рис. 1.4.

На даному рівні представляються об'єкти наступних категорій:

- співробітники, моделируемые і представля їхнім персональним агентами-асистентами;
- мережні ресурси, що представляються піктограмами "комп'ютер".

План кімнати представляється у відповідність з топологією, формами і пропорціями, що відповідають реальним для даного офісу.

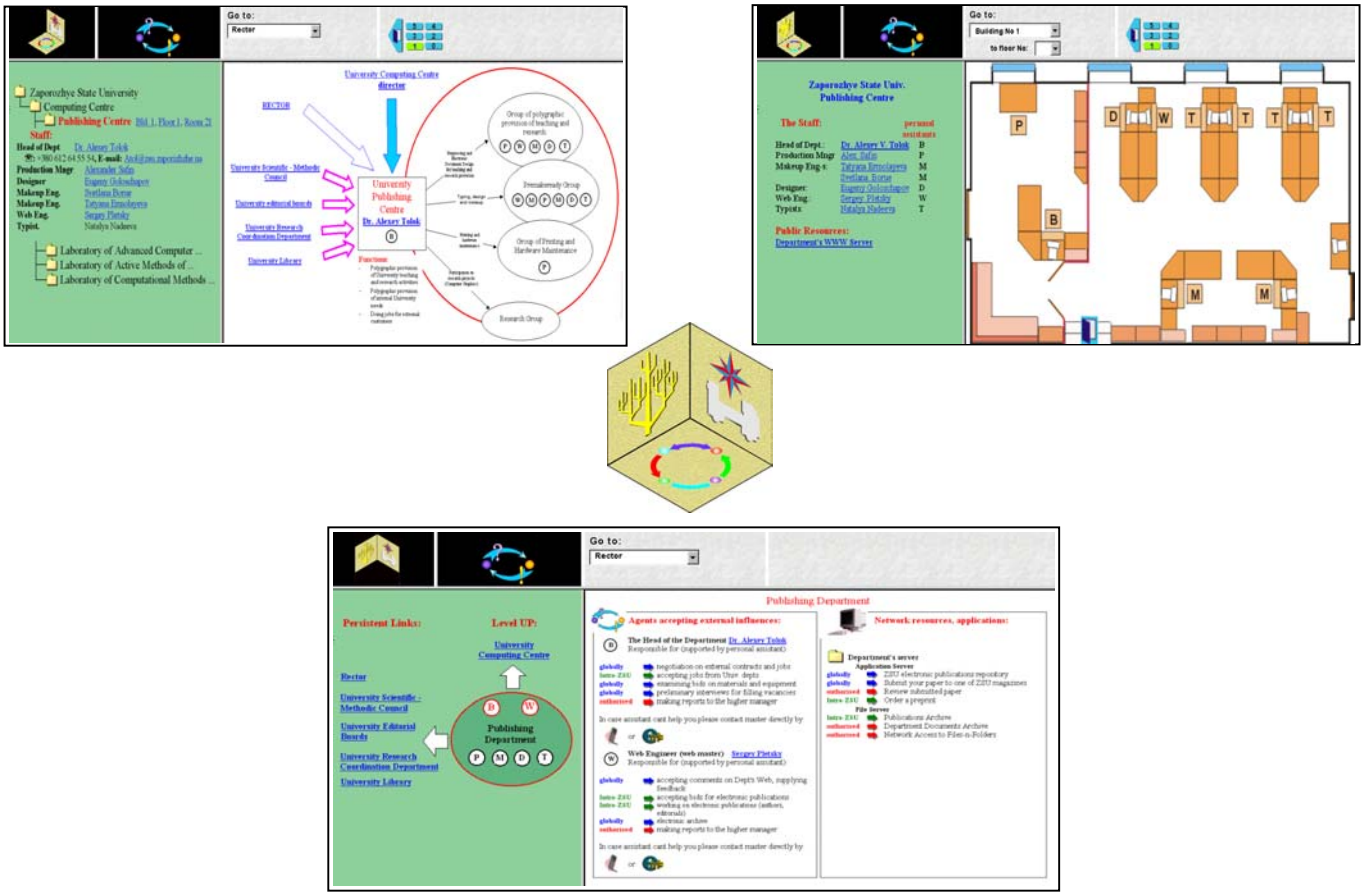


Рисунок 1.5. – Гіперкуб для підрозділу "Видавничий центр"

Текстовий фрейм і на цьому рівні використовується для організації перехресних зв'язків з іншими проекціями ЄІП:

- посилання на персональні домены штатних співробітників підрозділу є виходами в організаційну проекцію;
- зв'язку з функціональною проекцією представляються за допомогою піктограм персональних асистентів і поділюваних ресурсів.

На прикладі розглянутої топологічної проекції ЄІП видно, що всі три проекції ЄІП тісно взаємозалежні й у сукупності утворять деякий просторовий гіперкуб, що і є розроблювальною в даній роботі інтегрованим інформаційним середовищем підприємства. Переходи гіперкуба для підрозділу "Видавничий центр" представлені на Рис. 1.5.

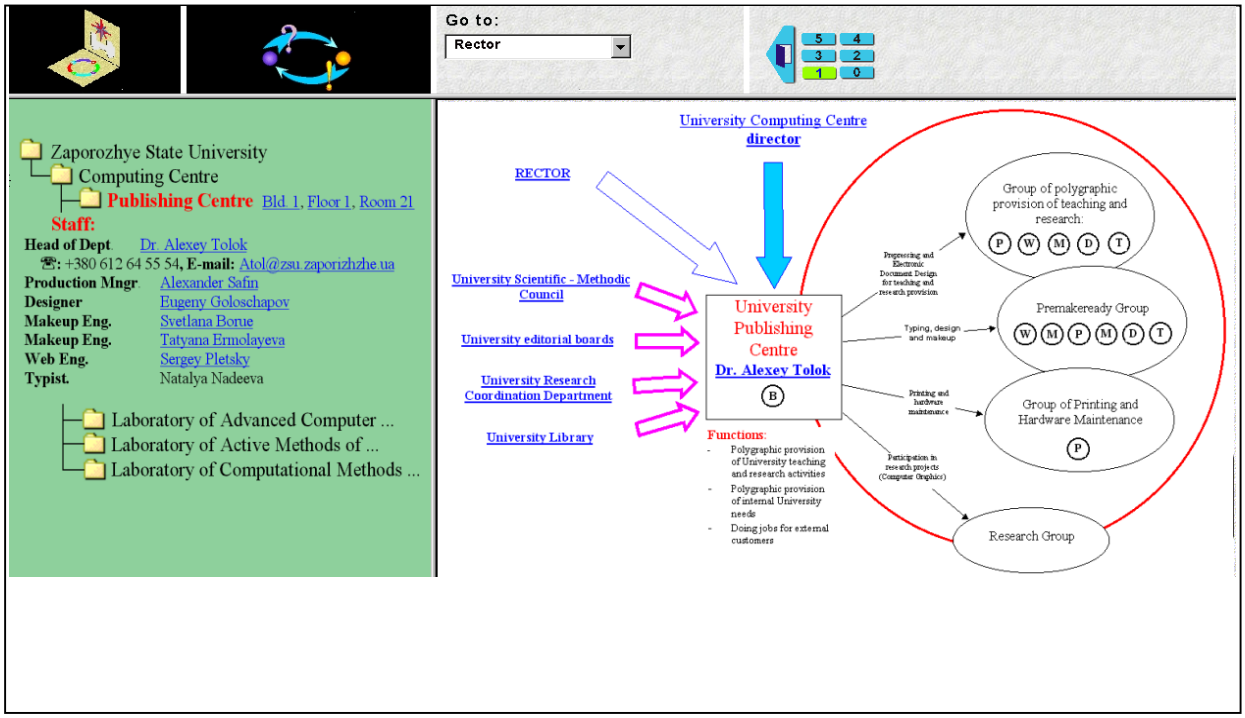


Рисунок 1.6 – Організаційна проекція для Видавничого центра.

1.1.2 Организаціна проекция ЄІП

Графічний фрейм організаційної проекції містить структуру відповідного підрозділу. Приклад представлення структури Видавничого центра приведений на Рис. 1.6. Структура відбиває склад підрозділу, функції і функціональні зв'язки між його компонентами. Модель проекції розрізняє зв'язку двох типів: вертикальні зв'язки підпорядкування (сині стрілки на Рис. 1.6.) і горизонтальні зв'язки (фіолетові стрілки на Рис. 1.6.) між підрозділами і.співробітниками одного організаційного рівня. Для реакції на зовнішні впливи призначені об'єкти, у функцію яких входить обробка того або іншого впливу і повернення відповідної реакції. Ці об'єкти позначені пиктограмой "комп'ютер".

Ще однією активною групою об'єктів є співтовариство персональних агентів-ассистентов даного підрозділу. Вони представлені відповідними ідентифікаторами: У - начальник підрозділу, W - ВЕБ - інженер і т.д. Дана група об'єктів є зв'язком з функціональною проекцією ЄІП даного рівня (підрозділу).

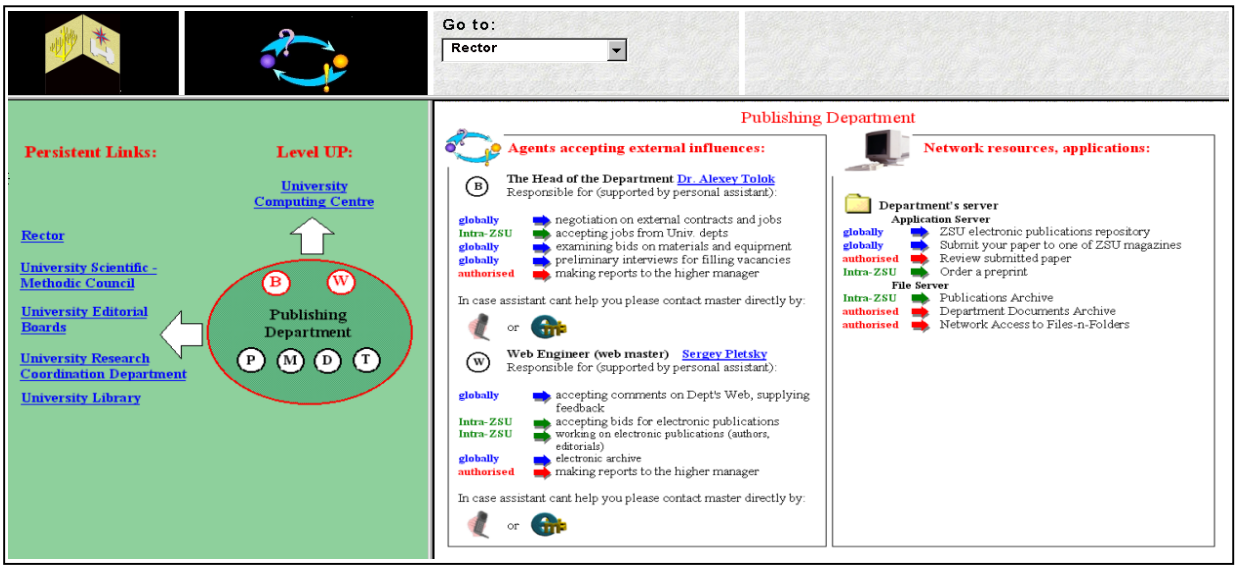


Рисунок 1.7 – Приклад функціональної проекції ЄІП для Видавничого

Текстовий фрейм представляє місце підрозділу в організаційній структурі підприємства - деревоподібній структурі, що відображає зв'язку підпорядкованості між структурними одиницями організації. Крім того, представлена довідкова інформація про персонал і структурні одиниці даного підрозділу.

1.1.3 Функціональна проекція ЄІП

Розробка функціональної проекції ЄІП базується на моделі і методі її представлення у виді набору співтовариств інтелектуальних програмних агентів, що розглядаються в Розділі 2. Приклад функціональної проекції ЄІП для Видавничого центра приведений на Рис. 1.7. З погляду структурної декомпозиції даної проекції виділяється два типи активних функціональних компонентів - агентів: персональних асистентів, що реагують на зовнішні впливи - запити й інтерфейсів з поділюваними мережними інформаційними ресурсами і додатками, що не є агентами. Для кожного інтелектуального агента приводиться динамічний список завдань, що він у стані виконувати замість свого "майстра" - реального співробітника або підрозділу, що належить

даному підрозділу. Текстовий фрейм представляє функціональні зв'язки з іншими підрозділами різних рівнів організаційної проекції.

1.2 Диакоптичний підхід до моделювання відкритих організацій

Одним з істотних висновків, до якого приводить аналіз розглянутого приклада, є твердження про те, що задача моделювання поводження співтовариства функціональних елементів, що, у свою чергу, є одним з елементів у такій системі, як функціональна проекція віртуального інформаційного простору [19], фактично зводиться до задачі моделювання взаємодії елементів сістеми або співтовариства усередині системи вроздріб. Природними частинами, на які декомпозуется NP - повне співтовариство функціональних агентів, є графи взаємодії (плани) елементів системи (співтовариства) у процесі виконання політики у відповідь на прикладений вплив. Такий диакоптичний підхід до дослідження складних систем шляхом їхньої декомпозиції (або розчленовування) був запропонований Кроном [7]. Диакоптичний підхід до моделювання складних електротехнічних систем запропонований школою Демирчана К.С. і зветься «макромоделювання» [20]. Метод макромоделювання в застосуванні до моделювання складних електричних ланцюгів з вентильними елементами розвивався в роботах Борю С.Ю. [21].

Стосовно до досліджуваній нами області застосування загальний метод розчленовування системи формулюється в такий спосіб:

1. Кожний з агентів, членів співтовариства, є замкнутою системою. Ця замкнута система визначається роллю агента - тобто набором політик, що агент виконує; безліччю системних параметрів, тобто тих параметрів, що сприймаються агентом у складі зовнішніх впливів; безліччю станів агента.

2. Взаємодія агентів усередині співтовариства визначається за допомогою понять вплив і реакція на базі рівнянь його стану і поводження.

3. Моделювання реакції співтовариства на зовнішній вплив здійснюється за допомогою підстановки моделі агента виділеного функціонального елемента замість більш складної моделі, що включає в себе всі агенти співтовариства.

4. Моделювання виконання політики базується на виділенні підграфа взаємодії необхідних для виконання даної політики агентів. Топологія підмножини таких агентів змінна, оскільки залежить від як політики, так і від стану взаємодіючих агентів.

5. Модель співтовариства агентів (виходячи з 1. - 4.) еквівалентна моделі агента - члена співтовариства. Таким чином, модель співтовариства є моделлю агента в іншому співтоваристві.

6. Співтовариство агентів є стабільною системою. Це означає, що не існує зовнішнього впливу, сприйманого співтовариством і такого що призводить це співтовариство до руйнування.

Перераховані принципи розчленовування системи не суперечать відомій парадигмі об’єктне - орієнтованого програмування. Це підтверджує, що застосовуваний підхід має право на життя при моделюванні таких складних програмних систем, що еволюціонують, якими є співтовариства інтелектуальних інформаційних агентів у функціональній проекції моделі віртуального інформаційного простору підприємства.

1.3 Підхід до побудови моделі організації на базі співтовариства інтелектуальних агентів

Приведемо розроблене формальне представлення співтовариства інтелектуальних агентів, що моделює функціональний підрозділ підприємства. Розробка моделі співтовариства почата з формалізації його функціонального елемента (агента). Потім, визначені правила взаємодії цих елементів, описані безлічі станів і еволюція елементів моделі й у завершення модель агента, поширюється на співтовариство в цілому.

1.3.1 Підхід до формалізації взаємодії агентів

Взаємодія агентів у розглянутій моделі містить у собі передачу агентом A_i впливу $a^*(f^*, X^*, Y^*)$ агенту A_j і, у разі потреби, чекання агентом A_i реакції (результатів впливу) $\tilde{Y}$ від агента A_j. Випадок впливу без чекання результату являє собою модель передачі директиви. Для агента A_i, що видав директиву, важливо тільки те, що директива отримана виконавцем (агентом A_j) і цього досить, щоб агент A_i, безумовно продовжив виконання своєї політики. Випадок взаємодії шляхом видачі директиви представлений на Рис. 1.8а.

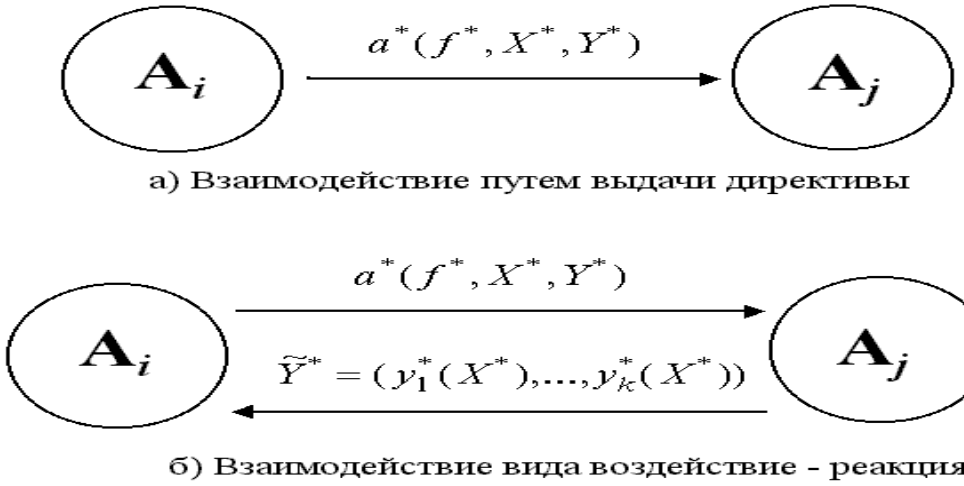


Рисунок 1.8 – Види взаємодій агентів у співтоваристві

Взаємодія виду вплив - реакція, на відміну від випадку передачі директиви, припускає синхронізацію виконання агентами A_i і A_j своїх політик (див. Малий. 1.8б). Такий вид взаємодії агентів, як легко бачити, покриває розглянуті раніше випадки.

Таким чином, модель взаємодії агентів усередині співтовариства може бути описана рівняннями, що характеризують специфіку різних видів взаємодії.

Для завершення формального опису моделі взаємодії інтелектуальних агентів у співтоваристві віртуального інформаційного простору нам необхідно

сформулювати спосіб аналізу результатів, що повертаються. Необхідність у такому аналізі виникає, наприклад, у тому випадку, коли агент A_i, що ініціює вплив a^*, не знає точно, який з агентів A_j у стані виконати політику f^* найбільше повно і найбільш конструктивним образом.

Припустимо, що агент A_i ініціював вплив $a^*(f^*, X^*, Y^*)$ одночасно на трохи агентів $A_j^1,...A_j^q$. Нехай також, кожний з агентів $A_j^1,...A_j^q$ повернув агенту A_i наступні результати:

		$Y^* =$	y_1^*	y_2^*	y_3^*	.....	y_l^*	.....	y_m^*
$A_j^1 \rightarrow$		$\tilde{Y}_1 =$	--	$\tilde{y}_2^1(X^*)$	$\tilde{y}_3^1(X^*)$	.....	--	.....	$\tilde{y}_m^1(X^*)$
$A_j^2 \rightarrow$		$\tilde{Y}_2 =$	$\tilde{y}_1^2(X^*)$	--	$\tilde{y}_3^2(X^*)$	.....	$\tilde{y}_l^2(X^*)$		--
$A_j^3 \rightarrow$		$\tilde{Y}_3 =$	$\tilde{y}_1^3(X^*)$	$\tilde{y}_2^3(X^*)$	--	.....			
....									
$A_j^i \rightarrow$	**Φ =**	$\tilde{Y}_i =$	--	--	$\tilde{y}_2^i(X^*)$		--		$\tilde{y}_m^i(X^*)$
...									
$A_j^q \rightarrow$		$\tilde{Y}_q =$	$\tilde{y}_1^q(X^*)$				$\tilde{y}_l^q(X^*)$		$\tilde{y}_m^q(X^*)$

Нехай, нарешті, для кожного стовпця отриманої в такий спосіб матриці **Φ** відоме відображення λ_l, що переводить елементи цього стовпця в числове значення з проміжку $[0,1]$.

Тоді задача аналізу результатів і вибору агента A_j^i, найбільш підходящого для виконання політики f^*, може, наприклад, зводиться до задачі пошуку рядка матриці **Φ**, для якої сума $\lambda_l(\tilde{y}_l^i(X^*))$ максимальна:

$$A_j <-- A_j^i : \sum_{l=1,m} \lambda_l(\tilde{y}_l^i(X^*)) = \max \qquad (1.1)$$

Для побудови відображень λ_l для результатів нечислової природи раціонально скористатися методом кластеризації, запропонованим у [9].

1.3.2 Облік станів і еволюції агентів - членів співтовариства

Еволюція інтелектуальних агентів у співтоваристві, виходячи з того, що розроблена модель дискретна, - є процес переходів кожного з агентів A з одного стану s_i в інший стан s_j. Важливо відзначити, що від стану агента A залежить те, як він виконує політику f, а також обмеження на вхідні параметри впливу X^*. Таким чином, еволюція агента - є еволюція його ролі. Ці розуміння приводять до наступного визначення безлічі станів інтелектуального агента A:

Безліч станів агента A: $S_A = \{s_1,...,s_n\}$ - є безліч елементів $s_i, i = 1,...,n$, кожний з яких являє собою кортеж наступного виду:

$$s_i = \{r(X_A), q(F_a), t(F)\} \qquad (1.2)$$

де

$r(X_A)$ - функція обмежень на системні параметри агента A, що накладаються в стані s_i;

$q(F_a)$ - функція обмежень на авторизацію політик, виконуваних агентом A;

$t(F)$ - функція, що визначає переходи зі стану s_i, здійснювані політиками $f_1,...,f_i,...,f_m$ агента A.

Моделі співтовариств із декількома крапками сприйняття зовнішніх впливів виходять простим з'єднанням моделей для кожного виділеного елемента.

1.4 Підсумки

У даному розділі приведена методика декомпозиції гіперкуба ЄІП на його базові складові - проекції. З методичної точки зору декомпозиція ЄІП на проекції підкоряється задоволенню двох аспектів: інтерфейсного і функціонального. Декомпозиція по інтерфейсним критеріям регулюється можливістю застосування тих або інших елементів УВІІ. Декомпозиція по

функціональній приналежності визначається операційним призначенням структурованої сукупності компонент - проекції. Топологічна проекція містить компоненти відбивають у ЄІП природне розташування, форми, пропорції і взаємозв'язок по рівнях вкладеності моделей реальних елементів підприємства. Організаційна проекція моделює структуру й організаційну взаємодію підрозділів підприємства, представляючи тим самим відносини керування і взаємодії. Функціональна проекція моделює виконання процесів, що відбуваються усередині підприємства, підрозділу.

Представлена формальна математична модель динамічного співтовариства інтелектуальних функціональних, що еволюціонує, компонентів для реалізації елементів різних проекцій ЄІП. Дана модель розроблена на основі диакоптичного підходу до моделювання складних динамічних систем, з використанням принципу макромоделювання. Функціональні елементи моделі представлені інтелектуальними агентами. Для моделювання агента розроблена узагальнена модель на базі теорії кінцевих автоматів і загальноприйнятих принципів взаємодії інтелектуальних агентів у МАС. Механізми взаємодії агентів, включені в розроблену модель, відповідають сучасним пропозиціям по стандартизації протоколів зазначених взаємодій.

Розроблена модель досить повна і має достатні засоби для застосування в практичній реалізації компонентів проекцій ЄІП.

Моделі проекцій ЄІП будуються на основі операційних компонентів різних рівнів архітектури ЄІП. Одним з важливих додаткових результатів є показана можливість реалізації проекцій і елементів УВІІ на базі багаторівневої розподіленої інтегрованої інформаційної системи підприємства. Викладені способи реалізації дозволяють створювати гнучкі адаптивні і мобільні компоненти проекцій ЄІП на базі зазначених архітектурних принципів і методики застосування активного словника даних.

Розроблена формальна модель досить повна і має достатні засоби для застосування в практичній реалізації компонентів проекцій ЄІП.

2 Архітектура розподіленої ІС організації для представлення проекцій ЄІП

Моделі проекцій, об'єктів і відповідних інтерфейсів представляються на базі архітектури [5], що лежить в основі ЄІП розподіленої інформаційної системи (ІС), Така ІС повинна мати наступні властивості:

- ІС повинна мати засоби одержання ресурсів і вирішення проблем, що користувач, можливо, має намір одержувати або вирішувати;
- ІС повинна надавати інтуїтивно зрозумілі користувачу інтерфейси з навігацією, природної для його області додатка,
- ІС повинна прозорим і адаптивним образом абсорбувати всі програми, засоби й інші програмні компоненти, що уже «думають за користувача».

Першим рівнем архітектури ІС ЄІП є користувальницький або зовнішній рівень. Інтерфейси цього рівня: запит, переданий користувачем ЄІП і результат, що повертається ЄІП браузеру користувача у формі HTML коду. Наступний рівень – рівень Інформаційної Системи Підприємства. Задачею цього рівня є перетворення користувальницького завдання і генерація віртуального запиту до Інформаційної Системи Підприємства. Після обробки цього запиту й одержання результату настає черга ще одного компонента даного рівня. Процесор Результатів запиту генерує відповідному результату HTML код і передає його на більш високий користувальницький рівень. Схема архітектури верхнього (Користувальницького) рівня і рівня ІС підприємства представлена на Рис. 2.1.

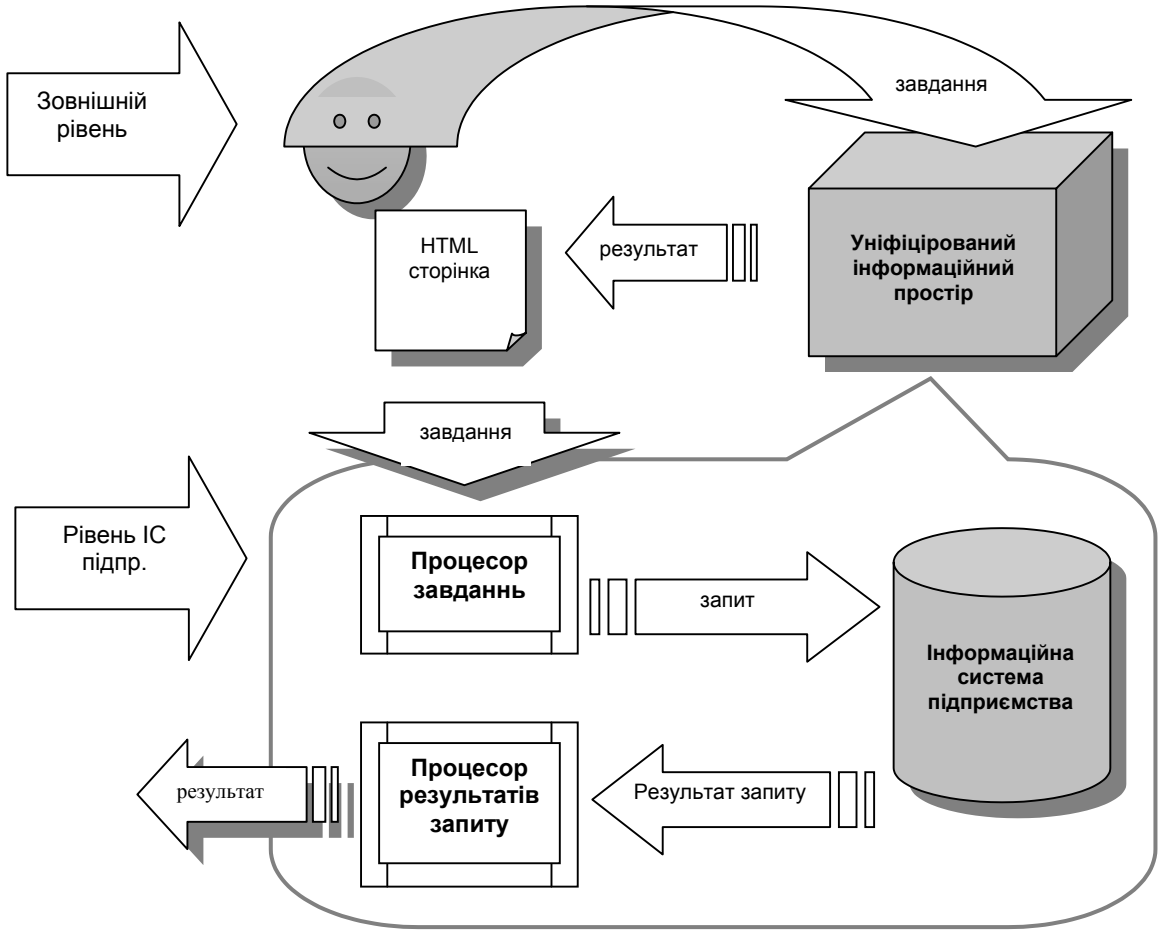


Рисунок 2.1 – Зовнішній рівень та рівень ІС підприємства в архітектурі ЄІП

Після того, як здійснене перетворення користувальницького завдання в запит, виникає можливість описувати ЄІП і УВІІ в термінах моделі даних, об'єктів/сутностей даних і їхніх зв'язків, а також обробляти завдання як запити за допомогою СКБД. Такий перехід від завдання і HTML до запиту і SQL дає досить багато для формалізації інтерфейсів, процесів доступу до даних і супроводу даних, оскільки розроблювач одержує в розпорядження такі могутні засоби автоматизації, як модель даних і СКБД. Для реалізації цього рівня архітектури необхідні: модель даних для опису інтерфейсних елементів, СКБД, процесор користувальницьких завдань і процесор результатів запиту. Фрагмент схеми даних БД ЄІП, що описує інтерфейсних елементи топологічної проекції, приведений на Рис. 2.2.

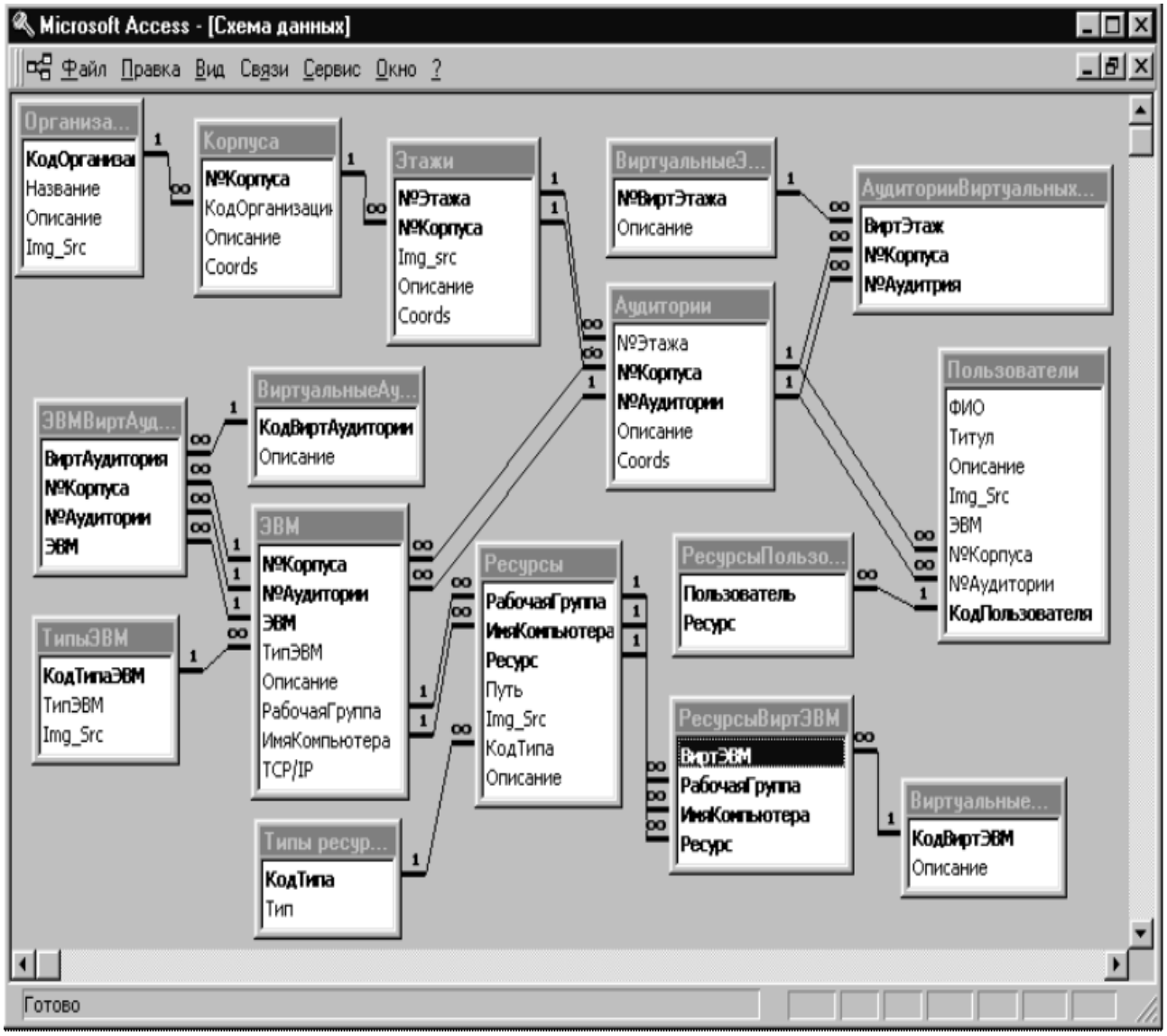


Рисунок 2.2 – Структура опису інтерфейсних елементів
в базе даних ЄІП

2.1 Процесор користувальницьких завдань

Запит користувача до ЄІП являє собою, насамперед, звертання до визначеного елемента інтерфейсу. Для доступу до визначеного об'єкта необхідна наявність його ідентифікатора класу (ІК) і ідентифікатора екземпляра класу (ІЕК). Тому вхідними параметрами Процесора Користувальницьких Завдань можна вважати ІК і ІЕК. Результатом роботи Процесора

Користувальницьких Завдань є SQL-запит. Задачею отриманого запиту до БД ЄІП є одержання наступної інформації:

- даних, що описують конкретний об'єкт класу, обумовлений ІК і ІЕК на вході Процесора Користувальницьких Завдань (ці дані існують в одному екземплярі);

- даних, що дозволяють установити зв'язок між створюваним документом і всіма документами, що описують об'єкти ЄІП, зв'язані з даним.

Кожний клас зв'язаний з декількома іншими класами (ця інформація зберігається в даних класів). Крім того, може існувати кілька конкретних об'єктів визначеного класу, зв'язаних з даним. Отже, кількість записів, повернутих у відповідь на запит до БД ЄІП і що описують зв'язані об'єкти, буде різним для кожного зв'язаного класу. Тому для одержання інформації про кожний зв'язаний клас доцільно генерувати окремий запит.

Для встановлення зв'язку з конкретним об'єктом необхідна наявність ІК і ІЕК цього об'єкта. Крім цих даних нам необхідно одержати ту інформацію, на основі якої буде створений елемент навігації, що представляє собою елемент інтерфейсу, зв'язаний з необхідним об'єктом.

Таким чином, формальна структура згаданих двох типів запитів до БД ЄІП буде такий:

Запит на одержання даних, що описують об'єкт, переданий користувачу:

```
SELECT [ІК_НА_ВХОДІ.ДАНІ_ЕКЗЕМПЛЯРА_КЛАСУ]
FROM [ІК_НА_ВХОДІ]
WHERE [ІК_НА_ВХОДІ.ІЕК]=ІЕК_НА_ВХОДІ;
```

ТУТ [ІК_НА_ВХОДІ] позначає ім'я таблиці, у якій зберігається інформація про об'єкт, [ДАНІ_ЕКЗЕМПЛЯРА_КЛАСУ] являють собою набір полів таблиці [ІК_НА_ВХОДІ], що описує об'єкти даного класу, І_НА_ВХОДІ.ІЕК - полючи, що є ключовими.

Запит на одержання даних, що описують зв'язані об'єкти:

```
SELECT [ІК_ЗВ'ЯЗАНИЙ_ОБ'ЄКТ.ІЕК],
        [ІК_ЗВ'ЯЗАНИЙ_ОБ'ЄКТ.ЕЛЕМЕНТ НАВІГАЦІЇ]
FROM
[ІК_НА_ВХОДІ] INNER JOIN [ІК_ЗВ'ЯЗАНИЙ_ОБ'ЄКТ] ON
[ІК_НА_ВХОДІ.ІЕК_НА_ВХОДІ] =
[ІК_ЗВ'ЯЗАНИЙ_ОБ'ЄКТ.ІЕК_НА_ВХОДІ]
WHERE [ІК_НА_ВХОДІ.ІЕК]=ІЕК_НА_ВХОДІ;
```

Тому що деякі об'єкти інтерфейсу можуть бути описані за допомогою різної кількості таблиць і для вибору необхідної інформації будуть застосовуватися різні умови, то структура конкретних запитів буде трохи відрізнятися від описаних вище схем. Тому для кожного класу необхідно зберігати свій набір SQL-запитів. Крім того, необхідно зберігати набір таблиць, з яких витягається інформація з кожного запитуваного елемента інтерфейсу, з описом структури кожної таблиці, тобто конкретним набором необхідних полів (атрибутів).

Виходячи з вищевикладеного, укрупнений алгоритм роботи Процесора Користувальницьких Завдань можна представити в такий спосіб:

а) Визначити клас, до якого належить необхідний елемент інтерфейсу по
ІК_НА_ВХОДІ.

б) Визначити ІК зв'язаних об'єктів.

в) Згенерувати SQL-запити, що відповідають даному класу, використовуючи
інформацію у відповідних таблицях, застосовуючи як умову добору інформації значення ІЕК_НА_ВХОДІ.

2.2 Процесор Результатів Запиту

Вхідною інформацією Процесора Результатів Запиту є набір таблиць даних, що сгенеровані у відповідь на запит до БД ЄІП. Структура цього набору визначається виходячи із шаблона запитуваного об'єкта – дані екземпляра класу плюс дані по кожному зв'язаному класі. Структура кожної конкретної таблиці визначається виходячи із шаблонових таблиць, використовуваних при створенні SQL-запиту перерахуванням усіх полів.

Вихідною інформацією є готова HTML-сторінка, передана користувачу. Шаблон цієї сторінки є статичним документом. Для перетворення шаблона в готовий документ у його вихідному тексті необхідна наявність керуючих елементів, що дозволяють заповнити його необхідною інформацією.

Для створення цих керуючих елементів використовується тагова модель HTML.

```
"Елемент" := <"ім'я елемента" "список атрибутів">
зміст елемента
</"ім'я елемента">
```

Уведемо наступний керуючий елемент (таг):

```
<ЗАПИТ ІМ'Я> </ЗАПИТ>
```

При використанні цього тага його зміст буде представляти із себе наступне:

```
<ЗАПИТ ІМ'Я=ім'я запиту>
<!-і  елементи шаблона -і>...<!-і  елементи шаблона -і>
 %ПОЛЕ_1%
. . . . . .
<!-і  елементи шаблона -і>...<!-і  елементи шаблона -і>
 %ПОЛЕ_N%
```

```
<!-і  елементи шаблона -і>...<!--  елементи шаблона -
->

</ЗАПИТ>
```

Тут `ІМ'Я` визначає таблицю, з якої повинна витягатися інформація, `%ПОЛЕ_N%` - поле даної таблиці. Якщо дана таблиця має кілька записів, Блок `<ЗАПИТ>…</ЗАПИТ>` дублюється в тексті HTML-документа відповідно до кількості записів, повернутих у відповідь на даний запит.

Алгоритм роботи Процесора Результатів Запиту виглядає в такий спосіб:

а) Визначити структуру вхідних таблиць на підставі шаблонів таблиць для побудови SQL-запитів, що відповідають об'єкту з `ІК=ІК_НА_ВХОДІ`.

б) Вибрати першу/наступну таблицю.

в) Якщо таблиця існує, те перейти до г), інакше перейти до е).

г) Вибрати першу/наступний запис.

д) Якщо запис не порожній, то замінити елементи `%ПОЛЕ_N%` блоку `<ЗАПИТ>…</ЗАПИТ>` значеннями відповідних полів і перейти до г), інакше перейти до б).

е) Повернути користувачу HTML-сторінку.

На жаль, при аналізі обробки віртуального запиту виникають додаткові проблеми. ІС рівня підприємства в дійсності являє собою досить складний агрегат безлічі локальних Інформаційних Систем і Функціональних Серверів, розподілених у мережі підприємства, і керуючих локальними додатками і ресурсами. Дані і функціональні характеристики цих додатків і ресурсів найчастіше мають семантичні перетинання. Одним з відомих методів вирішення проблеми семантичних перекриттів є використання Федеративних моделей даних і СКБД.

Методи федералізації даних у даний момент досить добре розроблені і широко обговорюються. В архітектурі ЄІП використовується метод і модель, розроблені в університеті Отто-вон-Герике, Магдебург групою проф. Зааке [22, 23], посилені застосуванням Активного Словника Даних [24]. Для реалізації функціональної проекції ЄІП у роботі використовується модель динамічних співтовариств інтелектуальних інформаційних агентів.

Таким чином, Федеративний рівень архітектури ЄІП може бути схематично представлений, як показано на Рис. 2.3. Цей рівень містить у собі два підрівні з відповідними інтерфейсами - Федеративний підрівень з функціями, досить близькими до функцій зовнішнього рівня федеративної моделі даних, і підрівень, що містить набір локальних серверів інформаційних ресурсів, взаємодіючих з федеративним підрівнем за допомогою інтерфейсу, що базується на запитах. Як показано на Рис. 2.3, основними функціями федеративного підрівня є керування прозорим об'єднанням моделей даних локального рівня, трансляція, розподіл і маршрутизація вхідних запитів, а також обробка результатів виконання локальних запитів, що, у загальному випадку приходять від більш ніж однієї локальної Інформаційної Системи і/або локального сервера ресурсів. Федеративний рівень архітектури ЄІП інтерпретує вхідний віртуальний запит у набір локальних запитів і, відповідним чином, направляє їх до компонентів локального рівня архітектури.

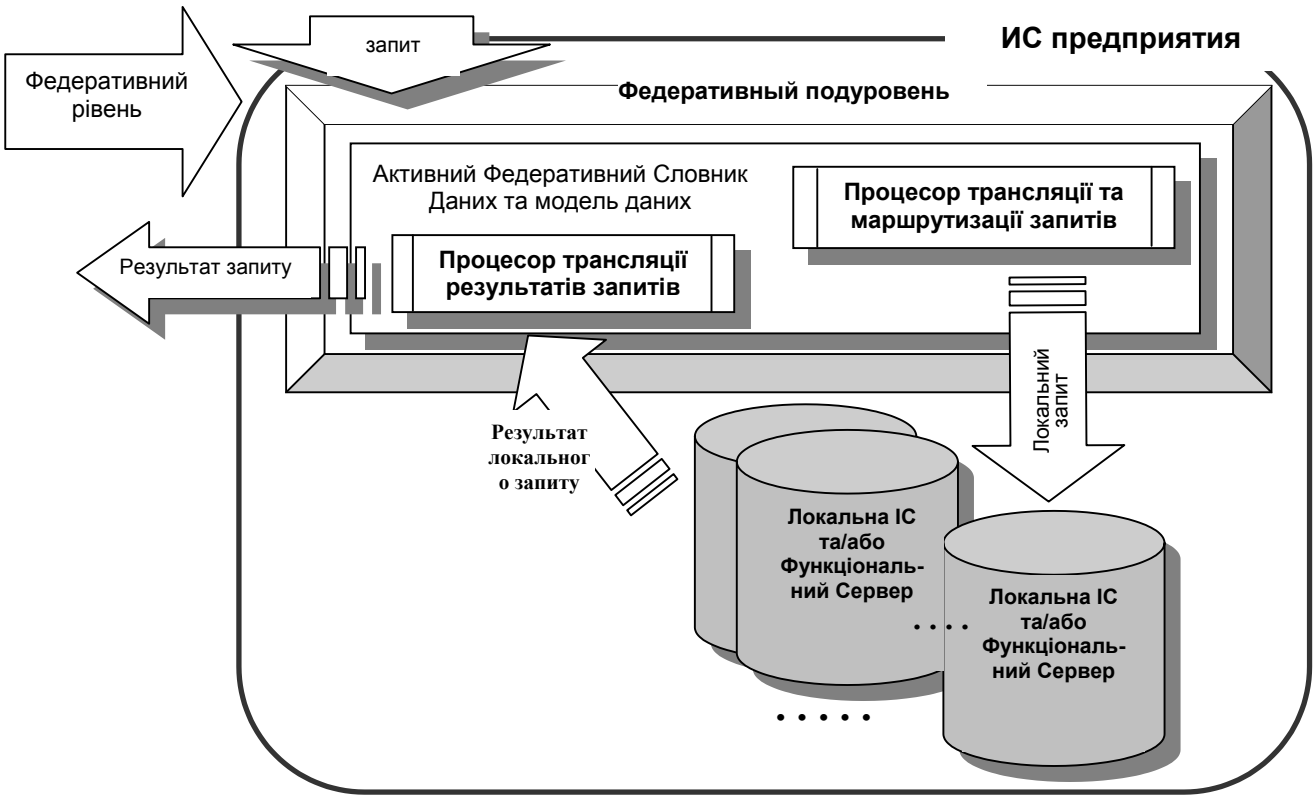


Рисунок 2.3 – Федеративний рівень архітектури ЄІП

Основною перевагою, якою ми володіємо на локальному рівні, є метод створення і керування Локальної ІС. Як керуючу оболонку ІС у роботі використовується Активний Словник Даних, а як керуючий компонент - Посилений Репозиторій Метаданих. Локальний рівень складається з трьох шарів: Шару Програмного Коду, Шару Репозиторія Метаданих і Шару Даних (див. Рис. 2.4). Словник даних з його активними функціями, керованими моделлю даних, служить у якості інтелектуального гнучкого інтерфейсу між більш високими рівнями архітектури і локальним даними і/або ресурсом.

Такі зміни в структурі локальної ІС виразно впливають на технологію створення інформаційної системи. Модифікована процедура розробки ІС базується на визначених властивостях Активного Словника Даних і лежачий у його основі розширеної моделі даних.

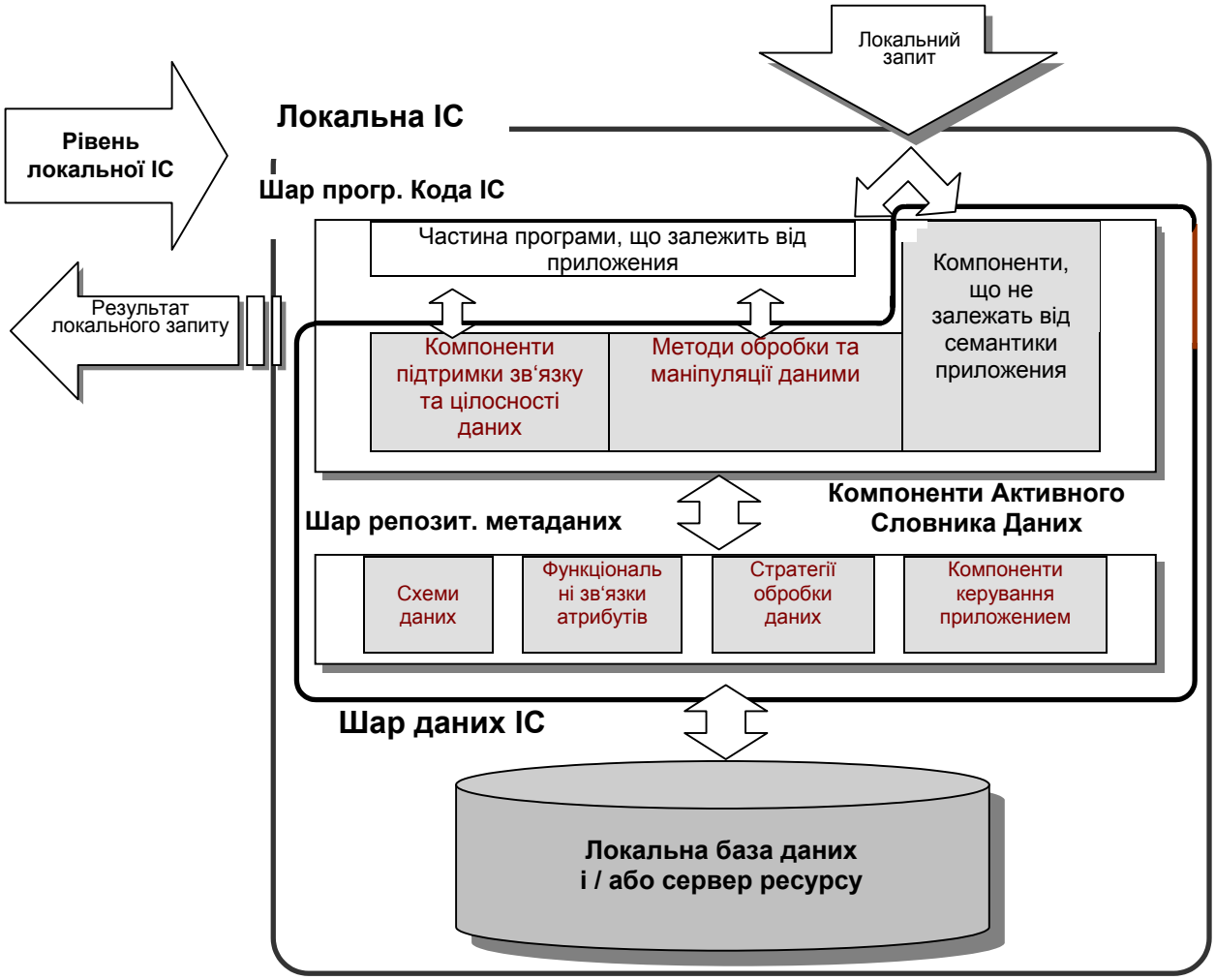


Рисунок 2.4 – Архітектура рівня Локальної ІС ЄІП

2.3 Активний репозиторій метаданих і його базова модель

Основною задачею Активного Словника Даних (АСД) є постачання інформаційної системи властивостями самоадаптації до змін у моделі даних. Якщо ми, для визначеності, обмежимо обговорення локальним рівнем архітектури ЄІП і моделлю Клієнт - Сервер, АСД може побут визначений як інтелектуальною, керованою моделлю даних посередник (медіатор) між додатками клієнтами ІС і серверною частиною бази даних. Принципова схема співробітництва Клиент-асд-сервер приведена на Рис. 2.5.

Базова модель АСД і Розширеного Репозиторія Метаданих являє собою розширену реляційну модель. Основним виконаним нами удосконаленням реляційної моделі є метод керування програмним кодом клієнтської частини ІС і настроювання на зміни моделі даних ІС. Цей метод базується на деяких нових формальних властивостях уведених нами керуючих атрибутів (атрибутів даних, що володіють деякими специфічними характеристиками) [24], що мають зв'язку з об'єктами програмного коду. Таким чином, активне поводження АСД є двонаправленимим - див. Рис. 2.5. Впливу в напрямку даних засновані на застосуванні відомих активних правил (ECA), у той час, як керування програмною частиною здійснюється за допомогою спеціальних порцій програмного коду АСД, впроваджених у програму клієнтської частини, під керуванням Сервера АСД.

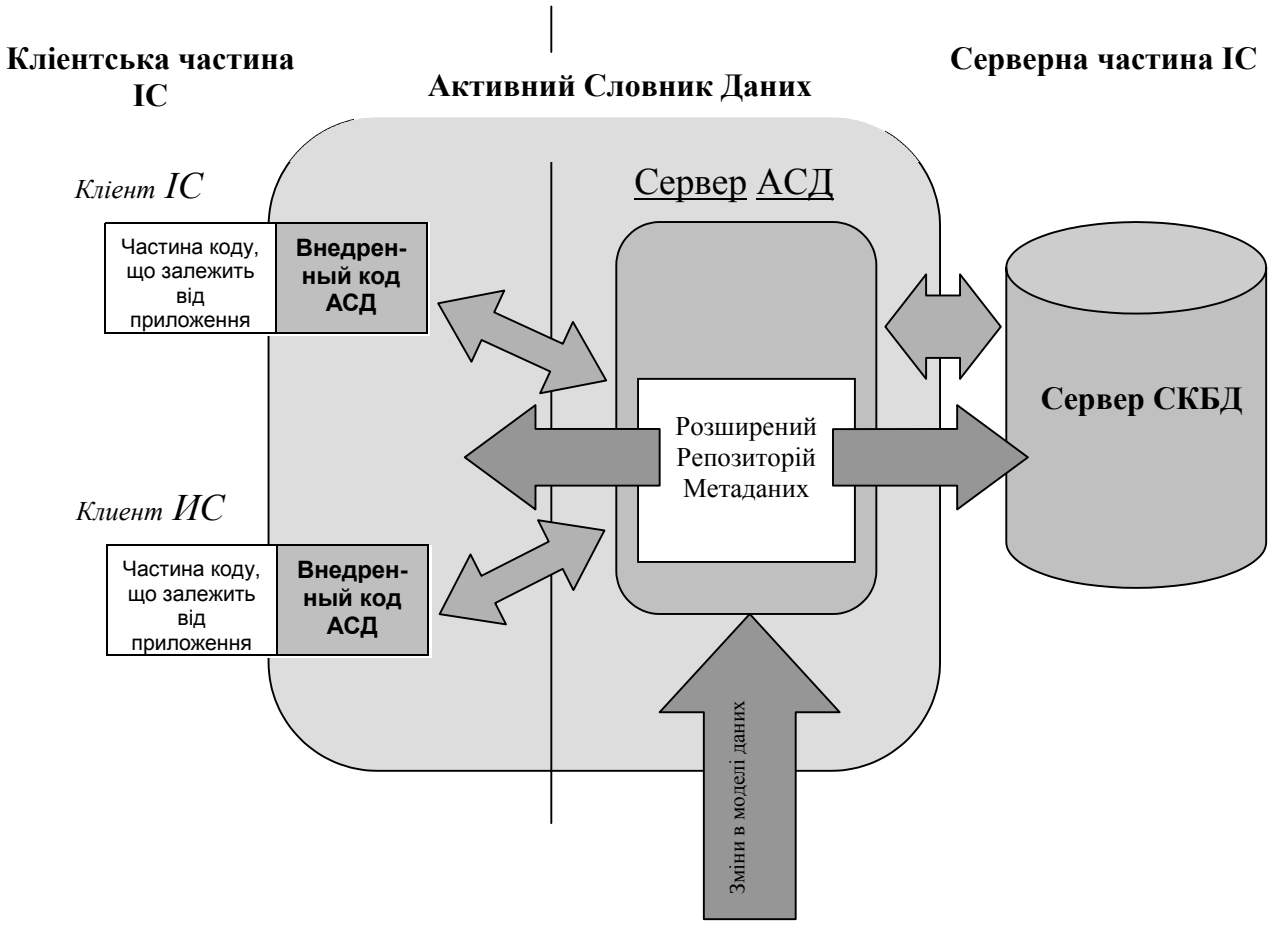


Рисунок 2.5 – Принципова схема взаємодії Клієнт-АСД-Сервер

2.4 Підсумки

У даному розділі викладені способи реалізації проекцій ЄІП на базі операційних компонентів різних рівнів архітектури ЄІП. Одним з важливих додаткових результатів є показана можливість реалізації проекцій і елементів УВІІ на базі багаторівневої розподіленої інтегрованої інформаційної системи підприємства. Викладені способи реалізації дозволяють створювати гнучкі адаптивні і мобільні компоненти проекцій ЄІП на базі зазначених архітектурних принципів і методики застосування активного словника даних.

3 Базові моделі представлення функціональних об'єктів Єдиного Інформаційного Простору

Даний підрозділ презентує формальний підхід, що імплементує функціональні об'єкти у кібер-середовищі, заснованом на агентах, для моделювання процесів інформаційного обміну у Єдиному Інформаційному Просторі (ЄІП) населеном агентами, які створюють суспільства для підтримки виконання різних бізнес-процесів, яки є характерними для вищого навчального закладу (ВНЗ) і віртуальної установи (ВУ).

3.1 Базові принципи моделювання функціональних об'єктів в ЄІП

В рамках даної роботи було запропоновано [10,19] середовище для моделювання процесів функціональних взаємозв'язків між розподіленими компонентами (які є інтелектуальними агентами) динамічної багатофункціональної інформаційної системи підприємства / відділу, особливістю якого є той факт, що асоціації між вузлами / акторами є нежорсткими і змінюються із часом. Підхід заснований на діакоптичному принципі [22] і методології [23] макромоделювання. Використання діакоптики дає такі елегантні рішення, як:

- представлення як компонентів системи, так і системи в цілому за допомогою функціонально еквівалентних, об'єднаних і спрощених моделей програмного забезпечення (інтелектуальних агентів із їхніми моделями поведінки, поданими відповідно до макромодельних програм);
- включення топології взаємозв'язків компонентів у загальну макромодель;
- включення спеціалізацій компонентів у загальну макромодель.

Переваги цього середовища такі: з’являється можливість взяти узагальнені моделі к представленню компонентів, їх комунікацій між собою, підсистем і системи як єдиного цілого; є гарні можливості працювати із підсистемами за допомогою більш простих моделей компонентів незалежно від їх внутрішніх рівнів складності; є можливості використати узагальнені архітектури,

інтерфейси і реалізовувати масштабовані задачі. Замість створення досить складних, жорстких моделей із сильно зв’язаними компонентами, метод дозволяє оперувати з колекціями "автономних" агентів, які динамічно формують суспільства для виконання того чи іншого процесу інформаційного обміну (завдання). Це зменшує складність, додає руху і дуже підходить для моделювання бізнес-процесів як процесів інформаційного обміну, оскільки пропонує підхід розділення процесу на частини. Набір моделей для представлення функціональних вузлів ЄІП включає: модель компоненти/функціональної системи [16], модель процесу [17], узагальнену модель агента [16].

Актори у середовищі є інтелектуальними (раціональними – у Nwana [25]) програмними агентами. Вважається, що завдання, які виконують ці актори – є наборами атомарних робіт. Кожний актор (агент) здатен виконати деяку атомарну роботу із множини дозволених атомарних робіт функціональної системи. Ці здатності формують роль відповідного агента. Нотація ролі, що використана у ЄІП [16], близька до нотації, що прийнята у ICRF [3].

3.2 Модель функціональної системи/компоненти

На рівні функціональної системи були зроблені деякі базові припущення для того, щоб спростити середовище і надати бажаний рівень реактивності агентів. Учасники функціональної системи вважаються жорстко орієнтованими на командну роботу і "чесну гру". Успішне виконання завдання має більший пріоритет ніж локальні цілі окремих агентів. Агенти, що приєднуються до суспільства, обмежені тим, що мають видавати вірні результати навіть якщо це не узгоджується з їх локальними цілями.

Модель функціональної системи, а також і модель функціонального компонента системи будується на ідеях "узагальнення" і "абсорбції" атомарних робіт із множини дозволених робіт $W = \{w_1, w_2, ...\}$ цієї функціональної системи. Вважається, що сенсорний вхід функціональної компоненти *i* приймає завдання

$W_i \subseteq W$. Деяка частина цього завдання (атомарна робота W_i^p) може бути виконана ("абсорбована") конкретною компонентою, а інші частини завдання або направляються іншим компонентам системи W_i^d, в тому випадку, якщо функціональна компонента знає кому переслати ці роботи, або відхиляється W_i^r. Функціональний компонент може створити додатковий набір атомарних робіт W_i^g для завершення виконання роботи W_i^p. Ці роботи W_i^g , разом із W_i^d, далі направляються іншим компонентам:

$$W_i \to F_O^i(W) \to \widetilde{W}_i, \tag{3.1}$$

де $W_i = \{W_i^p, W_i^d, W_i^r\}$, $\widetilde{W}_i = \{W_i^d, W_i^g\}$, $F_O^i(W)$ - макромодельна програма.

У спеціальному випадку компонента *i* може створити новий набір робіт W_i^g, навіть якщо вхідного впливу W_i не було, тобто, компонент може “викликати” нове (під)завдання:

$$F_O^i(W) \to \widetilde{W}_i, \tag{3.2}$$

де $\widetilde{W}_i = \{W_i^g\}$, $F_O^i(W)$ - макромодельна програма.

Тому, завданням для кожного функціонального компонента/моделі системи є відповідне виконання (3.1) і (3.2). На даний момент наша модель обмежується таким правилом, що тривалість виконання кожної атомарної роботи $w_j \in W$ - це заданий проміжок часу Δt. Але, вочевидь, це обмеження не є дуже сильним і може бути знято у майбутньому.

3.3 Модель процесу виконання робіт

Функціональна система призначена для виконання процесів. Процес зазначимо як потік виконання завдання. Процес Π_a починається із створення нового завдання $W_a \subseteq W$. Завдання W_a, а також додаткові завдання $\widetilde{W}_a$ пов’язані із процесом Π_a і помічені унікальним ідентифікатором цього процесу.

Компонент вважається ***пов'язаним з процесом*** Π_a тоді, коли цей компонент може виконати частину завдання W_a, $\widetilde{W}_a$, або може створити роботу W_a^g. Агент, що представляє цей функціональний компонент, ***входить до суспільства агентів, виконуючих завдання***.

Процес Π_a вважається виконаним, якщо всі компоненти припинили абсорбувати атомарні роботи всіх завдань, пов'язаних з процесом Π_a. Множина робіт $W_{\Pi_a}^z$, не абсорбована процесом Π_a, помічається як ***множина робіт, що не можуть бути виконані***. Див. Рис. 3.1.

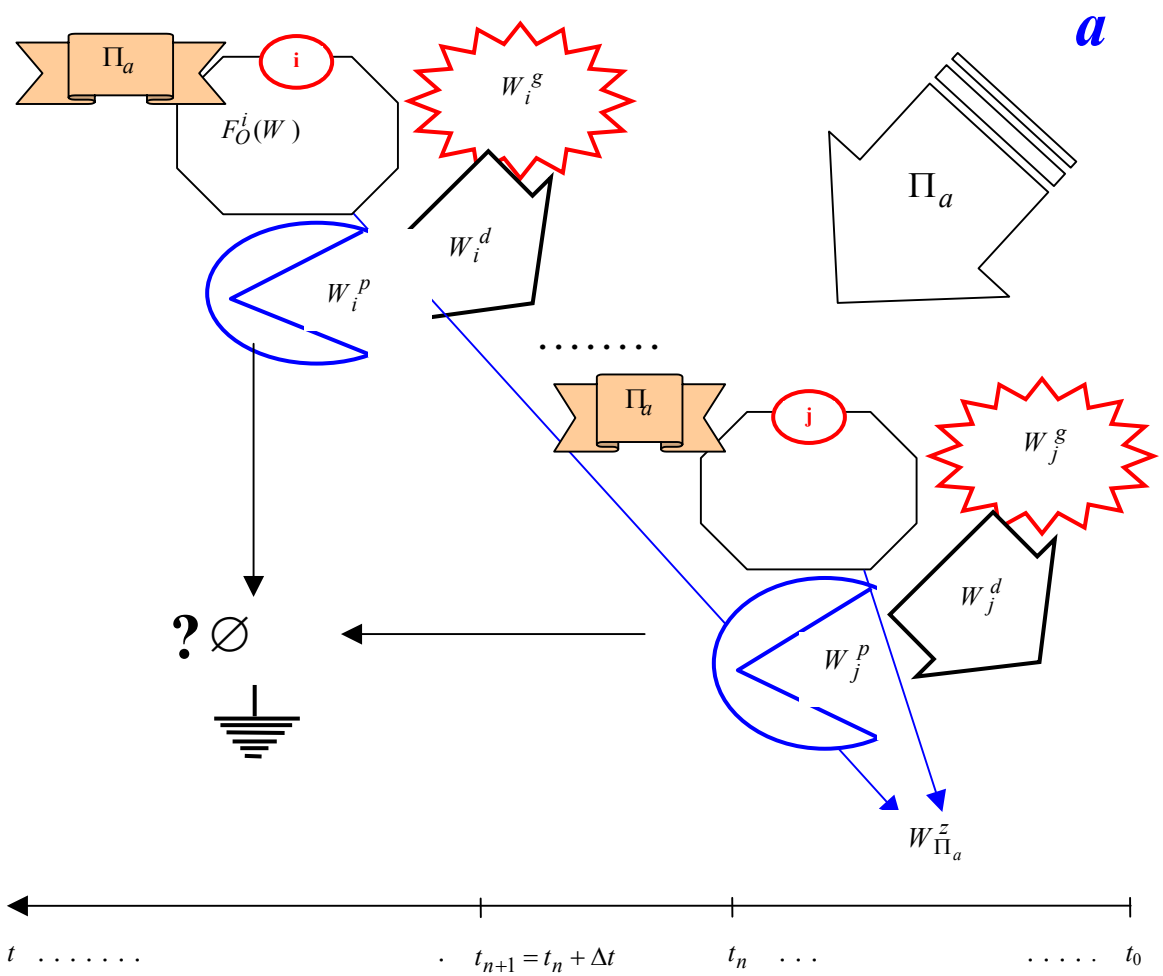


Рисунок 3.1 - Модель процесу

Моделювання процесу Π_a (в усталеному режимі) проводиться із застосуванням схем (3.1) і (3.2) до усіх компонентів системи, до тих пір, як весь процес не виконаний.

Залежність $F_O^i(W)$ моделюється в рамках узагальненої моделі агента [16], або за допомогою будь-якого подібного методу. Вимогою для цієї частини функціональної компоненти є адекватне виконання схем (3.1), (3.2).

На практиці буде розумніше обмежити множину дозволених атомарних робіт, що їх виконує система, а також вважати, що ця множина є скінченною: $W = \{w_1, w_2, ..., w_\sigma\}$. Тоді моделювання роботи функціональної системи у цілому по виконанню завдання організується як дворівневий процес, що виконується поступово, в дискретні проміжки часу t_n, $t_{n+1} = t_n + \Delta t$.

Модель виконання завдання: Нехай $W = \{w_1, w_2, ..., w_\sigma\}$ - множина припустимих атомарних робіт функціональної системи.

На першому (верхньому) етапі виконується збирання станів всіх компонент в об'єднану модель системи станів в період $t_n + \Delta t$.

Ця об'єднана модель має вигляд матриці $\Omega(t_n + \Delta t)$ розмірності $m \times \sigma$, де m - це кількість компонентів системи, а σ - це кількість атомарних робіт в завданні W. Строки матриці Ω (Рис. 3.2) є векторами $\Theta_i = \{k_1, k_2, ..., k_j, ..., k_\sigma\}$, що відображають стани компонентів, де k_j - це стан компонента *i* при виконанні атомарної роботи w_j. В найпростіших випадках, роль параметра k_j може бути такою:

$k_j = 0$ - компонент зараз виконує атомарну роботу w_j;

$k_j = l > 0$ - компонент зараз виконує атомарну роботу w_j і ще l таких самих робіт чекають у черзі;

$k_j = l < 0$ - компонент був здатен, але не виконав *l* атомарних робіт w_j (тобто , був "ледачий").

Матриця станів системи $\Omega(t_n + \Delta t)$ формується агентом-координатором із матриць $\mathbf{K}_i$ (розмірності $m \times \sigma$), які представляють стани компонентів. Матриці $\mathbf{K}_i$ створюються виконавцем макромодельної програми $F_O^i(W)$ моделі компоненти на другому етапі моделювання так, щоб забезпечити вхідні дані для формули:

$$\Omega = \sum_{i=1}^{n} \mathbf{K}_i , \tag{3.3}$$

Функціональний компонент може перенаправити собі одну чи більше робіт w_j із завдання W_i для виконання у наступний момент часу. В цьому випадку вектор D_a затримок роботи над процесом Π_a поновлюється так:

$$D_a[j] = D_a[j] + 1, \tag{3.4}$$

На другому (нижньому) рівні кожна компонента системи (агент) створює $\mathbf{K}_i$. Ці компоненти, як було зазначено вище, моделюються вільно($F_O^i(W)$), але так, щоб вхідна інформація для компоненти *i* була така: вектор Θ_i, матриця $\Omega(t_n)$ та матриця $\mathbf{K}_i$, означені на попередній момент часу t_n. Матриця $\mathbf{K}_i$ будується згідно з правилом, поданим на Рис. 3.3 і тому відображає поведінку компонента за інтервал часу $[t_n, t_n + \Delta t]$,

де k_{lj}, $l \neq i$: **1** – компонент *i* віддає роботу w_j компоненту *l*,
0 – в іншому випадку;

k_{ij}: **-1** – компонент абсорбує (або здатен виконати) роботу w_j за проміжок часу $]t_n, t_n + \Delta t]$,
1 – компонент *i* віддає роботу w_j собі,
0 – компонент *i* не може виконати роботу w_j за даний проміжок часу $]t_n, t_n + \Delta t]$.

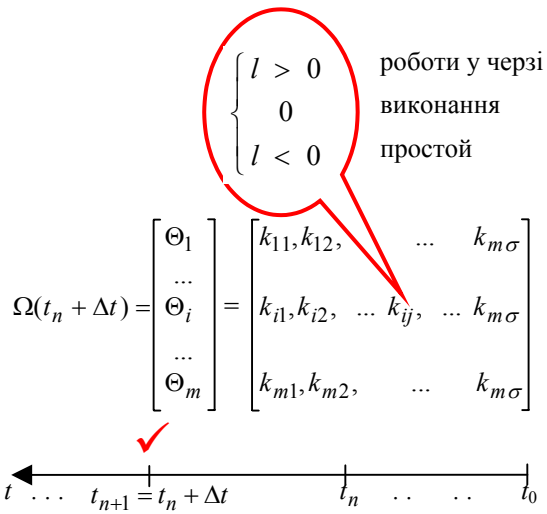


Рисунок 3.2 - Стан системи в $t_n+\Delta t$.

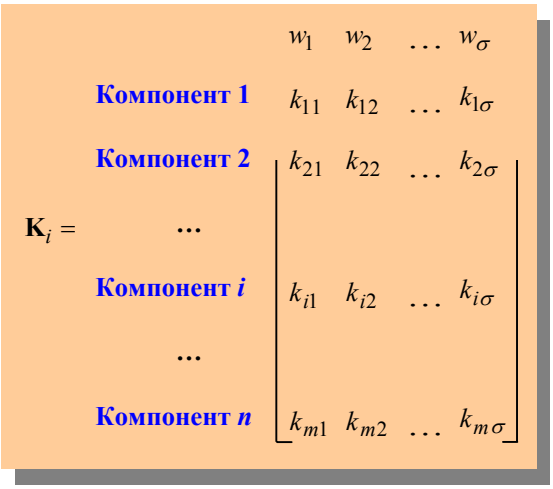


Рисунок 3.3 - Можливості і наміри компонентів в $]t_n,t_n+\Delta t]$.

Аналіз значень $\Omega(t_n)$ може забезпечити уніфіковану оцінку завантаженості компоненти, розподілення “ледачих” станів у кожний інтервал часу $]t_n,t_n+\Delta t]$, і тому можна буде розмірковувати про необхідність зміни поведінки компоненти $F_O^i(W)$ в майбутньому.

3.4 Узагальнена модель агента

На рівні агента середовище забезпечує ключові характеристики агента: розподіленість, автономность, раціональність і адаптовність. Агент приймає зовнішній вплив, перевіряє, чи узгоджується цей вплив із роллю агента і чи відповідає його поведінці, і виконує необхідну макромодельну програму – виконує або відхиляє атомарну роботу. Функція макромоделі полягає також у тому, щоб раціонально сформувати відгук, що містить отримані результати. Результати можна представити як функції від параметрів вхідного впливу.

Формально узагальнений агент є реактивним, раціональним, складається із сенсорного інтерфейсу, що сприймає зовнішній вплив, з каскаду із трьох скінченно-станових машин для верифікації вхідного впливу, локальної бази знань і блоку виконання макромоделі. Узагальнений агент є операційною оболонкою, що забезпечує кістяк будь-якого агента середовища. Агенти

спеціалізуються на базі множин макромодельних програм, що відповідають їхнім ролям. Макромодельні програми вважаються політиками агентів і зберігаються у її локальних базах знань.

Зроблено припущення, яке обгрунтовується прикладами, що приведени в розділі 3 даного звіту: кожний функціональний елемент (агент), діє за таким узагальненим алгоритмом:

```
DO  While .T.
    A = Wait(Action)      //Зреагувати на вхідний вплив
    Ỹ = Process(ActionEvent)//Обробити отриманий вплив
    IF (Requested)
        SendResults(A, Ỹ) //Повернути реакцію на вплив
    END IF
LOOP
```

де

- Wait() - процедура чекання події впливу;
- Process() - процедура обробки отриманого впливу, що виконує наступну послідовність дій:

  1) Перевірити свої повноваження (чи є право реагувати на отриманий вплив $a(f, X, Y)$ у поточному стані s_i);

  2) Перевірити відповідність отриманих параметрів $X = (x_1, ..., x_n)$ політиці f і граничним умовам стану s_i;

  3) Перевірити формальну відповідність вектора результату $Y = (y_1(X), ..., y_m(X))$ політиці f і стану $s_{i;}$

4) Виконати політику *f*, сформувати вектор результатів $\tilde{Y} = (\tilde{y}_1(X),...,\tilde{y}_k(X))$, змінити стан $s_i \rightarrow s_j$;

- SendResults() - процедура повернення реакції на оброблений вплив.

Структурна схема інтелектуального агента, що моделює поведінку функціонального елемента суспільства, приведена на Рис. 3.4. Кожний із блоків **1 - 3** являє собою скінченний автомат, на вхід якого подаються:

- для блока **1** - ланцюжок «символів», що описує зовнішній вплив;
- для блока **2** - ланцюжок «символів», що описує вхідні параметри зовнішнього впливу;

для блока **3** - ланцюжок «символів», що описує очікувані результати, що об'єкт, що робить вплив на наш агент, розраховує одержати в результаті виконання запитаної політики *f*.

В залежності від того, у яке із станів переходить кожний із кінцевих автоматів у результаті розбору вхідного ланцюжка, агентом приймається та або інша стратегія поведінки. Так при переході одним з автоматів **1-3** устан, що відхиляє вхідний ланцюжок, формується результат, який говорить про те, що даний агент *A* у даному стані s_i не може виконати політику *f* за однією із наступних причин:

- агент *A* в даному стані s_i не авторизований для виконання політики *f* - стан блока, що **відхиляє -- 1**;
- вхідні параметри $X = (x_1,..., x_n)$ не відповідають системним параметрам агента А, або не відповідають обмеженням на системні параметри агента *A*, що накладається станом s_i -- стан блока, що **відхиляє -- 2**;
- описи очікуваних результатів $Y = (y_1(X),..., y_m(X))$ не відповідають тим результатам, що може одержати агент А виконуючи політику *f* (наприклад, $y_1(X)$ містить ім'я файла, а $\tilde{y}_1(X)$ в результаті виконання політики *f*

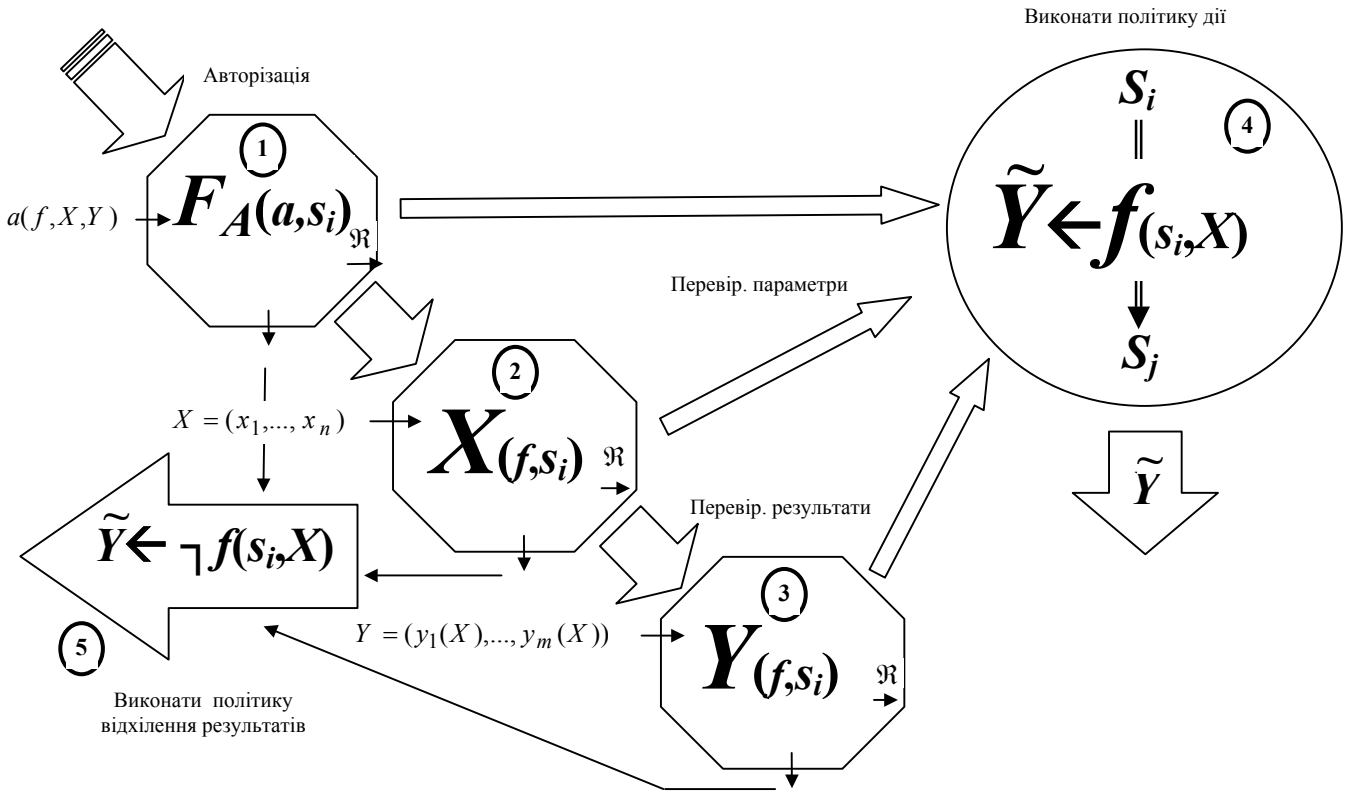


Рисунок 3.4 - Формальна узагальнена схема агента.

одержить значення, що представляє собою вектор (0, 0.25, 0.5, 0.75, 1)) - стан блока, що **відхиляє – 3.**

Блок **4** є виконавець, до функцій якого входить виконання політики *f*, запитаної зовнішнім впливом *a*. Зі схеми, приведеної на Рис. 3.4, випливає, що цей блок буде виконуватися тільки тоді, коли кожний із кінцевих автоматів **1 - 3** у результаті аналізу своїх вхідних ланцюжків перейде в стан , що дозволяє. Завданням блока **4** є виконання політики *f*, формування результату $\tilde{Y} = (\tilde{y}_1(X),...,\tilde{y}_k(X))$ і перевід агента *A*, у разі потреби, із стана s_i у стан s_j.

Блок **5** є тією частиною моделі, що формує відхіляючу реакцію агента А на *вплив a*. реакція агента, що відхиляє вплив, залежить від від *політики f*, запитаної впливом а, стану s_i, *параметрів* X, Y і далеко не завжди **є фактично негативною**.

Виходячи з вищевикладеного, формальна модель агента може бути подана в такий спосіб:

$$A = \{F_a, F_X, F_Y, S, X_A, F, F_O\}, \tag{3.5}$$

де: $S_A = \{s_1,..., s_n\}$ - множина станів агента *A;*

$X_A = \{x_1,..., x_p\}$ - множина системних параметрів агента *A;*

$F_a = F_a(a, F, s_i) = \{\aleph_a, \wp_a(s_i), \Re_a, \delta_a(s_i)\}$ - скінченний автомат, який здійснює авторизацію політики зовнішнього впливу *a* в стані $s_i \in S$. Вхідна послідовність *a* записується у «символах» вхідного алфавіту $\aleph_a$. Множина станів скінченного автомата $\wp_a$ є функція від стану s_i агента *A* і елементи множини станів залежать від стану s_i агента *A*. Множина станів, що $\Re_a$ дозволяють, складається з одного елемента - стану, що позначений на мал.3.4. символом $\Re$. Функція переходів δ також залежить від стану s_i агента *A;*

$F_X = F_X(X, X_A, s_i) = \{\aleph_X, \wp_X(X_A, s_i), \Re_X, \delta_X(X_A, s_i)\}$ - скінчений автомат, що здійснює перевірку відповідності параметрів *X* отриманого впливу *a* на відповідність множині системних параметрів X_A і обмеженням стану s_i;

$F_Y = F_Y(Y, f, s_i) = \{\aleph_Y(f), \wp_Y(f, s_i), \Re_Y(f), \delta_Y(f, s_i)\}$ - скінчений автомат, що здійснює перевірку відповідності опису очікуваних результатів *Y* тим результатам, які можуть бути отримані при виконанні агентом *A* в стані s_i політики f;

$F = \{f_1,..., f_i,..., f_m\}$ - роль агента - множина політик, таких, що виконує агент *A*.

F_O - програмна компонента, що виконує політику *f ;*

$f \in F$ - політика дії - набір атомарних інструкцій, поданий у вигляді програми, що містить розгалуження. Подібні програми описані, наприклад у [20].

Структура політики дії, на наш погляд, є такою:

```
DO CASE
     CASE State = s1
```

```
                StartThread(<Id1>)
                BEGIN
                Execute <Instruction> on (X̃_i) results to (Ỹ_i)
                Broadcast <Action> with (X̃_i, Ỹ_i) and wait for (Ỹ_i)
                     from each party
                OptimiseResults()
                Require Aj  with (< Action >, X̃_i, Ỹ_i) and wait for (Ỹ_i)
                Invoke Aj  with (< Action >, X̃_i, Ỹ_i)
                ChangeState(s1)
                END
                CommitThread(<Id1>)
                .....
                StartThread(<Idk>)
                BEGIN
.....
                END
                CommitThread(<Idk>)
.....
        CASE State = sn
                .........
        OTHERWISE
                .........

   END CASE
```

де $<Id_i>$ - ідентіфікатор потока;

<Instruction> - інструкція, що виконує атомарну дію в межах даній політики

без залучення інших агентів;

<Action> - ідентифікатор впливу, що визначає частину політики дії, що не може бути виконана даним агентом. Для впливу *<Action>* може бути використаний один із наступних методів виконання:

1) **Broadcast** <*Action*> для випадку, коли авторізований для <*Action*> агент не відомий. У цьому випадку вплив <*Action*> розповсюджується на всі члени співтовариства.

2) **Require** *Aj* with $(< Action >, \tilde{X}_i, \tilde{Y}_i)$ для випадку, коли авторізований для <*Action*> агент відомий. Цей спосіб моделює випадок синхронної взаємодії з агентом *Aj*. Результати, повернуті агентом *Aj*, складуть частину результатів виконуваній політики.

3) **Invoke** *Aj* with $(< Action >, \tilde{X}_i, \tilde{Y}_i)$ для випадку, коли авторізований для <*Action*> агент відомий, і нам не важлива його реакція на вплив <*Action*>. У цьому випадку вектор значень, що повертаються, вірогідніше усього буде порожнім. Інтерфейс примітива **Invoke** містить параметр $\tilde{Y}_i$ на той випадок, якщо в майбутньому буде реалізований механізм передачі результатів третій стороні A_m.

$$X = \bigcup_i \tilde{X}_i, \quad Y = \bigcup_i \tilde{Y}_i$$

Такі впливи реалізуються за допомогою механізму комунікації агентів, формальна модель якого презентується в розділі 2 цього звіту.

3.5 Архітектура узагальненого агента

Розроблена архітектура узагальненого агента представлена на Рис. 3.5. Узагальнений агент має у своєму складі наступні компоненти:

- блок комунікації, що включає інтерфейсні компоненти для комунікації з користувачем та іншими агентами, процесор повідомлень і процесор змісту повідомлень;
- блок верифікації отриманого завдання, чи роботи, який включає компоненти перевірки політики, отриманих параметрів і шаблонів очікуваних результатів;
- блок обробки запитів до компоненти, яка є відповідальною за знання та досвід агента;

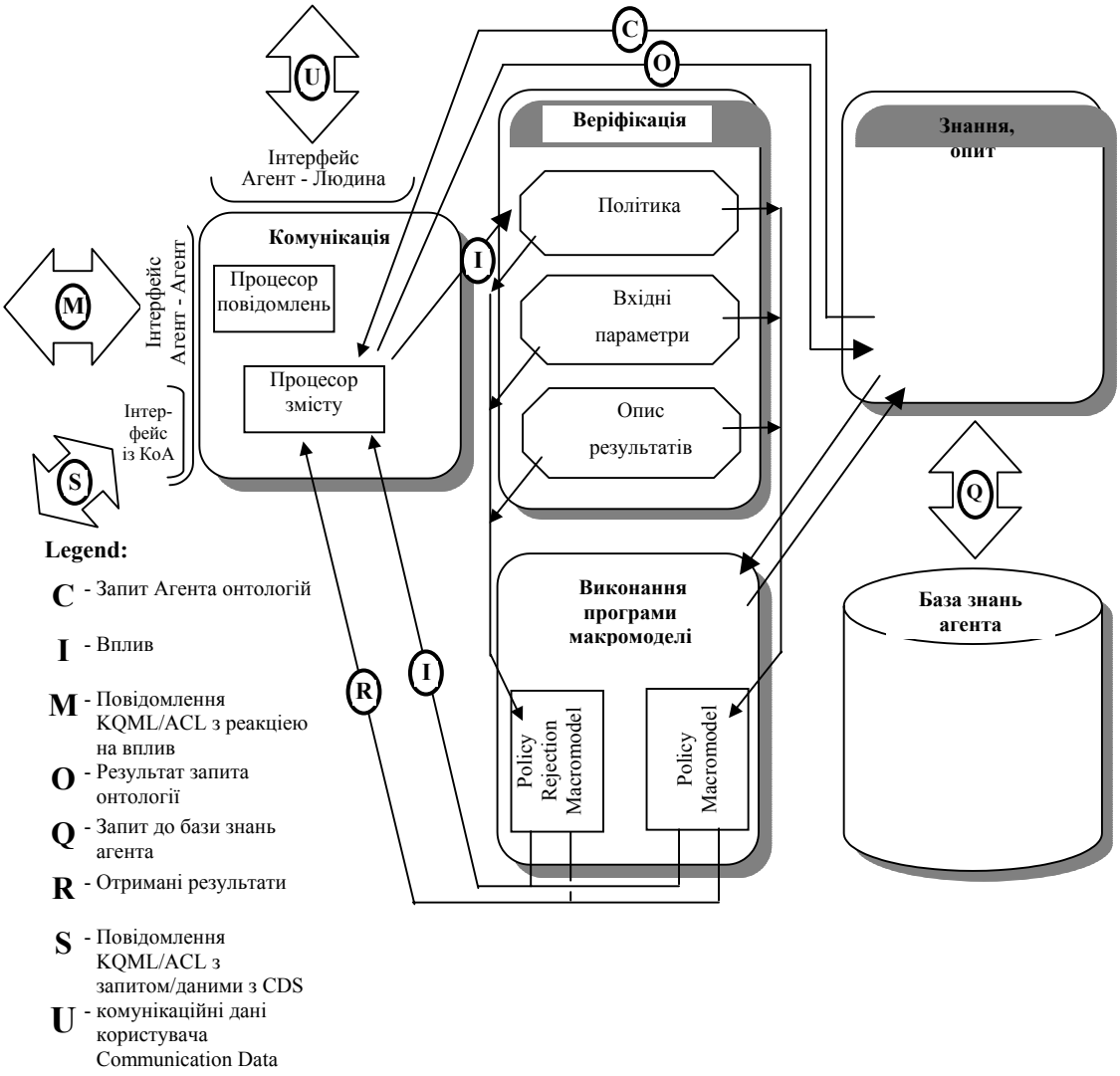


Рисунок 3.5 – Архітектура узагальненого агента.

– локальну базу знань агента;
– блок виконання програм макромоделей.

3.6 Підсумки

Розробка моделей функціональних об'єктів і методів їхньої взаємодії доцільна на основі застосування діакоптичного підходу для математичного моделювання.

Запропоновані вище математичні моделі агентів і завдань, які вони виконують, дозволяють організувати процес моделювання практично довільної

функціональної системи, опис якої можливо здійснювати в термінах «завдання», «робота» і т.і.

4 Формальні принципи і моделі взаємодії функціональних об'єктів ЄІП

Функціональна проекція VIS населена МАС, що представляють функціональні системи і компоненти на різних рівнях. Агенти – учасники динамічно формують проблемно-орієнтовні суспільства для виконання завдань отримання інформації, інтеграції, посередництва і обміну, що з’являються у рамках МАС на одному чи іншому рівні.

В реальному житті такі завдання розділяються і виконуються групами функціональних компонентів. Традиційні моделі процесів інформаційного обміну часто базуються на структурованих наборах жорстких відношень між функціональними вузлами. При моделюванні з використанням більшості середовищ вищеназвані завдання (потоки робіт) представляють як плани – наперед задані орієнтовані графи, мережі Петрі [26] та ін. Такі підходи часто є надто статичними. Наприклад, розглянемо процес обговорювання результатів лабораторних робіт (з курсу хімії). Такий процес не завжди можна промоделювати у вигляді ієрархії жорстко розміщених акторів. Люди часто використовують більш м'які відносини: мозковий штурм, неформальна дискусія, тощо. Друга вада традиційних підходів це то, що жорсткі моделі відношень не є масштабованими. Велика перевага моделі виконання плану [17], що використана у нашому середовищі, – це відсутність статичних наперед заданих специфікацій завдань.

Завдання, в рамках нашого підходу, “викликаються” агентом-посередником і виконуються агентами середнього рівня. Агенти середнього рівня динамічно формують суспільства для виконання наявних завдань. Агент приєднується до суспільства тоді і тільки тоді, коли він сприймає вхідні дані, що містять (під)множину атомарних робіт (частину завдання) для виконання. План виконання завдання уточнюється в подробицях під час процесу покрокового виконання завдання. Процес впроваджується командою агентів-учасників, діючих у кооперації один з одним. Агент-координатор виконує функції

координації дії команди агентів і відображає діяльність кожної команди. Агенти – учасники суспільства діють як моделі функціональних компонентів відповідних об’єктів реального світу.

Ключовими проблемами в організації взаємодії агентів, як автономних істот, є реалізація засобів передачі впливів і повідомлень (комунікації), а також узгодження спільних робіт (координації) між агентами що діють спільно для вирішення спільної проблеми, або виконання спільного завдання.

4.1 Принципи і методи організації взаємодії агентів

Взаємодія – одна з найважливіших характеристик агентів. Агенти взаємодіють для того, щоб спільно використовувати інформацію, виконувати завдання, досягати спільних цілей.

Генезерет і Кетчпел [27] стверджують, що здатність зв’язуватися на мові взаємодії агентів (ACL) – єдина характеристика, яка відрізняє агентів від іншого програмного забезпечення. Механізми зв’язку накладають додаткові обмеження на внутрішню структуру агенту, яка повинна їх ефективно використовувати [27].

В [28] виділяється три типи взаємодії, які можуть використовуватися в МАС:

- Непрямий обмін повідомленнями (indirect message passing);
- Пряма взаємодія, із використанням API або Remote Procedure Call (RPC);
- Використання загальнодоступної пам’яті, наприклад, дошки оголошень.

Вони стверджують, що взаємодія агентів за допомогою обміну повідомленнями буде більш гнучкою, ніж загальнодоступна пам’ять (із проблемами паралельного доступу і семафорами) і RPC (який обмежує гнучкість під час виконання), тому що вона дозволяє агентам вільно зв’язуватися і бути розподіленими. Крім того, цей вид взаємодії забезпечує гнучкість щодо ліній зв’язку.

4.1.1 Засоби комунікації агентів

До засобів взаємодії агентів відносять мови взаємодії агентів (FIPA ACL, KQML).

Мова взаємодії агентів забезпечує обмін знаннями і інформацією між агентами. FIPA ACL, на відміну від таких засобів як RPC, RMI, CORBA, що забезпечують обмін інформацією між прикладними задачами, має більш складну семантику [29]. Крім того, ACL має такі переваги в порівнянні із CORBA[30]:

– FIPA ACL управляє судженнями, правилами, діями, а не семантично незв’язаними об’єктами.
– Повідомлення FIPA ACL описує очікуваний стан, а не процедуру або метод.

Однак ACL не охоплює повного спектру об'єктів, якими могли б обмінюватися агенти, наприклад плани, цілі, досвід, стратегії.

На технічному рівні, при використанні ACL, агенти транспортують повідомлення по мережі, використовуючи протоколи нижчого рівня, наприклад SMTP, TCP/IP, POP3, або HTTP.

Мова взаємодії агентів (ACL) повинна дозволяти передавати інформацію будь-якого виду між різними агентами. Є два підходи до проектування мов взаємодії агентів:

– ***Процедурний***, включає обмін процедурними директивами/командами. Це може бути реалізовано за допомогою таких мови програмування як Java або Tcl.
– ***Декларативний***, де зв'язок заснований на декларативних інструкціях, типу визначень, припущень, знань, і т.п.

Через обмеження на процедурні підходи (наприклад, такі сценарії важко координувати, об'єднувати), були використані декларативні мови для створення мов взаємодії агентів. Одними з найбільш популярних декларативних мов є KQML [31] із своїми діалектами і FIPA ACL [27].

4.1.2 Методи координації в МАС

Як помічає К.Сікара [32], вивчення мультиагентних систем зосереджується на системах, в яких багато інтелектуальних агентів взаємодіють один із одним. Агенти, як вважається, є автономними об'єктами, типу програм або роботів. Їх взаємодії можуть бути або кооперативними, або егоїстичними. Тобто, агенти можуть разом досягати однієї мети (як у колонії мурашів), або вони можуть переслідувати власні інтереси (як у вільній ринковій економіці). Тому необхідно реалізовувати кожного мураша в колонії так, щоб примусити їх разом приносити їжу найбільш ефективним способом, або встановити такі правила взаємодії, щоб група егоїстичних агентів працювала разом для виконання конкретного завдання.

Таким чином, координація - центральна проблема в МАС зокрема, й у розподіленому штучному інтелекті (DAI) взагалі. Власне кажучи, координація - *процес,* у якому агенти беруть участь, щоб забезпечити суспільству несуперечливу, погоджену послідовність дій. Це означає, що дії агентів сплановані так, що вони не конфліктують один з одним, а все суспільство поводиться як єдине ціле. Є декілька причин, за якими МАС повинна бути скоординована [31]:

- *Запобігання анархії або хаосу*: координація необхідна або бажана, тому що в децентралізованої MAS може легко встановитися анархія;
- *Встановлення глобальних обмежень*: завжди існують глобальні обмеження, що МАС повинна задовольнити, щоб успішно виконати завдання;
- *Розподілені знання, ресурси, інформація*: практично завжди в МАС є ресурси, робота з якими вимагає координації;
- Залежність між діями агентів: цілі агентів часто взаємозалежні;
- *Ефективність*: навіть коли агенти можуть працювати незалежно, координація дозволяє заощадити час.

Існує багато підходів забезпечення координації в МАС. Усі ці підходи можна розділити на чотири категорії [33]:

- організаційне структурування;
- укладання контракту;
- планування діяльності МАС;
- переговори.

Організаційне структурування: Це найпростіший метод координації, що складається із визначених і довгострокових відношень між агентами (див. огляд [33]) . При цьому часто використовують ієрархічні структури *master-slave* або *client-server*. Ця методика використовується в двох варіантах:

- Головний агент планує і розподіляє завдання (роботи) між підпорядкованими агентами. Підпорядковані агенти, врешті-решт, повинні повідомити головному агенту про результати їхньої роботи. Крім того, на відміну від головного агента, підпорядковані агенти мають часткову автономність.
- Використання класичної дошки оголошень - загального інформаційного поля - для забезпечення координації. У цій схемі агенти використовують дошку для обміну інформацією. Головний агент (планувальник) призначає агентам операції читання/запису з дошкою. Ця схема використовується Веркманом у його DFI системі (див. огляд [33]). Цей підхід може використовуватися, коли завдання розподілене, є центральний агент планування або коли завдання вже були призначені, *апріорно,* агентам. Sharp Multi-Agent Kernel (SMAK) (див. огляд [33]) теж використовує цю стратегію.

Останній варіант показує, що координація в організаційному структуруванні не завжди зв'язана з ієрархією. Наприклад, у системі DVMT (див. огляд [31]), що використовує технологію дошки оголошень, координація проходить серед рівних агентів.

Вузьким місцем у системах, заснованих на технології дошки без прямої взаємодії агентів, може бути велика кількість агентів у суспільстві, навіть у випадку дошки, розділеної на секції. Крім того, всі агенти повинні мати загальне уявлення про дошку. Тому більшість систем, заснованих на цій

технології, використовують невеликих гомогенних агентів (як, наприклад DVMT).

Дурфи (див. огляд [33]) вважає, що таке централізоване управління, як в ієрархічній методиці, суперечить основним припущенням про DAI. У цій методиці передбачається, що, принаймні, один агент має глобальне представлення всього суспільства, однак у багатьох областях це не реально. Якщо все-таки ця методика координації використовується, проектувальник повинен гарантувати, що всі підпорядковані агенти мають оптимальний ступінь деталізації щоб час, витрачений на декомпозицію загального завдання на більш дрібні, не перевищило часу її виконання одним агентом.

Укладання контракту: У цьому підході, що припускає децентралізовану структуру, агенти приймають дві ролі:

- менеджер, що поділяє вхідне завдання на підзавдання і шукає виконавця, щоб виконати їх;
- виконавець, що виконує підзавдання. Виконавець може рекурсивно стати менеджером і розбити вхідну підзавдання на ще більш дрібні завдання і доручити їх іншим агентам.

Пошук виконавця виглядає у такий спосіб (FIPA CNP [27]):

Менеджер повідомляє про наявність вхідного завдання;

Виконавці оцінюють це завдання з погляду можливості її виконати;

Менеджер одержує таблицю виконавців;

Оцінює отримані пропозиції, вибирає виконавця і доручає виконати йому завдання;

Чекає результатів виконання завдання.

Це цілком розподілена схема, в якій кожний агент може бути як менеджером, так і виконавцем. Цей підхід застосовується в багатьох прикладних задачах.

Хухнс і Сингх (див. огляд [33]) звертають увагу на те, що ця модель координації забезпечує розподіл загального завдання, і засоби для самоорганізації групи агентів, і найкраще застосовна коли:

- прикладне завдання має чіткий ієрархічний характер (природу);
- прикладне завдання ділене на великі підзадачи;
- є мінімальний взаємозв'язок серед підзадач.

Переваги цього підходу координації є такими: динамічний розподіл завдання завдяки пошуку оптимального виконавця; збалансоване завантаження агентів (зайняті агенти можуть не брати участь у виконанні загального завдання); і надійний механізм для розподіленого керування і відновлення результатів при збої.

Агенти в такій системі досить пасивні і не застосовні для багатьох прикладних завдань. Нарешті, таке суспільство агентів досить інтенсивно використовує мережа, що може знизити усі його переваги.

Планування діяльності МАС. Є два типи планування діяльності МАС:

- централізоване планування діяльності МАС;
- розподілене планування діяльності МАС.

У централізованому плануванні діяльності МАС звичайно є координаційний агент, що, отримавши всі індивідуальні плани від інших агентів, аналізує їх, щоб виявити потенційні протиріччя і конфлікти у взаємодії агентів (наприклад, конфлікт між агентами по використанню загального ресурсу). Потім координаційний агент намагається змінити ці індивідуальні плани й об'єднує їх у план МАС, в якому суперечливі взаємодії усунуті. У заключному плані МАС, додані команди зв'язку, щоб синхронізувати взаємодії агентів у потенційно можливих конфліктах.

У розподіленому плануванні діяльності МАС, ідея полягає в тому, щоб забезпечити кожного агента моделлю планів інших агентів. Агенти взаємодіють, щоб створювати і модифікувати їхні індивідуальні плани, що не конфліктують із планами інших агентів.

Координація в розподіленому плануванні діяльності МАС - набагато складніше, чим у централізованій формі, тому що там не існує агентів, що володіють глобальним представленням про всю розподілену систему.

Переговори. Це один із самих складних методів координації. Будь-яка форма взаємодії між людьми вимагає деякою мірою явних або неявних переговорів. Тому, не вражає той факт, що багато дослідників переговорів як методу координації застосовують саме людські стратегії ведення переговорів. Крім того, при координації агентів цим підходом часто використовують різні методи ИИ, логіки, випадково заснованого міркування, спрямованого обмеженням пошуку, і т.д.

Важливим завданням цього розділу звіту є розробка моделей комунікації і координації агентів, що формують суспільства і виконують завдання (моделюють діяльність реальних функціональних компонент і систем ВНЗ) в ЄІП. Функціональна проекція ЄІП, як вже зазначалося, населена МАС, що представляють функціональні системи і компоненти на різних рівнях. Агенти – учасники динамічно формують проблемно-орієнтовані суспільства для виконання завдань одержання інформації, інтеграції, посередництва й обміну, що з’являються у рамках МАС на однім або другому рівні.

Велика перевага моделі виконання плану [17] використана в нашому середовищі – це відсутність статичних наперед заданих специфікацій завдань.

Завдання, у рамках нашого підходу, “викликаються” агентом-посередником і виконуються агентами середнього рівня. Агенти середнього рівня динамічно формують суспільства для виконання завдань. Агент приєднується до суспільства тоді і тільки тоді, коли він сприймає вхідні дані, що містять (під)множину атомарних робіт (частина завдання) для виконання. Агенти обмінюються впливами (завданнями) як передбачено моделлю комунікації (див. п. 4.2.). План виконання завдання уточнюється в подробицях під час процесу покрокового виконання завдання. Процес впроваджується командою агентів-учасників, що діють у кооперації один з одним. Агент-координатор

виконує функції координації дії команди агентів і відображає діяльність кожної команди. Модель координації наведена в п. 4.3. Агенти – учасники суспільства діють як моделі функціональних компонентів відповідних об'єктів реального світу.

4.2 Модель комунікації агентів, що репрезентують функціональні компоненти ЄІП

На рівні комунікації вважається, що агенти, які приймають участь в процесі виконання завдання, спілкуються за допомогою таких актів спілкування:

- *директива* – безумовне виконання дії підпорядкованим агентом (Рис. 4.1).
- *детермінований запит із детермінованою реакцією* – надсилаючи повідомлення агент хоче, що б йому повернули деякі результати (Рис. 4.2).
- *детермінований запит із оптимізацією результату* – після одержання результатів запиту, агент, перед використанням результатів, повинний їх спочатку оптимізувати (Рис. 4.3).
- *недетермінований запит із оптимізацією результату* – використовується, коли заздалегідь не відомий агент-одержувач повідомлення. Після того як повідомлення буде послано всім агентам усередині системи й отримані від кожного такого агента результати, буде обрано кращого агента - виконавця (Рис. 4.4).

Якщо представити зовнішній вплив на суспільство і реакцію агента на цей вплив у вигляді наступного співвідношення:

$$\widetilde{Y} = a(f, X, Y) \qquad (4.1)$$

де $a(f, X, Y)$ - вплив, f - політика, $X = (x_1, ..., x_n)$ - параметри;

$Y = (y_1(X), ... y_m(X))$ - опис запитуваних результатів;

$\widetilde{Y} = (\widetilde{y}_1(X), ..., \widetilde{y}_k(X))$ - реакція (результати);

і, крім того, взаємодія агентів усередині співтовариства, у вигляді:

$$\widetilde{Y}^* = a^*(f^*, X^*, Y^*) \quad (4.2)$$

то- для впливу виду «директива» буде справедливо:

$$X^* \subseteq X,\ Y^* = \widetilde{Y}^* = \emptyset,\ f^* \subseteq f; \quad (4.3)$$

- для впливу виду «детермінований запит із детермінованою реакцією» буде вірно:

$$X^* \subseteq X,\ Y^* = \widetilde{Y}^*,\ f^* \subseteq f; \quad (4.4)$$

- для впливу виду «детермінований запит із оптимізацією результату» буде вірно:

$$X^* \subseteq X,\ Y^* \subseteq \widetilde{Y}^*, f^* \subseteq f; \quad (4.5)$$

- для впливу виду «недетермінований запит з оптимізацією результату» буде вірно:

$$X^* = X,\ Y^* = \widetilde{Y}^*, f^* \equiv f; \quad (4.6)$$

Наведені рівняння дозволяють описати будь-який вплив на програмного агента в мультиагентной системі, що виконує завдання (див розділ 3).

4.3 Модель координації виконання множин атомарних робіт

Використовуючи результати розділу 3, диакоптичний підхід до моделювання суспільств агентів, що описаний у [16,17], побудуємо модель координації агентів, засновану на активній дошці оголошень.

Використання активної дошки оголошень відрізняється від класичного підходу координації лише тим, що всі операції читання/запису з загальним полем пам'яті робить тільки один агент - координаційний агент, що, по суті, інкапсулює у самому собі дошку оголошень, і забезпечує методи роботи з нею. Інші агенти, при необхідності прочитати або записати на дошку деяку

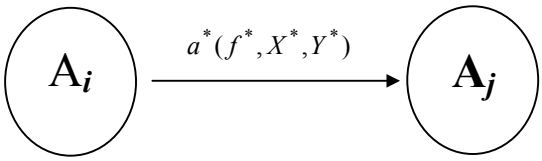


Рис. 4.1 – Директива

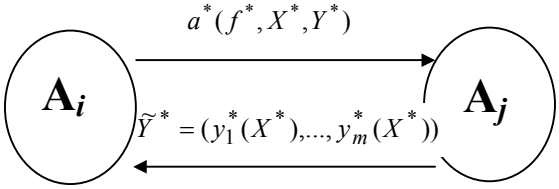


Рисунок 4.2 – Детермінований запит із детермінованою реакцією

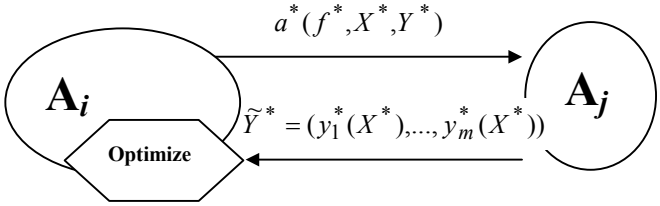


Рисунок 4.3 – Детермінований запит із оптимізацією результата

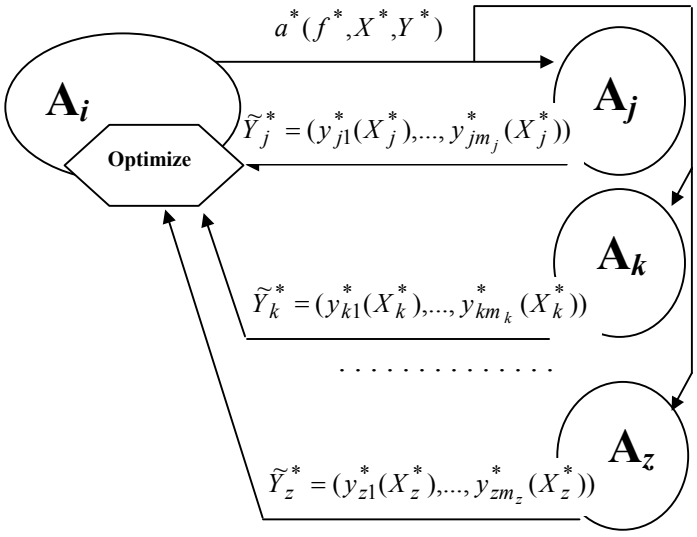


Рисунок 4.4 – Недетермінований запит із оптимізацією результатів

інформацію, повинні взаємодіяти з координаційним агентом за допомогою ACL/KQML повідомлень.

Щоб більш докладно описати модель координації, послідовність дій агентів, взаємодію агентів суспільства з координаційним агентом, повернемося до приклада з підрозділа 3.1. [34].

Для простоти розглянемо лише процес виконання агентом **PM** роботи w_2 (див. Рис. 4.5).

Робота w_2 надходить до агента **PM** у момент часу t_1. Однак вона не може бути виконана за час $]t_1, t_1 + \Delta t]$, тому що для її виконання необхідний результат роботи w_6 агентом **DBA,** що потрібно запросити у координаційного агента - **COA**. Макромодель $F_O^{PM}(w_2)$ виконання роботи w_2 агентом **PM** може бути подана наступним алгоритмом:

```
...
 Avail := Require("COA" with ("ProvideResults", Ỹ(w6) ))
IF .NOT. Avail THEN
    Redirect w2 TO PM  // w2 перенаправляеться PM на
                              //наступний момент часу t1 + Δt .
      ELSE
    Execute( w2 )  //виконання w2 агентом PM (див. Рис. 2.5)
    END IF
...
```

У цей же момент часу t_1 до агента **DBA** надходить робота w_6. Агент **DBA** виконує роботу w_6 в інтервалі часу $]t_1, t_1 + \Delta t]$ і відсилає результат $\tilde{Y}(w_6)$ координаційному агенту. Це може бути описане в такий спосіб:

```
...
    Ỹ := Execute( w6 )       //виконання роботи w6
    Invoke("COA" with ("AcceptResults", Ỹ ))     //відправлення
                                        // Ỹ агенту COA
...
```

Реакція координаційного агента **COA** на отримане від агента **DBA** вплив типу **AcceptResults** має такий вид:

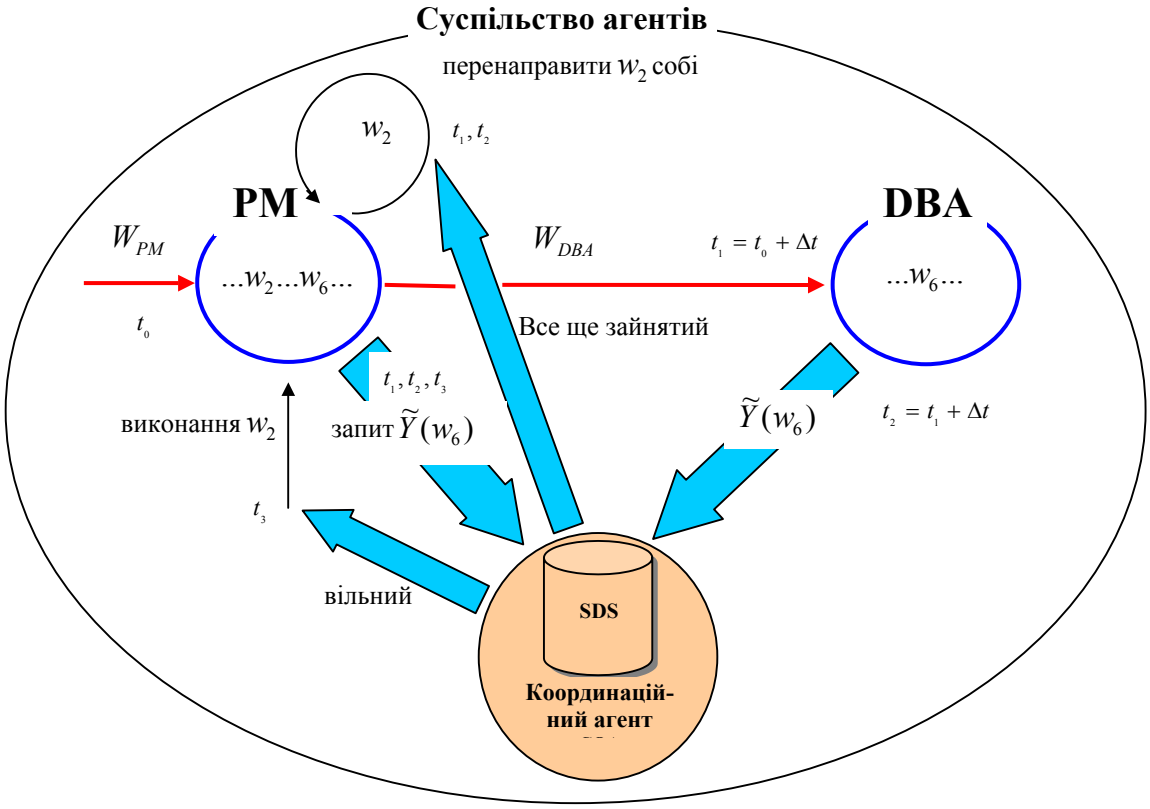


Рисунок 4.5 – Модель координації.

```
...
   Execute("AcceptResults", DBA, Ỹ )
...
```

Реакція **COA** у відповідь на запит **PM** результатів роботи w_6, може бути описана у вигляді наступного алгоритму:

```
...
 Avail := Execute("ProvideResults", Ỹ(w6))
IF .NOT. Avail THEN
   Reply("PM" with ("AcceptResults", "DUMMY"))
 ELSE
   Reply("PM" with ("AcceptResults", Ỹ(w6)))
 END IF
...
```

Таким чином, у момент часу t_2 реакція **COA** буде відхилена, тому що результатів роботи w_6 ще ні, а в момент часу t_3, запит агента **PM** буде задоволений.

Необхідно відзначити, що результати виконаних робіт, що передаються між агентами, повинні бути подані в деякому універсальному форматі, «зрозумілому» всім агентам. Таким форматом може бути KIF (Knowledge Interchange Format) [35] - формат обміну знаннями - це формальна мова для обміну знаннями між несумісними комп'ютерними програмами, написаними різними програмістами, у різний час, на різних мовах і т.д.

Повертаючись до приклада з [34], запишемо на мові KQML[31] повідомлення, що забезпечують взаємодію з координаційним агентом.

KQML-повідомлення для відсилання результатів виконаної роботи координаційному агенту може мати такий вид:

```
(ask-one
        :sender         "DBA"
        :receiver       "COA"
        :in-reply-to    Null
        :reply-with     Null
        :language       KIF
        :ontology       COA ontology
        :contents       ("AcceptResults", $\tilde{Y}(w_6)$ ) )
```

KQML повідомлення для запиту результатів роботи у координаційного агента:

```
(ask-one
        :sender         "PM"
        :receiver       "COA"
```

:in-reply-to Null

:reply-with $\widetilde{Y}(w_6)$

:language KIF

:ontology COA ontology

:contents ("ProvideResults", $\widetilde{Y}(w_6)$))

KQML повідомлення - повернення результатів координаційним агентом:

(tell

:sender "COA"

:receiver "PM"

:in-reply-to Null

:reply-with Null

:language KIF

:ontology COA ontology

:contents ("AcceptResults", $\widetilde{Y}(w_6)$))

Таким чином, основними функціями координаційного агента є одержати, деякий час зберігати й відсилати по запиті проміжні результати виконання робіт у співтоваристві.

З огляду на головну функцію координаційного агента в співтоваристві - зберігати проміжні результати робіт, можна сказати, що основними припустимими типами зовнішніх впливів, на які повинний реагувати координаційний агент є:

А) одержання результату деякої роботи;

Б) запит результату деякої роботи.

Далі в цій роботі одержання координаційним агентом результату деякої роботи будемо називати *зовнішнім впливом першого типу*, а запит у

координаційного агента результатів деякої роботи - *зовнішнім впливом другого типу*.

У термінах моделі комунікації [16,17], див. підрозділ 4.2., зовнішній вплив першого типу буде директивою, тому що від координаційного агента потрібно лише прийняти результат; зовнішній вплив другого типу буде детермінованим запитом із детермінованою реакцією, тому що координаційний агент повинний повернути запитуваний результат або повідомити, що такого результату ще немає.

Очевидним є той факт, що ефективність роботи всього співтовариства залежить від того, наскільки швидко координаційний агент буде реагувати на зовнішні впливи. Іншими словами критичним параметром роботи координаційного агента є час реакції. Цей висновок дозволяє накласти деякі обмеження на архітектуру самого координаційного агента; на макромоделі, що описують реакцію агента на зовнішні впливи; на організацію збереження даних - результатів робіт. Тому наступні підрозділи другого розділу будуть присвячені цим проблемам.

4.4 Модель еволюції функціональних компонентів в МАС

Одна із найголовніших характеристик деякої функціональної компоненти (і ЄІП у цілому) - це його здатність змінюватися. Середовище для моделювання ЄІП повинно мати засоби для роботи із змінами, що з’являються у реальному світі – треба використовувати моделі, що здатні підтримувати різні типи еволюції. С точку зору даного дослідження, під еволюцією розуміється процес проактивного саморозвитку і самоадаптації інтелектуальних активних компонентів (агентів) відповідно до змін у тому середовищі, де вони мешкають – тобто, у ЄІП.

Згідно з моделлю еволюції [36], що застосована у ЄІП, дане середовище підтримує :

- Зміни у обмеженнях на стан агента – *здатності* до виконання роботи

- Зміни у концептуалізаціях (уявленнях) агента про своїх партнерів – учасників суспільства виконавців завдання
- Зміни у *інформаційних ресурсах і їх метаданих*

Еволюція здатностей. Згідно із [16] еволюція здатностей – це процес переходу агента (скажімо, A) із одного стану s_i в інший s_j. Агент A як автономний об'єкт, виконує ці переходи згідно із рішеннями, які він сам прийняв під час виконання тієї чи іншої атомарної роботи. Внаслідок цього "манера" виконання агентом A політики f, а також обмеження на політику вхідних параметрів X_f, залежить від стану агента A. Тому, еволюція агента – це еволюція його ролі.

Множина станів агента A: $S_A = \{s_1,...,s_n\}$ - це множина триплетів $s_i, i = 1,...,n$:

$$s_i = \{r(X_A), q(F_A), t(F)\} \qquad (4.7)$$

де $r(X_A)$ - множина обмежень, що застосовуються в стані s_i до системних параметрів X_A агента A (обмеження на параметри);

$q(F_A)$ - множина обмежень в стані s_i до авторизованих політик агента A (обмеження на політики);

$t(F)$ - функція, що визначає переходи зі стану s_i до інших дозволених станів із S_A, які з'являються після виконання агентом A політик $F = \{f_1,..., f_j,..., f_m\}$.

Еволюція уявлень. Еволюція уявлень близько пов'язана з обстеженням можливостей учасників суспільства по виконанню завдання. Згідно з [16, 17] комунікація між агентами і виконання роботи організована через параметричні відгуки, що містять інформацію про наявні можливості щодо виконання конкретної роботи. Можливість, що повертається виконавцем A користувачу B, це функція від параметрів завдання (політики) $c_A^f = c(X_f)$, $c_A^f \in [0,1]$. Агент (який представляє функціональний компонент реального світу) обстежує

можливості своїх співробітників для того щоб назначити роботи виконавцям із вірогідно кращими можливостями у майбутньому. Уявлення про вірогідні можливості агентів-співробітників мають вигляд матриці:

$$\mathbf{C} = \begin{bmatrix} & w_1 & \dots & w_j & \dots & w_m \\ A_1 & c_1^1 & & c_1^j & & c_1^m \\ \dots & & & \dots & & \\ A_i & & \dots & c_i^j = c_{A_i}^{w_j}(X_{w_j}) & \dots & \\ \dots & & & \dots & & \\ A_n & c_n^1 & & c_n^j & & c_n^m \end{bmatrix} \qquad (2.8)$$

де розмірності n і m зростають в процесі еволюції, відображаючи появлення нових знань про агентів-співробітників (n) і про роботи, які вони, можливо, зможуть виконувати. Верхня границя розмірності n - це кількість агентів-учасників МАС, яку контролює володар матриці **C**. Максимальне значення m - це розмірність множини W дозволених атомарних робіт вищеназваної МАС.

Зміни у інформаційних ресурсах. Зміни в даних і метаданих інформаційних ресурсів підтримуються локально відповідними розподіленими інформаційними системами – постачальниками інформації. В рамках нашого підходу постачальників інформаційних ресурсів представляють агенти – оболонки, які еволюціонують відповідно до змін.

4.5 Підсумки

В даному розділі представлені розроблені моделі і методи взаємодії автономних інтелектуальних компонент (агентів), що діють в функціональній проекції ЄІП. Розв'язаними є ключові проблеми організації взаємодії агентів, як автономних істот: реалізація засобів передачі впливів і повідомлень (комунікації), узгодження спільних робіт (координації) між агентами що діють

спільно для вирішення спільної проблеми, або виконання спільного завдання, здатності агентів постійно поновлювати свої уявлення про партнерів (еволюціонувати) у процесі спільної роботи.

5 Формування коаліцій для виконання завдання на базі торгів

Завдання, що виконуються організаціями фізичних, так само як і штучних акторів, можуть бути охарактеризовані властивостями розподіленості і невпевненності. Розподіленість означає, що різні частини завдання виконуються різними виконавцями. Такі виконавці є головним чином егоїстичними і діють відносно автономно, приймаючи власні рішення і аналізуючи наслідки. Такі актори відрізняються один від одного своїми можливостями до виконання певним діяльностей, обмеженнями ресурсів, своїми станами і відповідними обмеженнями на стани, своїми уявленнями про зовнішнє середовище і про своїх партнерів, а також за власними намірами, цілями і пріоритетами. Інакше, гравці, які взаємодіють в межах організації, шукають співвиконавців для того, щоб виконати свою роботу найбільш оптимальним чином, хоча такі коаліції не є визначеними повною мірою. Невизначеність також має декілька вимірів. Актори суб'єктивно не є впевненими щодо узгоджень і намірів, або, на більш високому рівні, щодо можливої поведінки інших акторів. Також невизначеною є суб'єктивна готовність до кооперативної праці, суб'єктивна оцінка передбачуваності і довіри одного виконавця до інших. Потік діяльностей також є невизначеним. Актори приймають свої суб'єктивні рішення щодо того, як, коли і з ким співпрацювати, або формуючи більш-менш стабільні команди, що базуються на колективних узгодженнях, або вибираючи оптимальні пропозиції під час торгів на компроміс кожний раз, як потрібно знайти нового співпрацівника для коаліції.

Підходи, що забезпечують засоби для моделювання акторів, діяльностей у потоку їх кооперативного виконання, для аналізу і передбачення їх поведінки, для рекомендацій і для проведення корекцій впливу для того щоб якось оптимізувати їх спільні проактивні зусилля і, як наслідок, керувати та координувати розподілені команди егоїстичних акторів, які працюють в умовах

високої невизначеності , є дуже потрібними в різних галузях, таких як: E-Commerce, E-уряд, віртуальні підприємства, дистанційна освіта, та ін.

Важним аспектом є той, що стратегії координації, які базуються на доброзичливості акторів, не суперечать таким, що базуються на егоїзмі акторів. Як відмічено Лессером у [37] "…існує більше спільного між кооперативними і егоїстичними механізмами координації, ніж зараз вважають – особливо якщо простори, в яких такі механізми використовуються…, стають дедалі складними в термінах зростання рівня невизначеності і неповноти інформації…"

Серйозні зусилля вкладаються дослідниками із областей розподіленого штучного інтелекту (DAI) та комп'ютерних технологій (CS) для рішення цієї проблеми. Відомі декілька елеґантних, як практичних, так і теоретичних результатів для різних областей. Гарні приклади можуть бути знайдені в [38], [39], [40], [41], [42], [43]). Наприклад, в E-Commerce моделі для формування коаліції базуються на проведення пре- і пост-торгів, як запропоновано в [44], з використанням COALA [45] як системи відладки загального призначення для вивчення кооперативної поведінки в коаліціях агентів.

Більшість підходів базуються на парадигмі агентів і стверджують що головною в цій галузі є проблема відповідної координації – керування взаємозалежностями між діяльностями [46]. Легко помітити, що розподілена координація також є прикладом спільних, високою мірою невизначених процесів і що об'єкти координації [47] є розподіленими, часто обмеженими, з фіксованими ресурсами, і суб'єктивними намірами, що можуть конфліктувати.

Даний розділ звіту обговорює формальну модель для розподіленої координації формування динамічних коаліцій між раціональними (такими, що мають істотний егоїзм в намаганні отримати максимальну вигоду та такими, що кооперуються (намагаючись досягнути оптимального виконання діяльності у спільній праці один з одним) агентами. Такі агенти є функціональними акторами, які здатні виконувати спеціалізовані діяльності в організації. Коаліції формуються динамічно в процесі виконання завдання. Торги на залучення

виконавців в коаліцію завдання проводять кожний раз коли під час виконання завдання з'являється нова діяльність. Такий процес торгів вважається спеціальним типом координації і виконується під час так званої фази призначення виконавців, яка передує виконанню діяльності.

5.1 Організації, середовища і завдання

Організації діють в межах середовища. Вважаємо, що середовище моделюється функцією яка генерує завдання T як множини діяльностей w^i:

$$E \rightarrow T = \{w^1, w^2, ..., w^k\} \quad (5.1)$$

Такі завдання сприймаються організацією. Організація створена таким чином, щоб виконувати завдання, які забезпечуються середовищем у вигляді зовнішніх впливів. Згідно з принципами організаційного структурування (див, наприклад. [48]) вважаємо, що організація (якщо вона має плоску форму), або структурний рівень організації (якщо вона ієрархічна) буде множиною активних істот (акторів), які мають відповідну спеціалізацію і спілкуються згідно із стандартним зразком спілкування [34], [16]. Актори моделюємо як раціональні [25] програмні агенти, розроблені в рамках [16]. Спеціалізація агента задається через множину макромодельних програм для виконання діяльностей[16].

5.2 Вивчення змін можливостей агентів-партнерів

Агенти мають власні суб'єктивні очікування ***можливостей (capabilities***) своїх колег, в розумінні [49]. Такі очікування змінюються з часом, оскільки актори адаптують свої суб'єктивні уявлення, накопичуючи параметричні відгуки від своїх партнерів під час виконання завдань/діяльностей [16] , і є елементами такої матриці:

$$\mathbf{C} = \begin{array}{c} \\ A_1 \\ \dots \\ A_i \\ \dots \\ A_n \end{array} \begin{array}{c} \begin{array}{ccccc} w^1 & \dots & w^j & \dots & w^m \end{array} \\ \begin{bmatrix} c_1^1 & & c_1^j & & c_1^m \\ & & \dots & & \\ & \dots \; c_i^j = c_{A_i}^{w^j}(X_{w^j}) & \dots & & \\ & & \dots & & \\ c_n^1 & & c_n^j & & c_n^m \end{bmatrix} \end{array} \tag{5.2}$$

де виміри матриці **C,** *n* і *m,* ростуть в процесі еволюції актора, відображаючи появлення нових знань про агентів – партнерів (*n*) і діяльностей, які вони можливо здатні виконати (*m*). Верхня границя виміру *n* є число акторів-учасників організації, враховуючи і власника матриці **C**. Максимальне значення *m* є міцність множини *W* дозволених атомарних робіт для організації

5.3 Модель завдання із делегованими роботами

Вважаємо, що *завдання* $\mathrm{T} = \{w^1, w^2, ..., w^k\}$ це множина із однієї або більше діяльностей. Діяльність w^k, що є атомарною для даного актора, іншими акторами може вважатися як завдання, у відповідності до їх баз знань, а саме до тих частин, які розпізнають зовнішній вплив.

Кожний із акторів, притягнутий до виконання завдання, має власні уявлення про те, як, в якої послідовності виконувати атомарні діяльності і скільки зусиль треба затратити для виконання діяльності, оскільки забезпечено, що актор має певну робочу ємність відповідну до певної атомарної діяльності. Такі уявлення є суб'єктивними Частковими Локальним Планами (ЧЛП, PLP) для виконання певних атомарних діяльностей. ЧЛП формалізовані в "легковажних онтологіях завдання", які можуть бути написані на мові Standard OIL [50] і використовуватися відповідними макромодельними програмами. ЧЛП відрізняються від, скажімо, Глобальних Планів (GPGP) [51] тим, що в ЧЛП не містяться суб'єктивні уявлення про можливі дії акторів-партнерів. У випадку, якщо завдання розглядається у вигляді графа (див., наприклад, [52]) кінцева форма такого графа може змінюватися. Залежно від послідовності застосування акторів. Актори приєднуються до виконання завдання за результатами

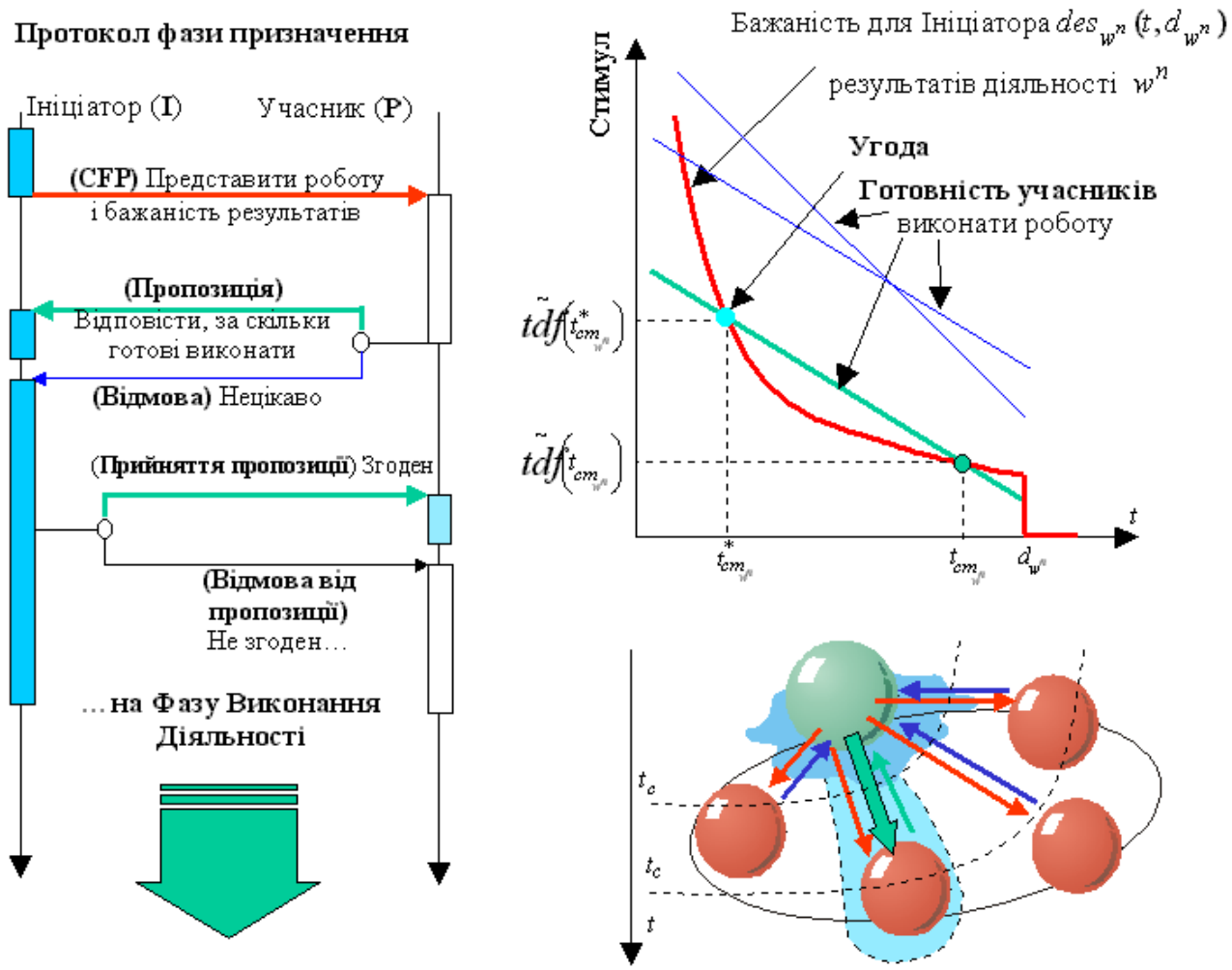


Рисунок 5.1 – Фаза призначення виконавців. Торги на передачу діяльності і приєднання до коаліції.

проведення торгів на призначення виконавців (під)завдання або діяльності. Такі торги (див.Рис.5.1) мають місце під час фази призначення виконавців кожний раз, коли (під)завдання/діяльність передається на виконання. Оскільки акторам дозволено генерувати нові завдання, очевидно, що граф завдання може мати цикли, бо вершина графа (актор) може генерувати підзавдання, що включають або приводять до виконання деякої атомарної діяльності, яка, в свою чергу, може бути назначена тому ж самому актору.

Атомарні діяльності можуть бути неявно пов'язані (як в *TÆMS* [51]) з строгими залежностями: діяльність w^i строго залежить від діяльності w^j у випадку, коли результати діяльності w^j є істотними для початку виконання діяльності w^i. В рамках даного дослідження нестрогі (слабкі) залежності (діяльність w^j полегшує виконання діяльності w^i) не приймаються до уваги і їх

залишимо для наступного дослідження. Таким чином, потік діяльностей обмежений простими залежностями передування, хоча і може містити ітеративні або рекурсивні цикли.

Після того, як актор отримав вплив, він може:

– Прийняти і виконати деякі із атомарних діяльностей, що містилися в (під)завданні;
– Відхилити деякі з атомарних діяльностей;
– Прийняти рішення про передачу множини діяльностей своїм партнерам, згідно із своїми уявленнями щодо можливостей партнерів (2), міри довіри партнерам (19), і готовності виконати діяльності (Секція 3)
– Вимагати створення нових діяльностей, виконання яких (за його уявленнями) буде істотним для виконання всього завдання або підзавдання

Якщо актор не відхилив одну чи більше діяльностей, він стає учасником динамічної коаліції для виконання завдання. Таким чином, коаліція для виконання завдання є відкритою динамічною системою, за визначенням Скотта [53].

Своїм приєднанням до коаліції актор повинен дотримуватися деяких правил, які регулюють співвідношення між доброзичливістю і власними інтересами. Ці правила можуть бути класифіковані за гіпотезою Дженнінгса про “узгодження та традиції” [52] як індивідуальні і суспільні узгодження актора та коаліційні конвенції:

- *Правило 1:* **Узгодження відносної кооперації**. Учасники коаліції відносно погоджуються на спільне досягнення загальної цілі: виконати завдання з максимальною можливою ефективністю (максимальною якістю, збалансованістю напруження, мінімальним часом…). Виконання цього узгодження залежить від розбіжностей між внутрішніми намірами автономного актора та загальною ціллю коаліції, яку формують для виконання завдання.

- *Правило 2:* **Узгождення при назначенні діяльності**. Під час фази призначення виконавців учасник коаліції, що пропонує виконати діяльність

(Ініціатор) намагається чесно представити функцію ***бажаності*** щодо цієї діяльності. У відповідь, можливі виконавці (Учасники) погоджуються чесно відповісти про свою готовність виконати запропоновану діяльність, віддаючи інформацію про ***ємність*** яку вони згодні віддати, у вигляді параметричного відгуку [16].

- *Правило 3:* **Узгодження про доставку результатів.** В разі, коли актор був обраний для виконання деякої діяльності, він намагається безумовно виконати цю діяльність і повернути результати одразу ж після завершення виконання роботи.

5.4 Параметри роботи: витрачувана ємність, час закінчення і бажаність результатів

Вважається, що актор A характеризується своєю ємністю (***capacity***) $N_A(w^j)$ відносно до конкретної діяльності w^j у дискретному простору часу. Під ємністю розуміємо здатність актора виконати діяльність за інтервал часу τ. Якщо, наприклад, A друкує сторінки А4, то $N_A(w^j = ("print\_1\_A4\_page") = 4$ у випадку, якщо A має 1 принтер, що друкує 4 сторінки за період τ і може бути подвоєне, якщо A матиме ще один такий же принтер. Ємність актора може вважатися **необмеженою** — якщо в будь-який момент часу актор здатний надрукувати стільки сторінок А4, скільки необхідно. У випадку, коли термін виконання w^j буде залишатися $1 \times \tau$, оскільки актор оперувати тільки з одним типом принтерів, мінімальний термін виконання діяльності не може бути менше за один інтервал часу. Такий випадок є ідеальним, коли немає інших діяльностей, що чекають на виконання. У випадку, коли ємність актора вважається **обмеженою** — якщо в розпорядженні актора є тільки фіксована кількість принтерів — $N_A(w^j)$ може бути рівномірно, або нерівномірно, а, скажімо, відповідно до пріоритетів, які мають різні покупці, розподілена між діяльностями, що знаходяться у виконанні даного актора.

Діяльність w^j може бути обмежена часом закінчення (***deadline***) d_{w^j}. Час закінчення – це така точка в часі, після якої результати виконання w^j більше не потрібні агенту покупця. Наприклад, спікерові парламенту непотрібні роздрукування програми сесії на A4 із якістю 600-dpi після того, як сесія пройшла. Це буде означати, що значення функції бажаності результатів діяльності w^j (див. наприклад, *TÆMS* - властивість якості [51] як непряму аналогію):

$$des_{w^j}(t, d_{w^j}) = \begin{cases} tdf(t), t \le d_{w^j} \\ 0, t > d_{w^j} \end{cases} \tag{5.3}$$

приймає нульове значення після часу закінчення, і приймає ненульові значення стимулу $tdf(t)$ як різновиду компромісу залежно від часу.

5.5 Фаза призначення виконавців: координація розміщення діяльностей і формування коаліції завдання

Вважається, що перед тим, як діяльність (або (під)завдання) передається актору і актор виконує її, проводиться фаза призначення виконавців. Фаза призначення організована в рамках спрощеного FIPA Contract Net Protocol [54] – див.мал.5.1. Під час призначення агент – ініціатор (**I**) шукає оптимальну ціну за виконання діяльності, розповсюджуючи функцію бажаності результатів (5.3), і можливі виконавці (учасники, **P**) відзначають її готовність потратити частину своїх можливостей для виконання діяльності у параметричній формі. Учасники відображають свою готовність приєднатися до коаліції завдання, повертаючи функцію, яка співвідноситься з бажаністю результатів ініціатора за період часу $]0, d]$. Після аналізу відгуків і визначення оптимального відгуку (Рис. 5.1.), ініціатор передає діяльність обраному контрактору. Учасники цього процесу торгів можуть як бути, так і не бути членами коаліції завдання. Якщо новачок був обраний для виконання діяльності, саме новачок стає членом коаліції.

Час, необхідний для проведення фази призначення виконавців вважається незрівнянно меншим в порівнянні з інтервалом часу τ.

5.5.1 Організація фази призначення виконавців агентом - ініціатором

В рамках запропонованого підходу вважається[17], [34], що ініціаторові можна передати діяльність іншому актору – партнеру у випадках, якщо він не здатен виконати цю діяльність сам через перенапругу, або через нестачу потрібних вмінь (особливість макромодельної програми). Задача ініціатора під час фази призначення виконавців двояка (див.Рис. 5.1):

- Ініціалізувати процес торгів через надання учасникам (що мають необхідні можливості для виконання діяльності – згідно з формулою 5.2) опису діяльності і відповідної функції бажаності результатів.
- Зібрати і проаналізувати параметричні відгуки від учасників для прийняття рішення про вибір одного з них для виконання діяльності

Вичерпну дискусію щодо поведінки ініціатора під час вирішення цієї проблеми можна отримати в [55]. Для стислості ми тільки позначимо використаний підхід.

Опис діяльності задається в формі, визначеної “легковажної онтології”, написаної на мові Standard OIL [50]. Значення компромісів в функції бажаності результатів (див. формулу 5.3) для діяльності w^j, істотні для виконання (під)завдання T, походять від:

- Бюджету завдання, яке виконує ініціатор — запропоноване значення стимулу за виконання завдання T
- ЧЛП (під)завдання T, яке забезпечує інформацію про можливі зусилля для виконання складових частин T і інформація про передування діяльностей в T

Рішення про вибір контрактора приймається при рішення задачі пошуку оптимальної заявки $\tilde{tdf}(t,d)$ серед усіх запропонованих відгуків учасників.

5.5.2 Міркування агента-учасника щодо запропонованого стимулу

Нехай в t_c агент - кандидат **P**, який приймає участь в торгах на розміщення діяльності отримує пропозицію виконати нову діяльність w^n від ініціатора **I**. Пропозиція супроводжується функцією бажаності результатів (5.3), яка показує залежність запропонованого стимулу від часу (див. Рис. 1.) Міркування учасника **P** під час проведення фази призначення виконавців полягають у оптимальному плануванні виконання запропонованої діяльності з урахуванням вже існуючих в черзі **P** діяльностей. Раціональна мета полягає в отриманні максимальної вигоди.

Потенційний контрактор **P** в момент часу t_c має інформацію про розподілення зусиль $R_{w^i}(t)$, $R_o(t)$ з урахуванням поточної множини діяльностей W_c в межах поточного періоду діяльності T_c.

В t_c **P** вже виконує або погодився і планує виконати множину діяльностей $W_c = \{w^1,...,w^i,...,w^k\}$. Діяльності w^i виконуються агентом **P** за проміжок часу $[t, t+\tau]$, кожна діяльність потребує $r_{w^i,t} \geq 0$ його ємностей за стандартний інтервал часу. Якщо $R_{w^i}(t)$ є дискретною функцією розподілення споживання можливостей агента з урахуванням w^i, її можна визначити як

$$R_{w^i}(t) = \left\{r_{w^i,t^i}, r_{w^i,t^i+\tau},...,r_{w^i,t^i+l^i\tau}\right\}, \tag{5.4}$$

Якщо початок виконання діяльності w^i є в момент t^i і завершення - в момент $t^i + l^i\tau$. Поточний період діяльності агента **P** $T_c = \left[t^s = \min_i t^i, t^e = \max_i (t^i + l^i\tau)\right]$ асоційований з W_c. Загальне розподілення споживання ємності агента за період T_c може бути обчислене як:

$$R_o(t) = \left\{\sum_i r_{w^i,t^s},...,\sum_i r_{w^i,t^s}\right\}. \tag{5.5}$$

Взагалі, загальне споживання ємності агента може бути обмежено функцією порогу $R_m(t)$:

$$R_o(t) \leq R_m(t). \tag{5.6}$$

Невикористана ємність раціонально резервується на критичний випадок. Одним із таких випадків може бути необхідність закінчити виконання

діяльності з використанням несподівано більшої ємності за узгоджений проміжок часу і за узгоджений стимул.

Рис. 5.2 показує значення $R_m(t)$, $R_o(t)$, $R_{w^i}(t)$ для множини поточних діяльностей $W_c = \{w^1, w^2, w^3, w^4\}$ за поточний проміжок діяльності $T_c = [t_c - 2\tau, t_c + 2\tau]$.

Мета **P** полягає в раціональній оцінці запропонованої $tdf(t)$ і у відповіді з власною пропозицією $\tilde{tdf}(t)$.

Схема міркувань, яку використовує **P,** така:

– Обчислити функцію розподілення ємності $R_{w^n}(t)$ для w^n і для часу t_{cm} необхідного для виконання w^n, приймаючи до уваги функцію розподілення ємності $R_o(t)$ для робіт із W_c і визначити значення стимулу $tdf(t_{cm})$ як першу точку параметричного відгуку $\tilde{tdf}(t)$ - зелена точка на Рис. 5.1.

– Спробувати максимізувати стимул $\Pi = \sum_{w^i \in W_c \cup \{w^n\}} des_{w^i}(t_{cm_{w^i}}, d_{w_i})$ шляхом перерозподілення ємностей, і, можливо, перевпорядкування діяльностей.

Рішення цієї оптимізаційної задачи забезпечить другу точку параметричного відгуку $\tilde{tdf}(t)$ - світло-блакитна точка на Рис. 5.1.

5.5.3 Обчислення агентом-учасником розподілу долей витраченої потужності

Перша річ, яку необхідно зробити **P**, це оцінити зусилля S_{w^n}. які **P** затратить на виконання w^n відповідно з описом діяльності у формі онтології завдання. Для нашого приклада з друкуванням, така оцінка буде дуже простою. **P** розділятиме загальну кількість сторінок що треба надрукувати на комбіновану фізичну ємність застосованих принтерів (сторінок за τ). Якщо ми розглянемо графічне подання планування виконання w^n (див. Рис. 5.3), **P** необхідно

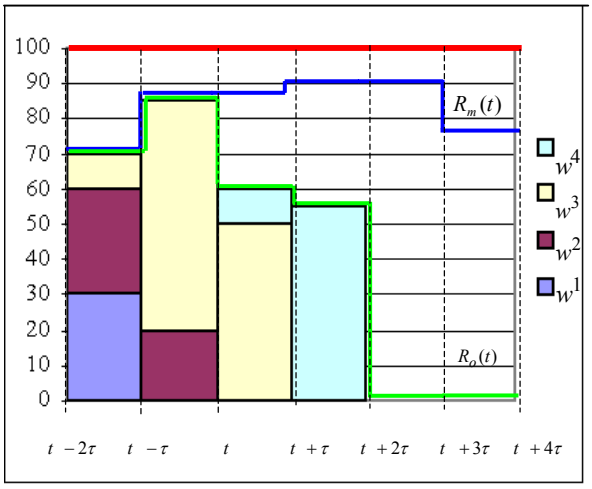


Рисунок 5.2 – Розподілення ємностей агента для діяльностей : - синя лінія, - зелена лінія

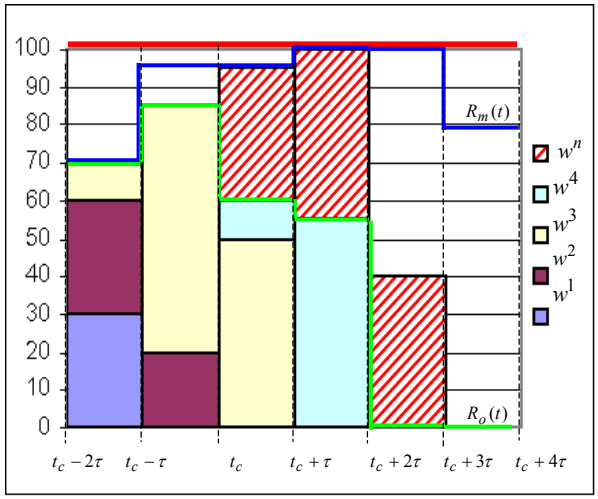


Рисунок 5.3 – Обчислення розподілення ємностей для нової запропонованої діяльності w^n з очікуваним зусиллям $S_{w^n}=120\times\tau$.

поступово замалювати невикористану площу між $R_m(t)$ і $R_o(t)$ починаючи від $t=t_c$ до того, як результуюча замальована площа не буде дорівнювати S_{w^n}. Рис. 5.3 демонструє результат для нової діяльності w^n з очікуваними зусиллями виконавця $S_{w^n}=120\times\tau$.

Наступний шаг міркувань **P** полягає в обчисленні часу t_{cm} необхідного для завершення w^n приймаючи до уваги розподілення ємності для робіт W_c. t_{cm} обчислюється як мінімальне значення, що задовольняє таким обмеженням:

$$\sum_{t^n=t_c}^{t_{cm}}\tau(R_m(t)-R_o(t))\geq S_{w^n} \tag{5.7}$$

Час виконання l^n діяльності w^n тоді буде $\frac{t_{cm}-t^n+1}{\tau}$. Функція розподілення споживання ємності $R_{w^n}(t)$ at $\{t^n, t^n+\tau,..., t^n+l^n\tau\}$ тепер може бути легко визначена як

$$R_{w^n}(t)=\begin{cases}R_m(t)-R_o(t), t_c\leq t<t_{cm}\\ S_{w^n}-\sum_{t=t_c}^{t_{cm}-\tau}\tau(R_m(t)-R_o(t)), t=t_{cm}\end{cases}. \tag{5.8}$$

P не має уяви про w^n в момент часу $t < t_c$, і буде нераціонально відкласти виконання w^n на час $t > t_{cm}$, w^n буде споживати всю ємність **P**, якщо $t_c \le t < t_{cm}$ до порогу $R_m(t)$, і деяка ємність може бути невикористана в останній проміжок часу $t = t_{cm}$.

Після того як t_{cm} визначена, **P** може так само оцінити свій майбутній прибуток через визначення стимулу $\tilde{tdf}(t_{cm}) = tdf_{w^n}(t_{cm})$ запропонованого **I** у випадку w^n завершується в момент часу t_{cm}. S

```
TCM1: t = t_max ;
TCM2: If R_{w^i}(t) ≠ 0
            then t_{cm_{w^i}} = t ; Exit TCM;
TCM3: t = t − τ ; Goto TCM2;
End
```

Після отримання рішення (12) разом із часами завершення $t^*_{cm_{w^i}}$, значення стимулу $tdf_{w^n}(t^*_{cm_{w^n}})$, які запропонував ініціатор **I**, використовується **P** для формування другої точки свого відгуку $\tilde{tdf}(t)$ - див Рис. 1. Параметричний відгук **P** після всього повертається ініціатору **I** у формі

$$\tilde{tdf}(t,d) = \left\{ \begin{pmatrix} des_{w^n}(t_{cm_{w^n}}, d_{w^n}), t_{cm_{w^n}} \\ des_{w^n}(t^*_{cm_{w^n}}, d_{w^n}), t^*_{cm_{w^n}} \end{pmatrix} \right\}. \tag{5.9}$$

5.6 Оцінка ініціатором міри довіри до агентів-учасників переговорів

Раціональний актор, при появі дуже привабливої діяльності (або(під)завдання) пропонує, або відповідно до особливостей своєї поведінки, може занизити попередньо задекларованої ємності, яку він тратить на виконання поточної множини діяльностей. Це може привести до зростання часу виконання цих діяльностей і може серйозно знизити бажаність результатів

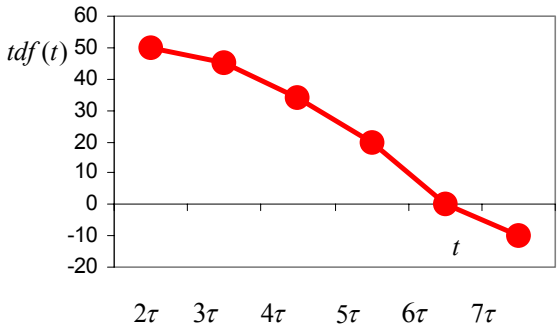


Рисунок 5.4 – Приклад функції $tdf_I(t)$

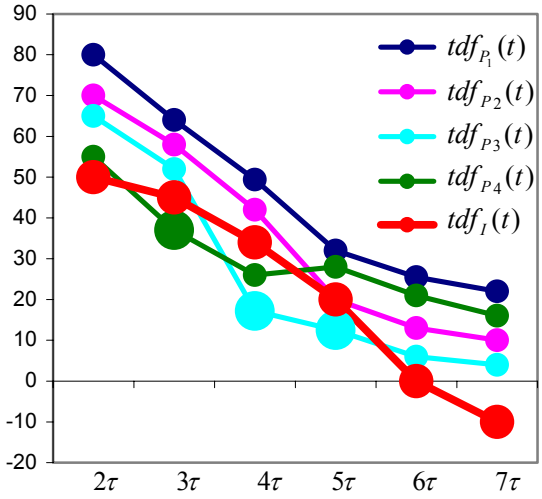


Рисунок 5.5 – Графіки функцій $tdf(t)$ агентів P і агента I.

агентів покупця, і, як наслідок, знизити міру довіри для актора, який не виконує своїх зобов'язань.

Механізм обчислення ***міри довіри*** (***credibility***) до акторів очевидно буде таким же як той, що використаний для обчислення змінних можливостей партнерів (2). Оцінки міри довіри змінюються з часом, оскільки актор міняє свої суб'єктивні уявлення шляхом порівняння значень бажаності (див. Рис. 5.5), які отримані по-перше – від часу виконання роботи, заявленої виконавцем на фазі призначення виконавців, і по-друге – від актуального часу, який використав виконавець для отримання результатів роботи. Елементи матриці міри довіри обчислюються так:

$$Cr_{i,j} := Cr_{i,j} \times \begin{cases} 1, t_r \le t_a \\ p_{w^j}(t_a / t_r), t_a < t_r \le d_{w^j} \\ 0, t_r > d \end{cases}, \qquad (5.10)$$

де t_a це час, за який сторони домовилися виконати роботу w^j, t_r це реальний час отримання результатів w^j, d це час завершення і p_{w^j} це ваговий коефіцієнт, що характеризує поточний пріоритет діяльності w^j для агента покупця.

5.7 Модель прийняття рішення агентом-ініціатором процесу переговорів

Агент – ініціатор **I** хоче знайти виконавця для деякої діяльності w_1. Для цього агент **I** повинен згенерувати функцію $tdf_I(t)$, яка відображає залежність максимального розміру винагороди (стимулу), яке агент **I** згодний заплатити за виконання діяльності w_1, від часу виконання діяльності. Далі, відповідно з протоколом переговорів, агент **I** відправляє пропозицію виконати діяльність w_1 потенційним агентам-виконавцям **P**.

Приклад функції $tdf_I(t)$ зображений на Рис. 5.4. Прокоментуємо цей малюнок. Якщо виконавець (агент **P**) виконає діяльність w_1 за час 2τ, то агент **I** готовий заплатити йому не більше ніж 50 одиниць винагороди, якщо діяльність w_1 буде виконана за час 3τ - то не більше ніж 45 одиниць винагороди, і так далі, якщо діяльність w_1 буде виконана за час 7τ, то агент **P** повинен бути заплатити штраф у розмірі 10 одиниць (це відмічено від'ємним значенням функції). Кожний із агентів **P**, після отримання пропозиції виконати діяльність, повинен або з генерувати функцію - відгук $tdf_P(t)$, або відмовитися від виконання роботи. На Рис. 5.5 дано приклад функцій $tdf_P(t)$ для чотирьох агентів.

Отримавши від кожного із агентів **P** функцію $tdf(t)$, агент **I** повинен визначити, якому із агентів **P**, і на яких умовах (час виконання і винагородження) доручити виконання роботи w_1. Іншими словами, необхідно знайти оптимальну точку $(t, tdf(t))$ із таблиці функцій $tdf_{P_i}(t)$. Вочевидь, жодна із точок, які знаходяться над графіком функції $tdf_I(t)$ (красна лінія на мал 5.4, 5.5) не є дозволеною, оскільки агент **I** не згоден платити винагороду, більшу за ту, яку він вказав в таблиці функції $tdf_I(t)$. На Рис. 5.5 такими точками є всі точки функції $tdf_{P_1}(t)$, і деякі точки функцій $tdf_{P_2}(t)$, $tdf_{P_3}(t)$, $tdf_{P_4}(t)$.

Множину X точок $(t, tdf(t))$ із таблиці функцій $tdf_{P_i}(t)$, які задовольняють умові

$$(t, tdf) \in X \text{ якщо } ((tdf = \min(tdf_{P_1}(t), tdf_{P_2}(t), ..., tdf_{P_n}(t))) \wedge (tdf \le tdf_I(t))) \ (\forall t \in \mathrm{T}), \qquad (5.11)$$

Будемо називати множиною допустимих точок.

Таким чином, усі допустимі точки лежать не вище графіка функції $tdf_I(t)$. Допустимі точки показані на мал.5.5 цифрами 1, 2, 3 і помічені напівжирним маркером. Дійсно, в точках 1, 2, 3 на мал.5.5 агенти **P** готові виконати роботу w_1 за винагородження, яке не перевищує те винагородження, яке агент **I** вказав у власній таблиці функції $tdf_I(t)$.

Множину X допустимих точок зручно записувати у вигляді наступної таблиці.

Номер точки	Точка (t,tdf)	Агент, якому належить ця точка
1	$(3\tau, 37)$	P_4
2	$(4\tau, 17.2)$	P_3
3	$(5\tau, 12.5)$	P_3

Якщо множина допустимих точок виявиться пустим, то агент **I** повинен або переглянути свою функцію $tdf_I(t)$ і запропонувати нові умови виконання роботи, або відкласти виконання цієї роботи на пізніше, коли агенти **P** будуть менше завантажені.

Із множини X точок $(t,tdf(t))$ необхідно знайти оптимальну, вирішуючи двокритеріальну задачу:

$$F(t^*(t,w_i),tdf^*(t,w_i,tdf(t))) \to \max, \qquad (5.12)$$

де $t^*(t,w_i)$ - функція, що описує критерій оцінки часу виконання діяльності w_i, тобто якщо час t виконання діяльності w_i близький до оптимального, тим більшим буде значення функції $t^*(t,w_i)$. При $t = t_{опт}$ функція повинна досягати максимуму.

$tdf^*(t,w_i,tdf(t))$ - функція, що описує критерій оцінки розміру винагороди агента, який виконує роботу w_i за час.

Цю задачу будемо рішати за допомогою метода лінійної згортки. Припустимо, що для агента **I** критерії винагороди і часу є рівнозначними, тому

задачу будемо рішати з параметрами 1, 1. Таким чином, ми отримуємо таку цільову функцію від двох змінних t і tdf:

$$F(t,tdf) = (t^*(t) + tdf^*(tdf)) \to \max, \tag{5.13}$$

Треба знайти максимум цієї функції на деякій множині X точок (t,tdf). Формально рішення цієї задачі можна записати як:

$$\max_X (t^*(t) + tdf^*(tdf)), \tag{5.14}$$

З урахуванням того, що потужність множини дозволених точок X невеличка, отриману задачу вирішуємо методом повного перебору.

5.8 Підсумки

В даному розділі звіту обговорюється формальна модель для розподіленої координації динамічного формування коаліції між Презентована формальна модель для розподіленої координації формування динамічної коаліції між раціональними (такими, що мають істотний егоїзм в намаганні отримати максимальну вигоду та такими, що кооперуються (намагаючись досягнути оптимального виконання діяльності у спільній праці один з одним) агентами. Такі агенти є функціональними акторами, які здатні виконувати спеціалізовані діяльності в організації. Співвідношення між доброзичливістю і раціоналізмом учасника коаліції регулюється узгодженням відносної кооперації, узгодженням при зазначенні діяльності та узгодженням доставки результатів

Переговори на залучення учасника до коаліції завдання проводяться кожний раз як в процесі виконання завдання з’являється нова діяльність. Процес переговорів розглядається як спеціальний тип координації і виконується під час так званої фази призначення виконавців, яка передує виконанню діяльності. Під час фази призначення, яка проводиться агентом – ініціатором з використанням спрощеного протоколу FIPA Contract Net Protocol учасників запрошують оцінити свої здатності по виконанню даної діяльності і відповісти параметрично (18) в термінах стимулу за проміжки часу. Оскільки ємність

учасників є обмеженою, частково використаною для виконання інших діяльностей і обмеженою деяким значенням резерву “на випадок”, вони проводять доволі складне міркування для того, щоб відповісти ініціатору. Це міркування проводиться за такою схемою: (перше) обчислити розподілення ємності і оцінити час і стимулу необхідного для виконання нової діяльності; (друге) Спробувати оптимізувати загальний стимул через пере розподілення ємності для множини своїх діяльностей, як наслідок, вирішити дискретну задачу нелінійного програмування (12) з обмеженнями (13-17). Оскільки вважаємо, що кардинальне число множини діяльностей, які виконує агент, не дуже велике за поточний період діяльності, проблема вирішується повним перебором ітеративне.

Учасники, намагаючись відкласти частину своїх менш цікавих діяльностей для збільшення стимулу, можуть порушити попередні угоди щодо часів закінчення цих діяльностей, таким чином не виконати своїх зобов’язань. Механізм (19) для оцінки значень міри довіри партнерів було запропоновано для забезпечення можливості агенту відчувати себе комфортно в довірчому просторі.

Моделі і підхід, запропоновані у даному розділі звіту, є суб’єктом для подальшого розвитку. А саме, рішення для наступних аспектів можуть спростити подальше підсилення середовища для моделювання. По-перше, під час пере розподілення ємностей учасники торгів можуть також оцінити як зміни часів закінчення робіт будуть впливати на міри довіри до них з боку інших акторів. По-друге, модель завдання повинна також бути покращена для керування діяльностями, результати виконання яких можуть знизити зусилля, необхідні для виконання поточної і запланованих робіт. Актор-ініціатор може також оптимізувати свою вигоду для множини попередньо переданих діяльностей після обрання контактора для поточної діяльності, якщо всі інші будуть згодні знизити очікувану вигоду на таку, яку пропонує ініціатор.

6 Забезпечення семантичної інтероперабельності

Ще одним із найважливіших завдань даній роботи є розробка підходів до вирішення проблем семантичної інтероперабельності у визначених суспільствах агентів, що виконують завдання. Цій розділ дає систематизований огляд стану проблеми і прецедентів її вирішення, які у майбутньому мають бути застосовані в ЄІП. В розділі також презентуються отримані рішення для забезпечення семантичної інтероперабельності в суспільствах і коаліціях агентів, що виконують завдання у ЄІП.

6.1 Проблема семантичної інтероперабельності

Задача спільного використання неоднорідних баз даних, відома сьогодні також як окремий випадок задачі досягнення інтероперабельності інформаційних ресурсів, виникла на початку 80-х р.р. XX сторіччя.

На той час у розвинутих інформаційних системах (ІС) уже була потреба спільного використання декількох (можливо, розподілених) баз даних, керованих різними СКБД, що супроводжувалося значними зусиллями з боку розроблювачів цих інформаційних систем. Прикладами рішення задачі про спільне використання мережних, ієрархічних і реляційних баз даних, що відносяться до початку 80-х р.р., можуть бути проекти СИЗИФ [56-58], POLYPHEME[59], проект Каліфорнійського університету [60] та інші.

На сьогоднішній день ця задача набула ще більш важливого значення, у зв’язку зі зростанням кількості різноманітних за змістом, структурою, обсягом інформаційних ресурсів (баз даних, баз знань, програмних компонентів і т.д.), створених на різних програмно-апаратних платформах. Інформація, представлена цими ресурсами, багаторазово дублюється, її спільне використання утруднене в силу різних специфікацій інформаційних ресурсів, прийнятих різними розроблювачами. Така ситуація послужила причиною розвитку досліджень в області спільного і повторного використання

компонентів інформаційних ресурсів. Важливість даної проблематики була підкреслена в [61] і виділена як один із трьох базових напрямків у дослідженнях на поточне десятиріччя.

Наведемо основні використовувані надалі визначення.

Інтероперабельність означає можливість створення систем з довільних неоднорідних, розподілених компонентів, на основі уніфікованих інтерфейсів [62].

Інтероперабельна інформаційна система (ІС) складається з компонентів, що представляють довільні інформаційні ресурси (програмні компоненти, бази даних, бази знань, файли даних і т.д.), які розглядаються незалежно від апаратно-програмної платформи і розміщення в просторі. Компоненти взаємодіють, обмінюючись заявками.

Задача досягнення інтероперабельності різнорідних інформаційних ресурсів може бути представлена як дві підзадачі:

– досягнення технічної інтероперабельності, тобто забезпечення спільної роботи різнорідних апаратно-програмних платформ. У даному огляді питання технічної інтероперабельності розглядатися не будуть, з огляду на той факт, що саме цій проблемі дотепер була присвячена велика частина зусиль консорціуму Object Management Group, OMG (потрібну інформацію можна почерпнути в [63, 64, 65]);
– досягнення семантичної інтероперабельності, тобто забезпечення спільного використання різнорідних інформаційних ресурсів.

У даному розділі пропонується огляд існуючих систем інтеграції неоднорідних баз даних і баз знань з погляду семантичної інтероперабельності і використані в цих системах підходи.

6.1.1 Семантична інтероперабельність

При створенні ІС на неоднорідних інформаційних ресурсах, для досягнення семантичної інтероперабельності необхідно вирішувати проблеми порівняння вмісту цих ресурсів, знаходження відповідностей і вирішення конфліктів між ними, а також проблему сполучення різнорідних ресурсів. Отже, задача побудови семантично інтероперабельної ІС розбивається на 3 частині:

- вироблення специфікацій, в достатній мірі уніфікованих і повних с точки зору конкретної задачі, для всіх інформаційних ресурсів;
- вироблення засобів порівняння можливостей доступних інформаційних ресурсів з потребами прикладних задач, розв'язуваних даною ІС;
- створення несуперечливої й адекватної моделі предметної області задачі шляхом композиції моделей предметних областей, що відповідають конкретним інформаційним ресурсам.

При цьому повна специфікація інформаційного ресурсу буде включати:

- специфікацію його структури і функцій (статичні характеристики);
- специфікацію обмежень цілісності;
- специфікацію поведінки інформаційного ресурсу (динамічні характеристики);
- специфікацію контексту, тобто області, у якій передбачається використання ресурсу.

Сучасні системи інтеграції неоднорідних інформаційних ресурсів використовують концепцію **медіатора** (див. наприклад в [66]), тобто посередника між розподіленими інформаційними ресурсами у межах інтероперабельної системи та користувачами цих ресурсів.

Визначення структури і поведінки конкретного інформаційного ресурсу завжди визначається семантикою предметної області, що її представляє цій ресурс. Специфікація цієї змістовної оболонки даних може бути виконана у вигляді інформаційної схеми (як, наприклад, у структурованих моделях даних

типу реляційної). Однак на більш загальному рівні, будь-яка специфікація семантики даних може бути записана на деякій формалізованій мові, наприклад, на численні предикатів 1-го порядку, або на багатосортній логіці. Кожен предикат, що є присутнім у цій специфікації, відповідає деякому семантичному правилу, що визначає несуперечливий стан бази даних.

Як інструмент створення таких узагальнених специфікацій була запропонована ідея використання **онтологічних специфікацій**.

У [62] наводиться визначення **онтологічної специфікації** інформаційного компонента як набору визначень і понять, а також правил (аксіом), зв'язаних з визначеннями і поняттями з предметної області (прикладного контексту).

Термін "онтологія" у даний час використовується в двох контекстах:

У філософському контексті, **онтологія** – система категорій, використовувана для розгляду з урахуванням конкретного бачення світу [67].

У контексті інформаційних систем, **онтологія** – формалізований опис загальноприйнятого розуміння деякої предметної області, за допомогою якого можуть спілкуватися люди, комп'ютерні системи [68]. Програмні компоненти, для взаємодії між собою в рамках інтегрованої системи неоднорідних ресурсів, використовують **онтології**. Передбачається, що онтологія не залежить від мови представлення предметної області.

Онтологія, на відміну від бази знань, не містить ані знань про методи рішення задач, що стосуються предметної області, ані знань, що дозволяють видавати відповіді на прямі запити про предметну область[69].

На відміну від метаданих, таких як тип, розмір атрибута, онтології повинні мати набагато більш багаті засоби вираження семантики даних. Онтологія може бути традиційною ієрархією понять і типів об'єктів, разом з точним описом кожного типу, однак може містити й аксіоми, що задають обмеження на можливі інтерпретації цих понять[69].

За сучасними уявленнями, розрізняють чотири типи онтологій: **онтології предметних областей**, **онтології задач**, **онтології методів рішення** класів задач, і **формальна онтологія** [67].

Онтологія специфічна для домену, або онтологія предметної області (domain-specific ontology) – опис предметної області, що не залежить від її використання.

Онтологія задач (task ontology) – опис термінів предметної області, прийнятої в конкретному класі задач.

Онтологія методів рішення класу задач (problem-solving methods ontology)– опис термінів і правил, у яких задані методи рішення класу задач. При цьому методи рішення задач – незалежні від предметної області специфікації алгоритму рішенні проблеми (задачі), які можна використовувати для різних предметних областей і (можливо) різних класів задач.

Формальна онтологія (formal ontology або top-level ontology) – опис абстрактних понять, таких як “простір”, “час”, об’єкт”, “подія” тощо.

6.1.2 Дослідження в області онтологій

Перші роботи з дослідження онтологій з’явилися на перетину таких галузей науки як штучний інтелект, філософія, логіка і теорія баз даних.

Одним з найперших проектів, у якому використовувалося поняття онтології, є проект CYC [70]. Метою проекту CYC було створення величезної (порядку 100 000 000 аксіом) бази знань про навколишній світ, яку можна було б використовувати в системах штучного інтелекту для того, щоб перебороти обмеженість сприйняття такими системами навколишнього світу в силу відсутності в їхніх базах знань набору загальноприйнятих понять, так званого загальноприйнятого змісту (common sense).

У рамках проекту CYC була розроблена мова представлення знань CycL. База знань була розділена на два рівні: епістемологічний (для визначення

понять, їх зв'язків, аксіом, що задають обмеження) і евристичний (представлений набором засобів логічного висновку, таких як "генератор аргументу", "порівняння аргументів", "знаходження протиріч", "відновлення логічних висновків", і функціонального інтерфейсу для спілкування з евристичним рівнем бази знань CYC). База знань використовує онтологію, організовану за принципом колекцій категорій (точніше, натуральних сортів Куайна), упорядкованих за допомогою абстракцій узагальнення / спеціалізації.

Колектив розроблювачів зі Стенфордського університету з кінця 1980 р. займається розробкою і стандартизацією мов представлення знань [34], інструментальними засобами створення і модифікації онтологій [69]. У середовищі Ontolingua [69] можна створювати онтології доменів, класів задач, методів рішення задач.

Особливе місце в дослідженні онтологій займає розробка формальної онтології, "…що розглядається як теорія апріорних форм і суті об'єктів…"[71]. Метою цих досліджень є розробка системи логічних примітивів (предикатів), представлених на деякій формалізованій мові, структурованих на підставі множини розглянутих раніше онтологічних угод про довільну предметну область. Особливу цінність здобуває формалізація визначень тієї чи іншої категорії, для того, щоб при побудові онтології використовувати строгі принципи поділу типів, а не інтуїтивні, евристичні правила [71].

Подальшою задачею є об'єднання цих систем примітивів, і приведення їх до загальної формальної онтології.

6.1.3 Системи інтеграції неоднорідних інформаційних ресурсів

Відповідно до класифікації, запропонованої в [66], існуючі у світі інформаційні системи можна розділити на три покоління, за рівнем інтероперабельності.

1-е покоління: інформаційні системи, засновані на структурованих базах даних. Домінуючий підхід при розподілі даних – використання федеративних систем баз даних. Основа для технічної інтероперабельності – локальні мережі.

2-е покоління: інформаційні системи ґрунтуються на структурованих базах даних, частково структурованих даних (текст, гіпертекст у форматі SGML), на форматах даних, специфічних для конкретної предметної області (наприклад, графіка, відео і т.д.). Дані можуть зберігатися на десятках локальних мереж, об'єднаних між собою. Вважається досягнутою технічна інтероперабельність, основна увага приділяється узгодженню мов звертання до даних і узгодження структур інформаційних компонентів.

3-е покоління: інформаційні системи, засновані на усіх відомих способах комп'ютеризованого збереження даних, особлива увага приділяється підтримці відео / просторових / часових / наукових даних. Дані можуть зберігатися як у глобальних корпоративних мережах, так і в Інтернет. Вважається досягнутою технічна, синтаксична, структурна інтероперабельність, і особлива увага приділяється узгодженню семантики використовуваних компонентів.

Вочевидь, зараз найбільш розвиненими є інформаційні системи 1-го покоління інтероперабельності, але для нас більш цікавими будуть системи 2-го і 3-го поколінь.

Існує декілька проектів систем 2-го покоління, що використовують погляд на світ з боку семантики, закладеної в метадані, і використовують онтології. До таких проектів відносяться SIMS [72], HERMES [73], InfoSleuth [74], TSIMMIS [75], Information Manifold [76], OBSERVER[77].

Ці проекти надають доступ до гетерогенних і розподілених інформаційних ресурсів.

Розглянемо ті з них, що надають можливість спільного використання неоднорідних баз даних.

SIMS (http://www.isi.edu/sims)

Модель предметної області (application domain) створюється із використанням системи представлення знань для того, щоб забезпечити фіксований словник описів об'єктів предметної області, атрибутів і відносин між об'єктами. Кожному інформаційному ресурсу ставиться у відповідність модель, у якій описана використовувана в цьому ресурсі модель даних, мова запитів, розташування в мережі, приблизні розміри, і т.д., а також зміст полів цього ресурсу в термінах моделі предметної області. Запити в SIMS формулюються загальною мовою високого рівня також в термінах моделі предметної області. SIMS визначає потрібні інформаційні ресурси за допомогою знань, закладених у модель предметної області й у моделях інформаційних ресурсів системи. Потрібні інформаційні ресурси визначаються під час виконання запиту.

HERMES (http://www.cs.umd.edu/projects/hermes)

Зовнішні інформаційні ресурси представлені у вигляді доменів, що виконують визначені функції з визначеними вхідними і вихідними типами даних. Опис зовнішніх ресурсів виконується за допомогою гібридних баз знань [78]. Звертання до цих доменів виконується за допомогою декларативної мови, заснованої на формальній логіці. На технічному рівні інтероперабельності використовується архітектура медіаторів.

INFOSLEUTH (http://www.mcc.com:80/projects/infosleuth)

Інформаційні ресурси доступні на рівні семантичних концепцій, що є досягненням автономії даних. Інформаційні запити формулюються природно, незалежно від структури, розміщення і навіть існування потрібних інформаційних ресурсів. INFOSLEUTH фільтрує ці запити, сформульовані на семантичному рівні, і знаходить потрібні значення в доступних на час запиту інформаційних ресурсах.

У проекті використовується загальна онтологія домену, і локальні відображення схем баз даних у цю загальну онтологію. Система виконує

попередню обробку і перетворює дані з окремої бази даних у записи, атрибути яких є концепціями загальної онтології домену.

TSIMMIS (http://www-db.stanford.edu/tsimmis)

Підтримується модель даних і загальна мова запитів для об'єднання інформації зі структурованих і частково структурованих джерел даних. Особлива увага приділяється автоматичному створенню трансляторів і медіаторів для доступу до різнорідних ресурсів. Результуюча інформація представляється в моделі обміну об'єктами (Object Exchange Model).

Information Manifold (http://www.research.att.com/~luy/imhome.html)

Призначена для підтримки інформаційних ресурсів (структурованих і неструктурованих). Архітектура система заснована на базі знань, що містить багату модель предметної області, у термінах якої і відбувається опис ресурсів. Користувач має можливість переглядати інформаційну базу як у виді бази знань, так і у вигляді окремих інформаційних ресурсів, і задавати запит на пошук на декларативній мові запитів. Головною задачею й особливістю цієї системи є можливість оптимізації запиту користувача.

Щодо систем 3-го покоління інтероперабельності відзначимо, що на сьогоднішній день вже існує декілька програм по створенню таких систем. Це:

- Digital Library Initiative (http://ai.bpa.arizona.edu/go/dl/);
- Спільний проект цифрових бібліотек університетів UCSB, UCB, Stanford, CDL, SDSC (http://www.npaci.edu/Thrusts/DI/libraries.html) під патронатом National Science Foundation (NSF);
- Проект по семантичній інтероперабельності у геоінформаційних системах OGC (Open GIS Consortium Inc.) (http://www.opengis.org/interop00.htm),

та інші.

Як можна побачити, у сучасних системах використовуються дві концепції архітектури медіатора:

- Централізований підхід, за яким необхідно існує центральний медіатор. Цей медіатор визначає, до якого ресурсу треба звернутися при відповіді на запит користувача, за допомогою центрального словника даних, або загальної онтології ІС. (TSIMMIS, Information Manifold).

- Децентралізований підхід, в якому кожен ресурс має програмного агента, що відображає онтологічну специфікацію ресурсу в загальну онтологію даної ІС. Відповідь на запит формується після комунікації агентів окремих ресурсів із агентом брокера ресурсів, який, на відміну від центрального медіатора, визначає релевантні ресурси в процесі відповіді на запит, виходячи із доступних/релевантних ресурсів (InfoSleuth, SIMS, HERMES).

6.1.4 Системи спільного використання знань

При спільному використанні різних баз даних цінність представляє обсяг і семантика даних, що зберігається в тій чи іншій базі. У той же час, розробка комплексних систем знань пов'язана зі значними зусиллями по формалізації і представленню знань, тому що цінність буде мати якість і повнота наявних концепцій. Тому поряд із системами інтеграції неоднорідних баз *даних*, не менше значення мають дослідження в області спільного використання компонентів баз *знань*.

Проект IBROW3 [79] призначений для розробки інтелектуального сервісу, що дозволить використання компонентів знань від різних розроблювачів, з використанням технологій WWW. У рамках проекту використовуються і онтології предметних областей, і онтології класів задач, і онтології методів рішення класів задач. Як базову мову для опису бібліотеки методів рішення задач в цьому проекті використовують мову UPML.

Паралельно IBROW3 розвивався проект DARPA за назвою High-Performance Knowledge Bases (HPKB) (http://www.teknowledge.com:80/HPKB). Метою проекту було масштабування повторно використовуваних компонентів знань і доступ до них за допомогою Internet. Нащадком проекту HPKB є проект RKF

(Rapid Knowledge Formation) (http://reliant.teknowledge.com/RKF/), що стартував у 2000 році, і головною метою якого є створення засобів для полегшення роботи розподіленим групам експертів на етапі створення нових баз знань.

6.1.5 Доступ до різнорідних даних в Інтернет

Особливе місце в системах інтеграції неоднорідних компонентів відіграє Інтернет і World – Wide Web. Величезна кількість інформації розсіяна по Мережі, і тут важливість має пошук релевантної інформації і відсікання непотрібної інформації. Сучасні машини пошуку (search engines), застосовувані на пошукових серверах, мають потребу в додаткових засобах пошуку тільки релевантної інформації. Онтології предметних областей, як засоби специфікації онтологічних угод між постачальниками інформації і користувачами, можуть змінити ситуацію до кращого. Використання онтологій дозволяє користувачу сформулювати свій запит на більш високому рівні абстракції, ніж це можливо при пошуку по ключових словах.

Розглянемо приклади систем, що використовують онтології для роботи з Інтернет.

OBSERVER (http://siul02.si.ehu.es/~jirgbdat/OBSERVER)

Ця система пропонує підхід використання безлічі вже існуючих онтологій для доступу до гетерогенних, розподілених і незалежно розроблювальних репозиторіях даних [77]. Реалізація такого підходу - ідеологія брокера онтологій предметних областей. Передбачається, що існує безліч заздалегідь створених онтологій предметних областей, і користувачу необов’язково “підбудовуватися” під конкретну онтологію. Користувач формулює свій запит на деякій мові, у термінах однієї чи декількох онтологій, і брокер «шукає» релевантні інформаційні ресурси, виконуючи транслювання запиту в придатні онтології, а в разі потреби, і сполучення декількох онтологій для більш точної відповіді на запит.

OntoSeek [80]

Ця система розроблена для контекстного отримання інформації з онлайнових “жовтих сторінок” та каталогів продуктів. Система може працювати як з однорідними, так і з неоднорідними каталогами продуктів. Для точної фіксації контексту може бути застосований інтерактивний підхід, коли користувач поступово уточнює зміст ключових слів, за допомогою лінгвістичної бази даних WordNet (http://www.let.uva.nl/~ewn). WordNet – це лінгвістична база даних, що складається із сінсетів(synsets) – груп слів, еквівалентних за змістом. WordNet є водночас і лексичним словником (створеним для декількох європейських мов), і онтологією, що представляє зв’язки між словами у словнику. Опис ресурсу реалізується у вигляді лексичного концептуального графа[81], де вершини відповідають словам, а іменовані дуги – семантичним відносинам між словами (наприклад, відносини типа “частина”, або “підклас”, або ін.), назви вершин і дуг також беруть із WordNet, під час створення концептуального графа конкретного ресурсу. Знаходження ресурсів, відповідних до запиту користувача, базується на порівнянні онтологій (лексичних концептуальних графів) цих ресурсів. А саме, при відборі ресурсів, відповідних до запиту користувача, OntoSeek виконує порівняння концептуального графа запиту із існуючими концептуальними графами ресурсів або з частинами цих графів.

OntoSeek має централізований сервер, на якому знаходиться база даних лексичних концептуальних графів відомих системі ресурсів, але створення таких графів виконується з боку клієнта.

Підхід, використаний в OntoSeek, відрізняється від підходу, який застосовується у моделі W3C Resource Description Framework (W3C RDF, http://www.w3c.org). У RDF опис структури даних (тобто, схема даних у вигляді <subject, predicate, object>), додається прямо у HTML/XML документ, а не зберігається окремо. Ніяких додаткових умов щодо семантичної узгодженості даних RDF не вимагає.

6.2 Методика динамічної модифікації онтологій в інформаційних системах медіаторного типу

Проблема інтеграції інформації (як даних, так і сервісів), спільне та повторне її використання, дуже важлива для багатьох галузей. Відомі системи, які забезпечують отримання даних із гетерогенних розподілених інформаційних ресурсів (ІР) і запитів до таких даних в уніфікованій формі, наприклад, OBSERVER [82], InfoSleuth [83]. Водночас, розробники цих систем підкреслюють статичний характер засобів, використаних для презентації семантики ІР. Велика кількість роботи стосовно модифікації семантики, а саме: (пере-)опис ресурсу, (пере-)реєстрація, та ін., виконується ручним способом, займає багато часу і робочої сили, і є джерелом різноманітних помилок і семантичних конфліктів.

Дослідження можливостей, які б забезпечили керування динамічно змінної семантики ІР є дуже важливим, оскільки існує дві причини необхідності таких змін:

- Семантика ІР може бути поповнена, для того щоб відображати предметну область більш точно
- Самі концептуалізації предметних областей змінюються з часом – гарним прикладом може бути підтримка своєчасних змін у контрольованій медичній термінології (обговорюється в [84])

Очікуємо, що сучасні вимоги до інтелектуальних медіаторних інформаційних систем, які оперують на множині різноманітних оболонок ІР, і пропонують розподілені сервіси, будуть містити такі:

- Інформаційна система(ІС) повинна бути здатна оперувати гетерогенними розподіленими ІР
- ІС повинна мати якийсь уніфікуючий семантичний базис для забезпечення інтеграції ресурсів і сервісів із різною семантикою

– ІС повинна бути здатна оперувати автономними ІР і сервісами, які були розроблені і підтримуються незалежно, динамічно приєднуються до ІС або лишають її

Одна із ключових проблем для розробників медіаторних ІС є забезпечення рішень щодо інтероперабельності. Проблема інтероперабельності є дуже складною і може розглядатися із різних боків. Важливим аспектом є задача семантичної інтероперабельності з урахуванням змін семантики з часом.

Проблема семантичної інтероперабельності часто декомпозується на такі підзадачі:

А) (Пере-)опис, (пере-)реєстрація семантики ресурсу.

В різних рішеннях цієї підзадачи використовуються такі види онтологій: онтологія ІР, загальна онтологія ІС, онтологія домену. *Онтологія ІР* представляє часткове відображення домену з точки зору певного ІР і створюється / поновлюється під час життєвого циклу ІР. Онтології ІР можуть бути отримані із вже існуючих ресурсів [85]. Також, онтології ІР можуть бути використані на етапі розробки ІР [86], [87]. *Загальна онтологія ІС* забезпечує уніфікуючі специфікації концептуалізації доменів, які були отримані шляхом відображення онтології ІР (це поняття близьке до поняття загальної онтології посилань, [88]). Такі відображення створюються під час (пере-)реєстрації ресурсу в ІС. Створення загальної онтології ІС може також виграти від використання онтологій доменів. *Онтології доменів* відображають семантику домену.

Б) Моніторинг змін семантики ресурсу і відповідних змін семантики домену.

Рішення цієї підзадачи повинні включати засоби для визначення критичної маси змін і ініціації процедур, які створюються для рішення підзадач А і В.

В) Забезпечення узгоджених змін описів семантики ресурсів у відповідь до змін в уявленні про предметну область, яке представляє конкретний інформаційний ресурс, а також коректна модифікація відображення цих модифікацій в загальній онтології ІС.

Якщо опис структури ресурсу вважається прикладом опису семантики домену цього ресурсу, рішення останньої підзадачи буде складатися із засобів, які можуть ініціалізувати повторне відображення описів структури ресурсу у відповідь на сигнали із сервісу моніторингу (підзадача Б).

Рішення даних підзадач можуть розглядатися у вигляді сервісів медіаторної ІС. Така ІС повинна мати доступ до сервісів оболонок ІР, мати свої власні сервіси і забезпечувати відгуки, які можуть ініціалізувати роботу сервісів на боці оболонки ІР. Створення таких сервісів буде дуже близько пов’язане із рішеннями проблеми семантичної інтероперабельності.

Даний звіт пропонує підхід для підтримки динамічно змінних онтології ресурсів і їх відображень в загальну онтологію медіаторної ІС. Онтології вважаються головною частиною опису семантики ресурсів. Запропонований підхід виконує підтримку змін економічно, обробляючи тільки такі елементи онтологій, що змінилися.

Медіаторний підхід для маніпулювання семантично гетерогенними, розподіленими, автономними ІР, які представляються в ІС своїми оболонками, які мають набори сервісів є відносно новою галуззю для досліджень. Своє походження цей підхід бере з появою нових можливостей в галузях віртуальних підприємств (VEMS), цифрових бібліотек (DLIB), а також в CSCW, E-Commerce. Проблема інтероперабельності для незалежних ІР із змінною семантикою плідно розроблялась в теорії баз даних. Відомі декілька підходів для підтримки змінної семантики в базах даних: техніка модифікаційних примітивів, версії схеми, види схем. Наприклад, Active Data Dictionary (ADD) [24] – це середовище, в якому забезпечено адаптацію схем реляційних баз даних до змінної предметної області. В [89] презентований підхід моніторингу змін в схемах баз даних для федеративних систем баз даних.

Цікаві пропозиції щодо підтримки змін у схемах в об’єктно-орієнтовних системах баз даних були розглянуті в [90], [91]. Так, Бенаталлах [91] дає опис

середовища для підтримки змін схеми, в якому методології модифікаційних примітивів і версій схеми комбіновані.

Задача визначення відповідностей між схемами ресурсів (матчінг схем) є дуже близькою до нашої задачі підтримки модифікацій. В обох випадках критерії для знаходження відповідностей в семантиці різних ресурсів, так само як і для міркування про відповідні модифікації семантики базуються на парадигмі інваріантів. Таксономія, яка класифікує більшість із відомих підходів для матчінгу схем, а також пропозиції щодо проведення матчінгу схем взагалі, показана в [92].

Методологія модифікаційних примітивів для баз знань використовується в OKBC [93] для доступу і модифікацій елементів знань, які є в OKBC-сумісних (орієнтованих на фрейми) систем презентації знань.

Дослідження по інтеграції розподілених IP розглядають проблеми синхронних модифікацій, злиття і вирівнювання онтологій. Підходи Chimaera [94] і PROMPT [95] дають гарні приклади середовищ підтримки процесів інтеграції. Злиття і вирівнювання онтологій. Алгоритм PROMPT [95] може бути застосовано для випадків коли онтології ресурсів модифікуються в процесі життєвого циклу відповідного IP, і їх потрібно злити або вирівняти з загальною онтологією IC.

Особлива увага приділяється еволюції онтологій в проекті On-To-Knowledge [96]. Полуавтоматична інтеграція онтологій [97] і збереження версій онтологій розглядаються консорціумом On-to-Knowledge як важливі і необхідні для рішення проблем успадкування інформації і проблем знаходження відповідностей.

6.2.1 Модель онтології: описи і модифікації

За натуральною аналогією модель онтології будемо позначати як структуру, яка складається із двох компонент: опис онтології і методи для маніпуляцій з онтологією.

Нехай *модель онтології* буде структурою $< O, \Omega >$, де:

O є (згідно з [98]) трійка $< X, \Re, A >$ з X - скінченною множиною термів, $\Re$ - скінченною множиною відношень між термами, A - скінченною множиною обмежень

Ω - є множиною модифікаційних примітивів, які застосовуються до елементів O

В презентованому дослідженні *базовими елементами* онтології будуть такі: X_1 – множина концепцій, X_2 – множина властивостей, $\Re$ – множина відношень, серед яких є відношення ‘is-a’ (“є”), A – множина обмежень на концепції, властивості і відношення. $X = X_1 \cup X_2$ і $X_1 \cap X_2 = \varnothing$.

За допомогою таких базових елементів можемо описувати онтології таких формальних типів:

- простий словник незв’язаних термінів

$$X \neq \varnothing, \Re = \varnothing, A = \varnothing \tag{6.1}$$

- “Легковажна онтологія”, наприклад: Thesaurus Ontology [88]

$$X \neq \varnothing, \Re \neq \varnothing, A = \varnothing \tag{6.2}$$

- Таксономія

$$X \neq \varnothing, \Re = \{is - a\}, A = \varnothing \tag{6.3}$$

- Пасивний словник

$$(X = X_1 \cup X_2) \neq \varnothing, \Re = \varnothing, A \neq \varnothing \tag{6.4}$$

- Загальний тип онтології (надає багатий словник термінів разом із багатою множиною семантичних відношень, елементами $\Re$ є “part of” (“частина”), “kind of” (“сорт”), ... [98]):

$$(X = X_1 \cup X_2) \neq \varnothing, \Re \neq \varnothing, A \neq \varnothing \tag{6.5}$$

Аналіз відомих мов, які використовуються для опису онтології (KIF [35], Ontolingua [99], OKBC [93], OIL [100], SYNTHESIS [101]) показує, що всі вони

не забезпечують засобів для *декларативного* опису маніпуляцій над елементами онтології (як це було зроблено, напр., в SQL/DDL, SQL/DML).

В даній статті примітиві модифікації онтології вважаємо декларативними засобами для маніпулювання елементами онтології. На самому вищому рівні ідея додання в модель онтології маніпулятивної частини близька до підходу модифікації схем OODB за допомогою модифікаційних примітивів (див. [102]).

Модифікаційні примітиви над елементами онтології загального типу будуть такі:

```
CONCEPT:          add, delete (retire), rename
PROPERTY:         add, delete (retire), rename, change
domain,
                change constraints on property values
RELATIONSHIP:     add, delete (retire), rename
ASSERTION RULE:   add, delete
```

Техніка модифікаційних примітивів часто комбінують з технікою версій (напр. [84], [91]). Тому розумніше буде замість простого вилучення елементів онтології, робити такі елементи застарілими. Примітиви модифікації онтології є низькорівневими методами, на базі яких можна конструювати більш складні операції маніпулювання елементами онтологій. Такі операції в цій статті не розглядаються для стислості.

Прості приклади специфікацій примітивів модифікації елементів онтології у вигляді OKBC – процедур наведені в таблицях 6.1, 6.2. В більш складних випадках алгоритми для коректного розповсюдження змін між пов’язаними елементами одної онтології потрібно буде задати.

Нехай фраґмент реляційної бази даних (IP), ER –схема якого дана на Рис. 6.1(а), змінюється: сутність, що представляє концепцію E0 вилучена, і додані нові концепції E1 (A0, A1, A3) та E2 (A0, A2, A4). Вони з’являться як нові концепції в онтології IP. Відповідна програма на OKBC показана на Рис. 6.1(б).

Ті елементи онтології IP, які змінилися, пізніше (після того, як сервіс моніторингу IP відзначить, що маса змін в онтології стала критичною) будуть

вирівняні з елементами загальної онтології ІС. Більш докладніше про це дивиться в п.6.2.5.

6.2.2 Інваріанти модифікації таксономії

Реалізація засобів для модифікації онтологій як для сервісів з боку медіатора ІС, так і для сервісів з боку оболонки ІР, буде залежати від вибору мови презентації і маніпулювання онтології. Інваріанти модифікації онтології є обмеженнями на можливі зміни елементів онтології (згідно з [58]). Інваріанти будуть різними залежно від типу онтологій. Також вони будуть різними залежно від формальної семантики тієї чи іншої мови презентації онтології. Такі інваріанти можуть біти використані в різних задачах:

- при (пере-)реєстрації – відображення онтологій ІР в загальні онтології ІС;
- при підтримці змін елементів онтології – матчінг змінених елементів онтології ІР з елементами загальної онтології ІС;

Для стислості в цій секції буде розглянуто набір інваріантів модифікації для *таксономії* (див. 6.4) – найбільш поширеного типу онтологій.

І1: **Інваріант ієрархії концепцій**. Ієрархія концепцій в таксономії повинна зберігатися як зв'язний направлений ациклічний граф - дерево. Ієрархія наслідування має тільки один корінь. Концепції повинні мати унікальні імена.

І2: **Інваріант різних імен**. Необхідно, щоб всі властивості і відношення, пов'язані з концепцією, ті, що наслідувані, та ті що визначені для цієї концепції, мали імена, що відрізняються.

І3: **Інваріант єдиного походження**. Всі властивості концепції повинні мати тільки одне походження.

Таблиця 6.1 - Модифікації категорії елементів CONCEPT

Елемент Онтології і відповідні модифікаційні примітиві		**Відповідна конструкція OKBC**
CONCEPT – головний елемент онтології ресурсу		**FRAME** – презентує **CONCEPT** незалежно від ієрархії вже існуючих концепцій
“Add concept”	ДОдає концепцію з унікальним ім’ям і пустою множиною властивостей	**create-frame ()**
“Rename concept”	Дає концепції нове ім’я	**put-frame-name ()**
“Delete (Retire) Concept”	Вилучає концепцію. Якщо існує властивість з областю визначення. Яка дорівнює. Або є підмножиною значень концепції, яка підлягає вилученню -> заборонити вилучення доти, доки не зміниться область визначення цієї властивості	**delete-frame ()**

Таблиця 6.2 - Модифікації категорії елементів PROPERTY

Елемент Онтології і відповідні модифікаційні примітиві		**Відповідна конструкція OKBC**
PROPERTY – характеристика CONCEPT. Може презентувати відношення між CONCEPTS		**SLOT** – презентує властивість, **PROPERTY**
“Add property”	Додає властивість до даної концепції (якщо така існує), властивість повинна мати унікальне для цієї концепції ім’я. Враховуючи всіх батьків цієї концепції.	**create-slot ()** **attach-slot ()**
“Rename property”	Дає властивості нове унікальне ім’я.	**rename-slot ()**
“Change the domain of property”	Змінює область визначення значень властивості, враховуючи всі дочірні концепції. Якщо нова область визначення є підмножиною екстенсіоналу деякої концепції, перевірити, чи визначена та концепція.	1. Для домену із одного значення: **replace-slot-value()** 2. Для багатозначних доменів, які презентуються як множина або перелік значень: **add-slot-value()**,

Елемент Онтології і відповідні модифікаційні примітиві		Відповідна конструкція OKBC
		put-slot-value() 3. Для області, що визначається фреймом, набір операторів OKBC
“Delete (Retire) property”	Вилучає властивість із концепції і із всіх дочірніх концепцій, окрім випадків, коли ця властивість була перевизначена в дочірній концепції як локальна	**delete-slot()**

I4: **Інваріант повного наслідування**. Концепція повинна наслідувати всі властивості із кожного з своїх батьків, крім випадків, коли таке наслідування порушує інваріанти **I2** і **I3**.

I5: **Інваріант наслідування областей визначення властивостей.** Якщо властивість з’являється і в самій концепції, і в її дочірніх концепціях, тоді наслідувана властивість повинна мати область визначення таку як і батьківській концепції, або бути її підмножиною.

Головна функція інваріантів модифікації онтології є проведення коректних модифікацій. Якщо порушується один із інваріантів, модифікації елементів онтології можуть призвести до різних типів конфліктів (конфлікту імен, областей визначення та ін.). Процедури вирішення конфліктів будуть залежати від мови і від системи бази знань, використаної для реалізації. Все ж таки є можливість показати правила вирішення конфліктів у незалежнім від мови вигляді для таксономії. За основу були взяти правила з [102].

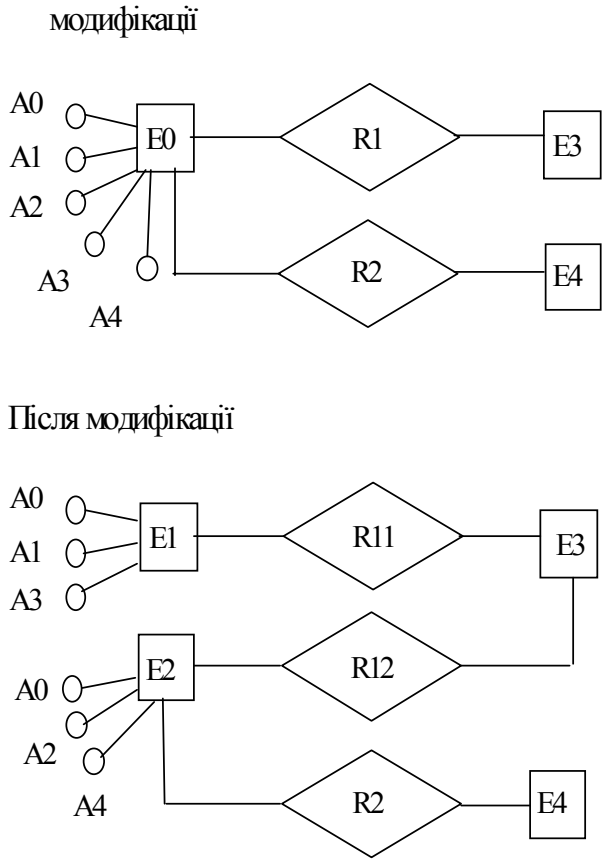
Перша група правил регулює вирушення конфліктів, спричинених наслідуванням і перевизначенням властивостей в дочірніх концепціях.

R1: Якщо властивість визначена в концепції локально, і її ім’я співпадає із ім’ям іншої , наслідуваної властивості, тоді локально визначена властивість перекриває наслідувану.

R2: Якщо дві або більше батьків концепції мають властивості з одним ім’ям, але з різними походженнями, обидві властивості наслідуються і

г(a)

а



ім(b)

```
if not frame-in-kb-p(E1 kb true)
 { create-frame (E1 :class true)
    attach-slot
         (E1 A0 kb (:template) true)
    attach-slot
         (E1 A1 kb (:template) true)
    attach-slot
         (E1 A3 kb (:template) true)
    create-slot (R11 kb)
    attach-slot
         (E1 R11 kb (:template)true)
 } else /*omitted*/
if not frame-in-kb-p(E2 kb true)
 { create-frame (E2 :class true)
    attach-slot
         (E2 A0 kb (:template) true)
    attach-slot
         (E2 A2 kb (:template) true)
    attach-slot
         (E2 A4 kb (:template) true)
    create-slot (R12 kb)
    attach-slot
         (E2 R12 kb (:template)true)
    attach-slot
         (E2 R2 kb (:template) true)
 } else /*omitted*/...
delete-frame (E0 kb true)
/* kb – is the name of corresponding
      resource ontology*/
```

Рисунок 6.1 – (a) Приклад модифікації структури ресурсу;
(b) відповідний алгоритм модифікації (OKBC)

R3: Якщо два або більше батьків концепції X мають властивість з одним ім'ям і з одним походженням, ця властивість наслідується тільки один раз. Якщо одна з таких властивостей була локально перевизначена у одного із батьків, вона має бути перевизначена і в дочірній концепції X.

Друга група правил стосується розповсюдження модифікацій в дочірніх концепціях.

R4: Модифікація властивості в концепції завжди розповсюджується на всі її дочірні концепції, крім тих, в яких ця властивість була локально перевизначена.

R5: Створення властивості в концепції вимагає щоб не було локально визначеної властивості в цій концепції за таким же самим ім’ям. Більш того, це потребує присутності не більш як однієї дочірній концепції, яка б містила властивість з таким же ім’ям.

R6: Тільки локально визначені властивості можуть бути вилучені з опису концепції.

R7: Модифікація імені властивості в концепції не розповсюджується на дочірні концепції.

Третя група правил стосується агрегації і вилучення відношень наслідування між концепціями, а також створення і вилучення відношень наслідування.

R8: Якщо концепція *Y* додається до переліку батьківських концепцій концепції *X*, всі конфлікти наслідування вирішуються правилами **R1**-**R3**.

R9: Вилучення концепції *Y* із переліку батьківських концепцій концепції *X* вилучає всі наслідувані властивості в *X* та в її дочірніх концепціях. Якщо *Y* була останньою концепцією в переліку батьків, вилучення *Y* робить *X* прямим нащадком кореневої концепції.

R10: Нові концепції можуть бути створені тільки як листя в таксономії. Якщо нова концепція *X* створюється без будь-якої індикації батьків, *X* стає нащадком кореневої концепції.

R11: Тільки “листові” концепції можна вилучити із таксономії.

6.2.3 Сервіси моніторингу і модифікації

Однією з тенденцій, яка має місце в сучасних інформаційних технологіях є зміщення акцентів від оперування колекціями даним до забезпечення множини інтелектуальних сервісів в Мережі.

Це означає, що сучасна генерація інформаційних систем медіаторного типу вимагає, щоб інформаційні ресурси надавали сервіси такій ІС для збору і

агрегації інформації у відповідь на запит користувача. Аналогічні архітектурні рішення можуть бути знайдені в InfoSleuth, OBSERVER, TSIMMIS [75], MOMIS [103]. Більшість цих систем є агентськими. Це означає, що разноманітні ресурси. Такі як впровадження оболонок ІР, формулювання запитів на пошук інформації, декомпозиція запитів, моніторинг ресурсів та інші, виконуються командами або коаліціями інтелектуальних програмних агентів.

Застосування агентів в медіаторних архітектурах привносить новий аспект семантичної інтероперабельності – інтероперабельність агентів. Одним із очевидних рішень для цього є введення агента онтології. Який би забезпечив загальні онтології для коаліцій агентів.

Агент онтологій, або обраний агент моніторингу (як в InfoSleuth) також може бути застосований для виконання завдань, перерахованих в п. 6.2.1: моніторинг змін в семантиці ІР, ініціація процедур (пере-)регістрації, (пере-)відображення онтології ІР в загальну онтологію ІС.

Деякі аспекти для розробки поведінки моніторингу такі. Зміни в онтології ІР можуть бути не дуже критичними, або критичними:

– Перейменування концепцій, властивостей, відношень не є дуже критичними модифікаціями онтології ІР, і не потребують негайної перереєістрації ІР в медіаторній ІС. Такі модифікації не впливають на ієрархію концепцій, і тому, необхідно тільки обновити відображення елементів онтології ресурсу відповідно до змін семантики в цьому ресурсі.
– Вилучення властивостей або концепцій використаних для визначення інших елементів онтології ІР, додавання нових концепцій і властивостей до ієрархії концепцій є дійсно критичними для онтології ІР, і потребують негайної перереєстрації і перевідображення елементів онтології.

Агент моніторингу в медіаторній ІС повинен бути толерантним до змін, якщо такі зіни не перевищують певне порогове значення. Поява змін другого типу повинна ініціювати процес перереєстрації.

Існує два можливих шляхів для розповсюдження змін: від онтологій ресурсів до загальної онтології ІС, і навпаки.

Нехай ініціаторами змін семантики у медіаторній ІС будуть оболонки ІР. Тоді сервіс загальної онтології ІС повинен мати засоби для модифікації своєї онтології, і для матчінгу / вирівнювання елементів онтології ІР аз елементами загальної онтології ІС. Алгоритми розповсюдження змін для прикладу із Секції 3 можуть бути сформульовані так:

Нехай: Sch_i – схема *i*-го ресурсу; O_i - онтологія *i*-го ІР; CO – загальна онтологія ІС; $C_A, C_D, C_R,$ – множини доданих, застарілих, перейменованих концепцій, $A_A, A_D, A_R, A_{CD}, A_{CC}$ – множини доданих, застарілих, перейменованих, таких, що змінили область визначення, таких, що змінили свої обмеження, атрибутів; R_A, R_D, R_R – множини доданих, застарілих, перейменованих відношень, AR_A, AR_D – множини доданих, застарілих обмежень.

З боку оболонки ресурсу алгоритм буде такий:

А) Сервіс оболонки (WOS) перевіряє структуру ресурсу і знаходить такі зміни: концепція E0 – застаріла, концепції E1 і E2 – додані. Відношення R1 – застаріле, відношення R11, R12 – додані. Властивості A0, A1, A2, A3, A4 – вилучені з концепції E0, потім властивості A0, A1, A3 – додані до концепції E1, властивості A0, A2, A4 – додані до концепції E2.

Б) WOS модифікує онтологію свого ресурсу полуавтоматично, перевіряє виконання інваріантів, і якщо необхідно, модифікує відображення елементів онтології відповідним елементам схеми ресурсу.

В) WOS генерує повідомлення про зміни.

З боку сервісу оболонки загальної онтології ІС (CIOS):

А) CIOS приймає повідомлення про зміни.

Б) CIOS виконує матчінг доданих і перейменованих елементів. Якщо результат матчінгу для доданих елементів є негативним, додає ці елементи до загальної онтології ІС..

B) Для застарілих елементів онтології IP CIOS перевіряє, чи існує інші онтології ресурсів, які містять подібні елементи. Якщо результат пошуку негативний, CIOS помічає елементи як застарілі.

6.3 Онтології ЄІП

6.3.1 Онтології завдання і переговорів: Забезпечення семантичної інтероперабельності

Аспект подання семантики завдання, діяльності, ЧЛП. Компромісу, відгуку є дуже важливим на різних ступенях спільного виконання завдання. Ініціатор приймає рішення про те, як декомпозувати і виконати (під)завдання/діяльність, або які нові діяльності необхідні для виконання вже заданих для виконання, або які із діяльностей ініціатор може передати партнерам, згідно із знаннями, що представлені у формі Онтології Завдання. Онтологія завдання дана на Рис. 6.2 в ER-стилі. Ініціатор і виконавці також повинні використовувати спільну семантику в процесі торгів на призначення виконавців. Вони використовують для цього Онтологію Завдання і Онтологію Переговорів. ER- діаграма для Онтології Переговорів представлена на Рис. 6.3. Ці онтології написані на Standard OIL [50].

Спілкування між ініціатором і учасниками на всіх етапах процесу торгів на призначення діяльності (див. Рис. 5.1) організоване згідно з [34]. Використані базові примітиві KQML. Ініціатор **I** розповсюджує запит "`Evaluate the Activity`" всім партнерам, які як **I** вважає згідно з (2), можуть виконати діяльність, і яким, згідно з поточним значенням міри довіри (19), можна довіряти, як кластер наступних розпоряджень в стилі KQML:

```
(ask-one
    :sender      "I"
    :receiver        "P"
    :in-reply-to Null
    :reply-with      P-TDF
    :language        (XML)
    :ontology        (Trade-off)
```

```
:contents          (<опис завдання выдповыдно онтологіі
                     завдання>  )
```

Учасники відповідають з параметричними відгуками, які показують їх готовність виконати діяльність, з таким розпорядженням:

```
(tell
    :sender      "P"
    :receiver        "I"
    :in-reply-to P-TDF
    :reply-with      Null
    :language        (XML)
    :ontology        (TDF-Feedback)
    :contents        ( $\tilde{tdf}(t,d)$ )    )
```

Онтології Завдання і Переговорів були розроблені і формалізовані у мові Standard OIL і служать як спільні концептуалізації для моделювання спільного виконання завдань в коаліціях агентів, що надають сервіси. Необхідність розробки таких онтологій була мотивована властивостями неповноти і неточності, властивих галузі B2B E-Commerce, а також необхідністю витримувати баланс між раціональністю і доброзичливістю.

Роль онтології завдання полягає в тому, щоб забезпечити формальну презентацію понять: завдання, діяльність, параметр, шаблон результату, зусилля, пріоритет, час закінчення, бюджет, а також частковий локальний план (ЧЛП). Ці поняття використовуються агентами, що представляють сервіси, для того щоб визначити: чи є діяльність атомарною, чи можуть агенти виконати такі діяльність, чи співпадають результати і параметри діяльності тим, що закладені у базу знань агента, чи потрібно шукати співвиконавців для виконання діяльності. Діаграма в стилі ER для онтології завдання показана на Рис. 6.2.

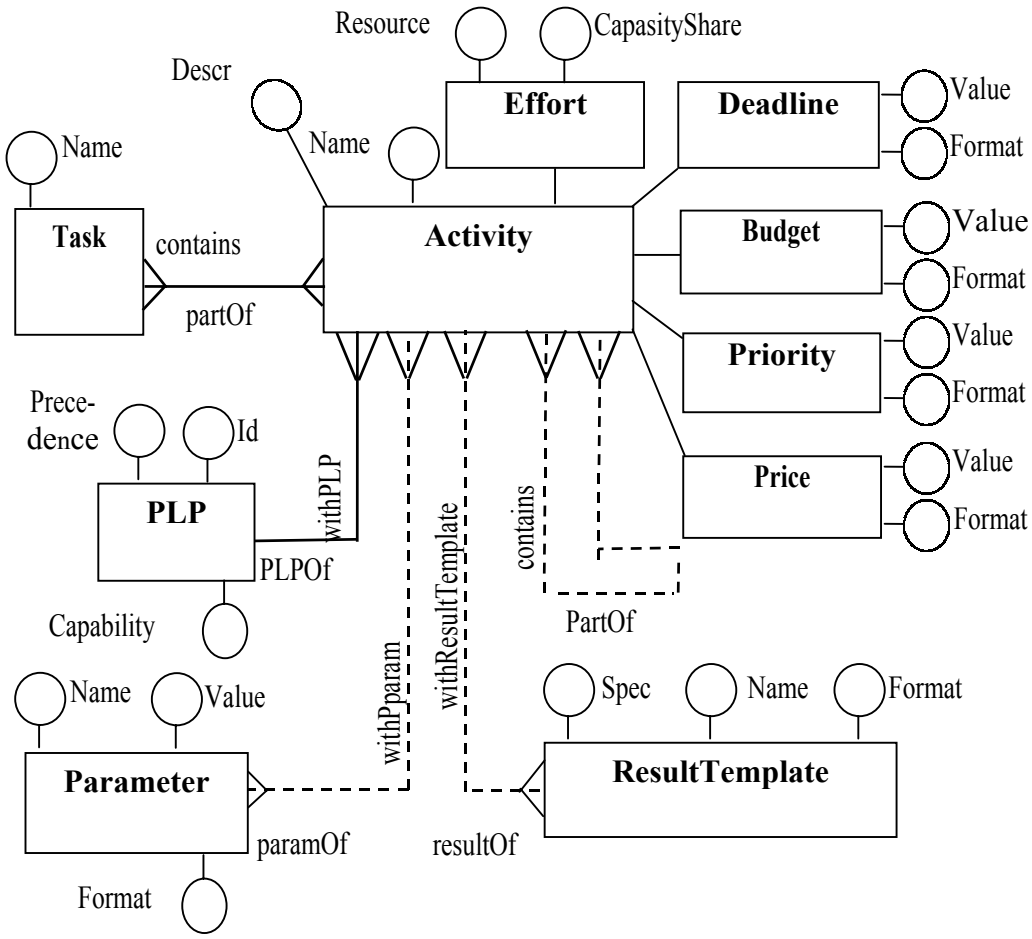


Рисунок 6.2 – ER-діаграма для онтології завдання

Онтологія переговорів надає спільну концептуалізацію термінів, що їх використовують агенти в процесі переговорів на назначення діяльності. Тип переговорів для фази назначень є контрактування. Відміна запропонованого підходу до переговорів від інших методів проведення переговорів полягає у застосуванні параметричного відгуку, що базується на понятті компромісу. Така параметризація забезпечує більше гнучкості у поведінці агентів і дозволяє обійтися без ітерацій по переговорам. Концепції онтології переговорів є такими: діяльність шаблон результату, бажаність результату, запропонований час закінчення, час і його розмірність, стимул, точка компромісу, відгук з очікуваною оцінкою компромісу. Діаграма в ER-стилі для онтології переговорів показана на Рис. 6.3.

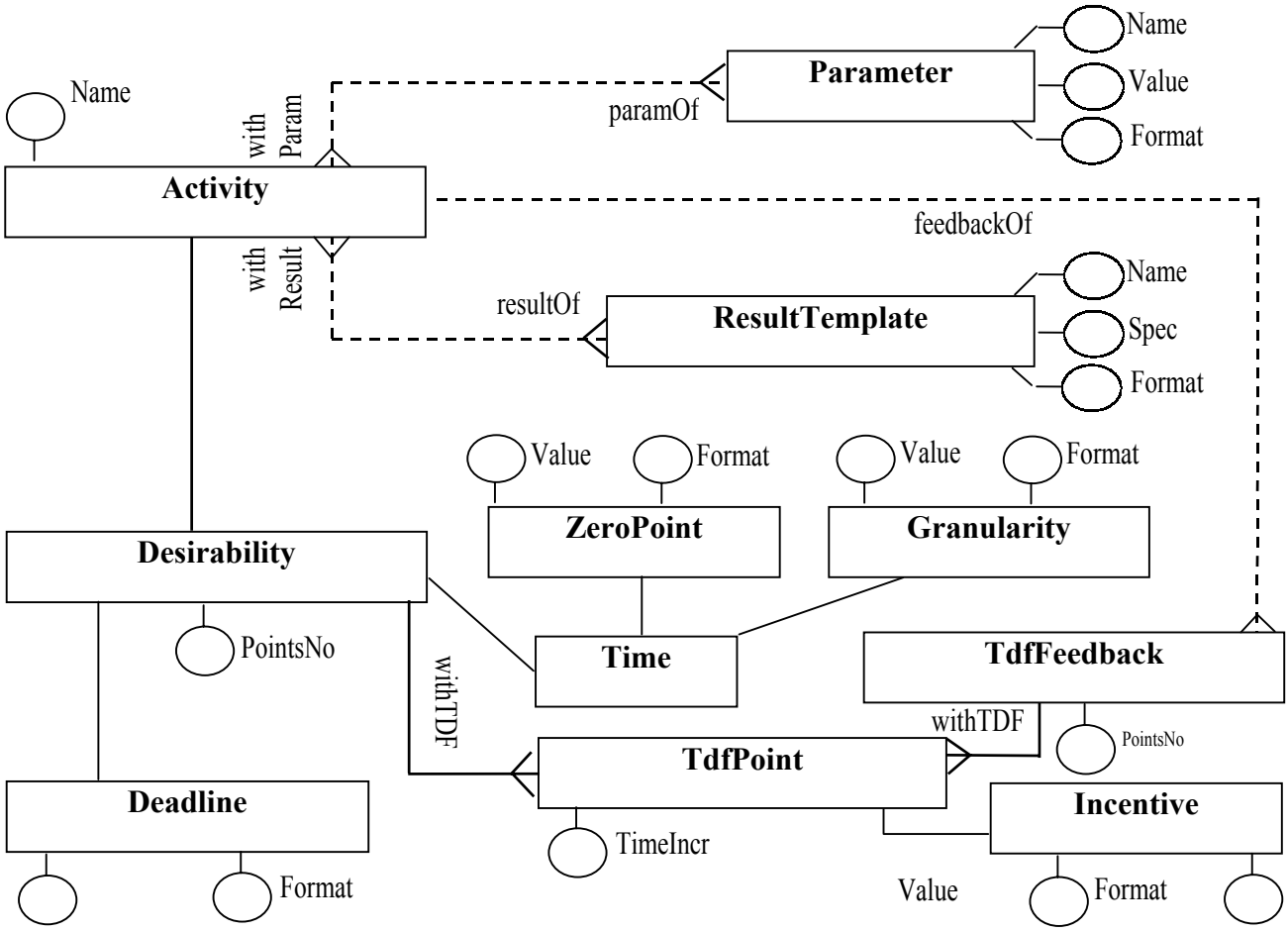


Рисунок 6.3 – ER-діаграма для онтології переговорів

Редактор онтологій OilEd версії 2.2a і машина виводу FACT були використані для розробки і верифікації онтологій. Версії онтологій у мовах OIL, RDFS, DAML і SHIQ доступні на сайті http://eva.zsu.zp.ua/eva_personal/ontologies/.

Зразки координації для розподіленого виконання діяльностей в галузі E-Commerce повинні брати до уваги баланс таких властивостей акторів, як раціональність і доброзичливість. В рамках даного підходу, що зараз на стадії розвитку, ця мета може бути досягнута через властивість акторів динамічно формувати коаліції для оптимального виконання завдання. Формування коаліції координоване контрактовими переговорами на назначення тієї чи іншої діяльності.

6.4 Підсумки

У даному розділі звіту був конкретизований стан сучасних систем інтеграції неоднорідних баз даних і баз знань, і підходів, використаних у цих системах.

На сьогодні, задача інтеграції неоднорідних ресурсів має багато рішень. Однак, серед узагальнюючих факторів при рішенні даної задачі виділимо такі:Перший фактор – використання онтологій як специфікацій предметних областей, задач, методів рішення задач.

Другий фактор – розвиток інтернет-технологій, що дозволяють зробити різнорідні інформаційні ресурси «ближче до користувача».

Третій фактор – застосування систем інтелектуальних агентів для реалізації архітектури медіатора неоднорідних розподілених інформаційних ресурсів.

Втім, дійсний момент – тільки перший етап у використанні цих факторів повною мірою.

Процес модифікації онтології взагалі дуже складно перекласти на програмне забезпечення, маючи на увазі можливі наслідки, які впливатимуть на всю інформаційну систему. Для середовищ, які характеризуються наявністю і локальних онтологій, розроблених і керованих автономно, і загальної онтології ІС, Uschold [88] запропонував рішення проблеми модифікацій з організаційною точки зору: створення спеціальної групи людей-експертів, які відповідали б за контроль (моніторинг) змін в онтологіях і синхронізували онтології ресурсів. Однак, навіть при такому рішенні корисною буде програмна підтримка, наприклад, для проведення всіх змін коректно, з вирішенням всіх конфліктів. В даному розділі презентований підхід до організації такої інтелектуальної підтримки, а саме для проведення моніторингу і модифікації.

Онтології були класифіковані за виразною силою їх формальних теорій (6.2)-(6.5) і за їх функцією в медіаторній ІС – онтології ІР, загальна онтологія ІС і онтології предметних областей. Запропонована модель онтології $< O,\Omega >$ яка складається із двох частин: декларативної $O =< X_1, X_2, R, A >-$ для опису

елементів концептуалізації, і маніпулятивної –набору примітивів модифікації Ω. Дана структура загального типу онтології (6.5). Модифікаційні примітиві забезпечено для кожного структурного елементу онтології загального типу. Набір інваріантів модифікації і набір правил вирішення конфліктів сформульовано для таксономії. Також окреслені функції сервісів ІС для проведення моніторингу онтологій, матчінгу і вирівнювання онтологій.

Одним із важливих висновків вважаємо такий, що добре знайомі ідеї і підходи із концептуального моделювання, засновані на модифікаційних примітивах і інваріантах, які відомі вже багато років, можуть бути використані і для нових галузей, пов’язаних з маніпуляціями з онтологіями, версіями, відображеннями онтологій, їх матчінгом і вирівнюванням.

Роль онтології завдання і переговорів, запропонованих в даному розділі, полягає в забезпеченні спільної концептуалізації концепції, структур і процедур, що їх використовують агенти в процесах аналізу діяльності, її декомпозиції, виконання або передачі іншим агентам. Ці онтології формалізовані за допомогою мови OIL і переведені в нотації DAML (RDF), RDFS, SHIQ, таким чином стає можливим використання цих онтологій в стандартах розмітки сервісів, що розробляються у W3C.

До великої кількості запланованої на майбутнє роботи належать: моделювання і концептуалізації альтернативних зразків переговорів (ітераційні переговори, аукціон…); подальше покращання онтології завдання шляхом забезпечення методів вирішення проблем і описом макромодельних процедур (що може вимагати використання розширень мови OIL); розробка прототипу B2B ринку консультацій по інвестиціям в агентській методології.

7 Приклади моделювання процесів у ЄІП

В даному розділі звіту розглянемо приклади моделювання типових процесів, які мають місце у ВУЗі. Перший приклад - є приклад моделювання процесу планування науково - технічної роботи [34], другий - є приклад, пов’язаний із процесами дистанційного навчання у віртуальному університеті [36]. Останній приклад розглядає застосування онтологій завдання і переговорів у процесах спільного виконання завдань в умовах відкритої системи раціональних акторів, яка є типовою для галузі електронної комерції.

7.1 Функціональна модель видавничого центру

Для ілюстрації застосованих підходів до моделювання розглядаємо типовий для аналізу віртуального інформаційного простору приклад підрозділу, функціональний опис якого дає відправну точку для розробки більш загальних принципів моделювання. Прикладом був обраний видавничий центр університету. Специфіка цього підрозділу вдало поєднує виконання як добре формалізованих технологічних функцій, так і участь у дослідницьких проектах, що припускає використання процедур аналізу, оптимізації, а також відсутність детального планування і чітко визначених статичних зв’язків. Крім того, специфіка підрозділу така, що в ньому співпрацюють як реальні штатні співробітники, такі, що мають статичні технологічні обов’язки, які виходять із існуючих технологічних ланцюгів, так і віртуальні компоненти – групи, які беруть участь у виконанні того чи іншого дослідницького проекту.

В рамках даної роботи розглянути тільки реальні (довгострокові) функціональні компоненти, які еволюціонують в рамках свого функціонального поля.

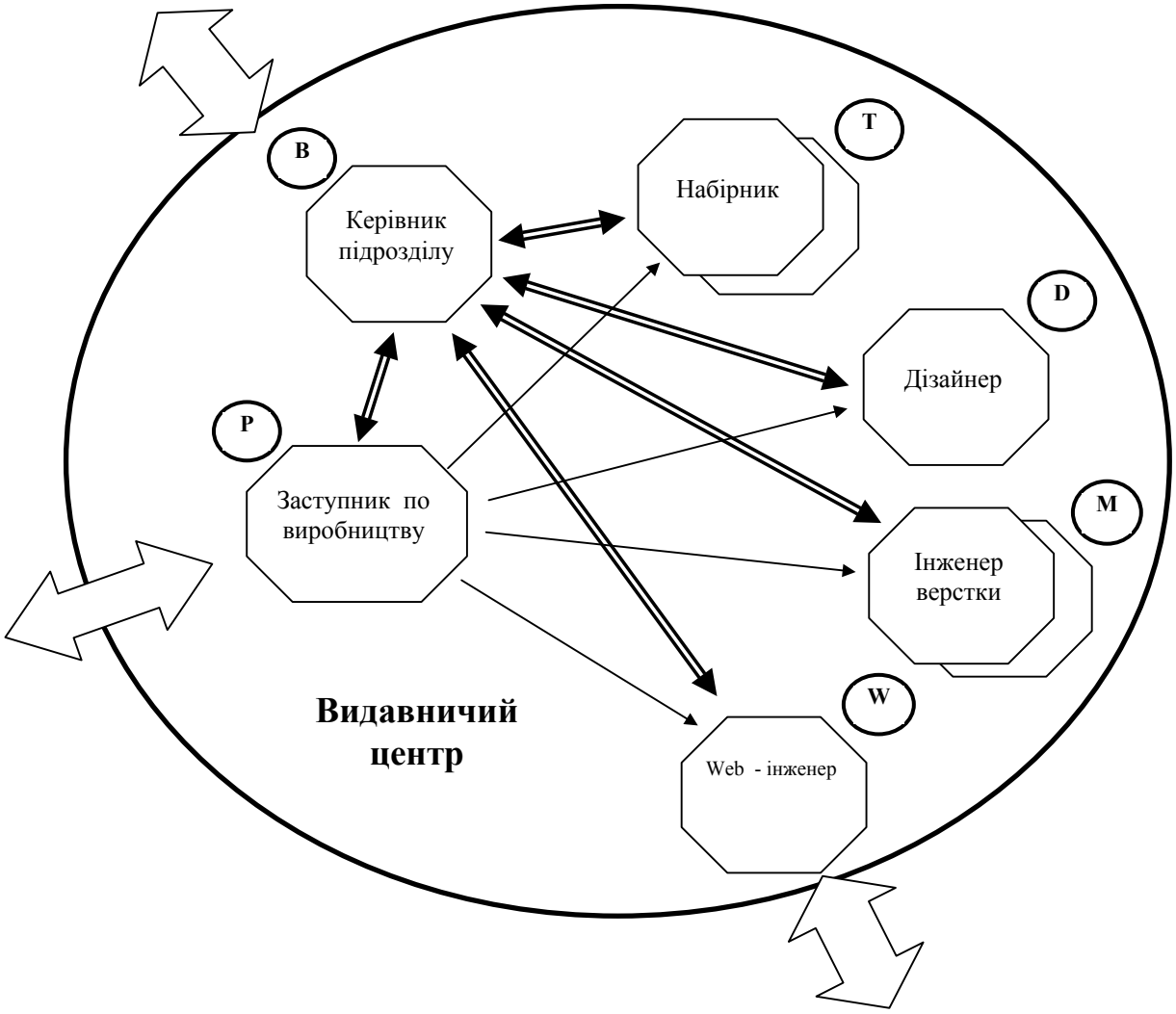


Рисунок 7.1 – Функціональні елементи і їх взаємодія у Видавничому центрі.

7.1.1 Структура і функції підрозділу, зовнішні зв'язки

Реальна складова Видавничого центру (Рис. 7.1) включає в себе такі функціональні компоненти: директор центру, заступник по виробництву, набірник, дизайнер, інженер верстки, Web - інженер. Стрілками на Рис. 7.1 частково показані різні типи зв'язків між функціональними елементами підрозділу: рівноправні, зв'язки підпорядкування. В загальному випадку, граф функціональних зв'язків між функціональними елементами є NP - повним. Цікавішими є не взаємодії в контексті підрозділу, а в контексті виконання

підрозділом деякої функції. Важливо відмітити, що виконання підрозділом тієї чи іншої функції викликається впливом на підрозділ зовні. Такі впливи показані на Рис. 7.1 широкими двонапрямковими стрілками.. Для сприймання зовнішніх впливів виділяються деякі функціональні компоненти. В схемі розглянутого підрозділу такими елементами є директор центру, заступник по виробництву, Web - інженер. Далі буде показано, що топологія взаємодії функціональних елементів підрозділу змінюється як у зв'язку з вибором функції для виконання, так і в процесі виконання обраної функції.

Ось деякі приклади функції, які виконує видавничий центр:

- Прийом у клієнта замовлення на випуск поліграфічної продукції;
- Надати клієнту електронну публикацію;
- Дати довідку о стані процесу прийняття статті до видавництва у журналі;
- прийняти замовлення на розміщення реклами;

Розглянемо одну із функцій докладніше.

7.1.2 Сценарій виконання завдання прийняття замовлення

Обрана процедура виконання функції прийняття замовлення на друк поліграфічної продукції цікава тим, що є процесом з ітераційним уточненням і містить у собі різні види взаємодії між функціональними об'єктами підрозділу. Схема цієї процедури подана на Рис. 7.2. Зовнішнім впливом, яке ініціалізує виконання функції прийняття замовлення є надходження заявки і специфікації замовлення від клієнта. Функціональним об'єктом, якому доручено приймати такий вплив на підрозділ, є директор центра. Процедура, що її виконує директор центру, є досить стандартною послідовністю дія по узгодженню різних параметрів контракту. Цікавіша буде не сама послідовність дій, а різні види взаємодій між функціональними елементами підрозділу. Так, крок **2** даної процедури припускає розподілення впливу **«проаналізувати заявку на відповідність можливостям виробництва»** одночасно на декілька

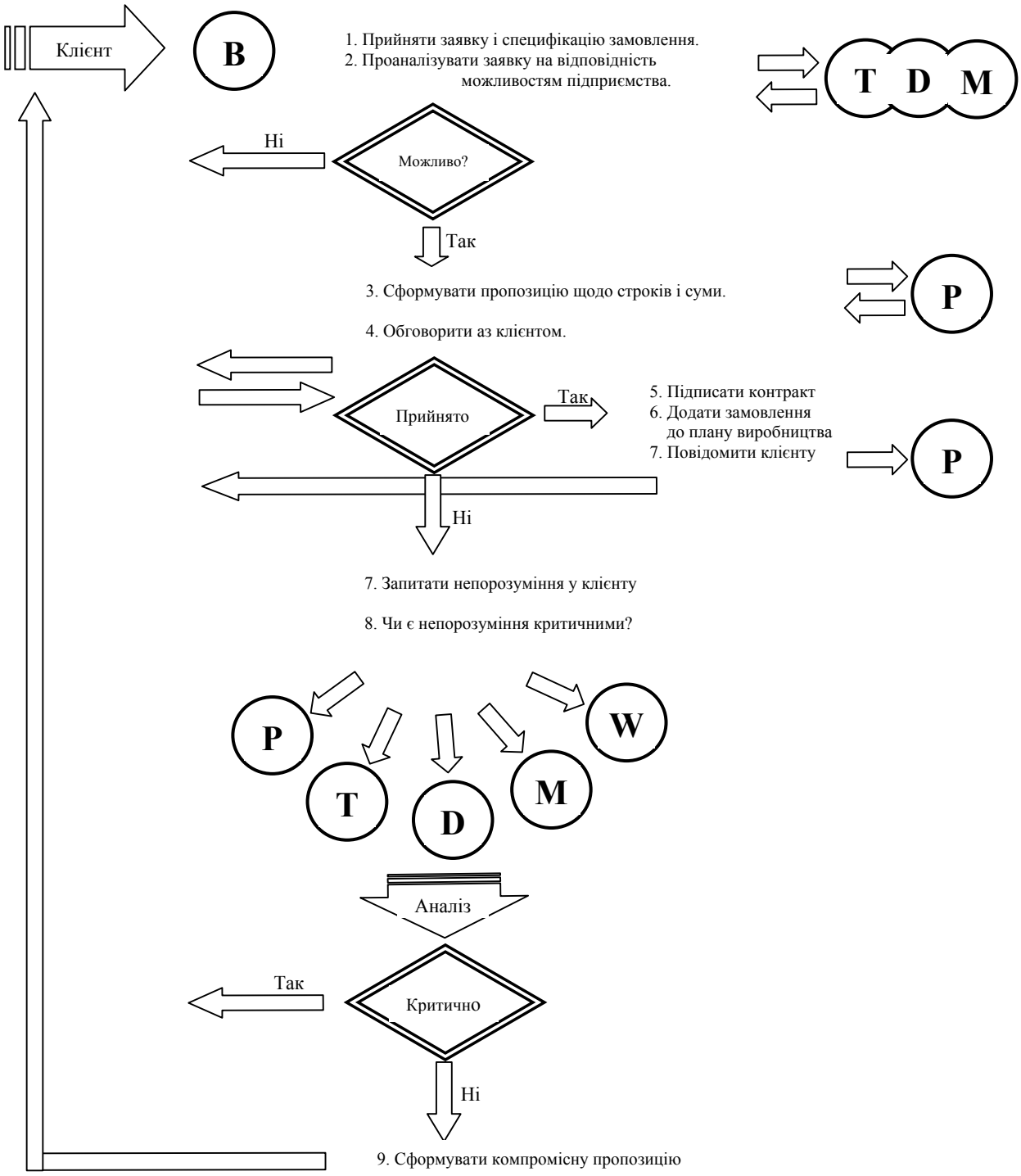


Рисунок 7.2 – Схема процедури прийняття замовлення.

функціональних елементів: **T** (набірник), **D** (дизайнер), **M** (інженер верстки), два з яких є групами (**T** і **M**), з наступним об'єднанням і оптимізаційним аналізом їх відповідей для отримання найбільш песимістичної оцінки. Крок **3** процедури припускає вплив **«сформувати пропозицію щодо строків виконання і вартості контракту»** на елемент **P** (заступник по виробництву) і

отримання його реакції у вигляді такої **«пропозиції»**. Крок **6** припускає вплив **«додати замовлення до плану виробництва»** на елемент **P** без чекання його реакції. Крок **8** одночасно розповсюджує вплив **«Чи є критичними зауваження?»** на декілька елементів (**P**, **T**, **D**, **M**, **W**) з метою наступної селекції із набору реакцій найбільш оптимістичного результату.

Вплив на кроці **8** є спробою керівника видавничого центру з'ясувати, на якій ділянці підрозділу отримані від клієнта обмеження є найбільш критичними. Метою елементу **B** в даному випадку є – визначити можливість формування конструктивної реакції підрозділу в цілому на зовнішній вплив, що іде від клієнта. Припустимо, обмеження, які клієнт накладає на кроці **7**, сформульовані таким чином:

- строк виконання замовлення не повинен перевищувати двох тижнів, починаючи з завтрашнього дня.;
- оптичне разрешение при друку не може бути нижче за 600 dpi;
- вартість одного екзмепляра не повинна перевищувати $1.25

Сценарій виконання крока **8** може виглядати таким чином:

Елемент **B**:

1. **Сформувати вплив** (Оцінити можливість виконання замовлення за шкалою: 0 - неможливо; 0.25 – можливо, але за рахунок призупинення виконання інших робіт; 0.5 – можливо, але необхідно сплатити сверхурочну роботу, придбати матеріали; 0.75 – можливо, але потрібує уточнення і неістотної корекції плана роботи; 1 – безумовно можливо. Параметри замовлення: специфікація замовлення у файлі SPEXXX.XLS, обмеження замовника (строк виконання замовлення не може перевищувати двох тижнів, починаючи з завтра, оптичне разрешение при друку не повинно бути менше за 600 dpi, вартість одного екземпляра не повинна перевищувати $1.25)
2. **Вплинути** на елементи **P**, **T**, **D**, **M**, **W**; чекати результатів.

Елементи **P**, **T**, **D**, **M**, **W**:

1. Отримати вплив.
2. Визначити, чи входить реакція на цій вплив до сфери власних повноважень.
3. Визначити параметри впливу, які відносяться до сфери власних повноважень (для елемента **P** такими параметрами будуть: специфікація замовлення у файлі SPEXXX.XLS, обмеження замовника (строк виконання замовлення не може перевищувати двох тижнів, починаючи з завтра, оптичне разрешение при друку не повинно бути менше за 600 dpi, вартість одного екземпляра не повинна перевищувати $1.25; для елемента **T**: специфікація замовлення у файлі SPEXXX.XLS, обмеження замовника (строк виконання замовлення не може перевищувати двох тижнів, починаючи з завтра; і т. д.).
4. Визначити, які результати із запитаних цей елемент може сформувати (Для даного впливу запитаний результат – це число, яке приймає значення із множини {0, 0.25, 0.5, 0.75, 1}, вказуючи, який/які із параметрів впливають на сформоване значення. Елемент, який не має відношення до виконання даного замовлення – таким елементом в даному випадку може бути **W** – повинен, дати відповідь 1 – безумовно можливо). Можливі приклади відповідей:
   - елемент **P** - 0.25 (строк виконання, вартість);
   - елемент **M** - 0.5 (строк виконання, специфікація замовлення).
5. Сформувати результат і передати його елементу **B**.

Елемент **B**:

3. **Проаналізувати отримані результати**: визначити інтегральну песимістичну оцінку і критичні параметри, які вплинули на формування цієї оцінки. Для наведених прикладів відповідей елементів **P** і **M** інтегральною оцінкою, вочевидь, буде 0,25 (строк виконання, вартість). ОТриманий результат, таким чином, вірогідніше всього, приведе або до подальшого

ітераційного уточнення з клієнтом критичних параметрів, або до відмови від підписання контракту.

7.1.3 Агенти, їх ролі і політики в моделі розглянутої процедури

Модель розглянутої процедури будується з використанням відомих підходів, що базуються на поняттях інтелектуального агента, ролі, політики і взаємодії (кооперації). Кожний із агентів у розглянутому прикладі фактично виконує роль персонального асистента того функціонального елемента (мастера), реакції якого він представляє. Так агент **B** моделює в даній процедурі поведінку директора видавничого центру, агенти **P**, **T**, **D**, **M**, **W** – поведінку його підлеглих. Аналіз приклада демонструє, що реакція агента на детермінований вплив може залежати від стану агента. Дійсно. Відповіді агентів **P**, **T**, **D**, **M**, **W** на кроці **8** залежать від їх завантаженості іншою роботою (за іншими контрактами), наявності або відсутності необхідних матеріалів, програмного забезпечення, технологічних обмежень. Легко побачити, що реакція агента на вплив може привести його в інший стан. Стан агента, в свою чергу, може накладати певні обмеження на впливи або параметри впливів, які приймаються агентами до виконання. Таким чином, при розробці моделі взаємодіючих агентів необхідно враховувати, що роль агента є функція від його стану, яка змінює стан. Також слід відмітити, що стан агента може накладати обмеження як на область значень вхідних параметрів, так і на результати, отримані на базі цих параметрів.

7.2 Моделювання процесу планування проекту

Одне із застосувань даного середовища моделювання - планування бізнес-процесів. Головна ідея близька до ідеї, використаної у проекті ADEPT [104]: моделювати бізнес-процеси як колекції автономних агентів, що вирішують проблеми, і здатних спілкуватися для моделювання виконання бізнес-процесу. Відповідний план бізнес-процесу може бути автоматично створено потім.

Розглянемо, як наше середовище можна використати для планування процесу розробки інформаційної системи. Будемо вважати, що організаційна структура, яка може виконати цей проект є така, як показано на Рис. 7.3. - Відділ ІС. Зробимо також такі припущення:

Відділ ІС, поданий на Рис. 7.3. є одним із кандидатів на виконання цього проекту. З точку зору змагання, відділ ІС, а також інші учасники вважаються агентами - учасниками, що представляють функціональні компоненти

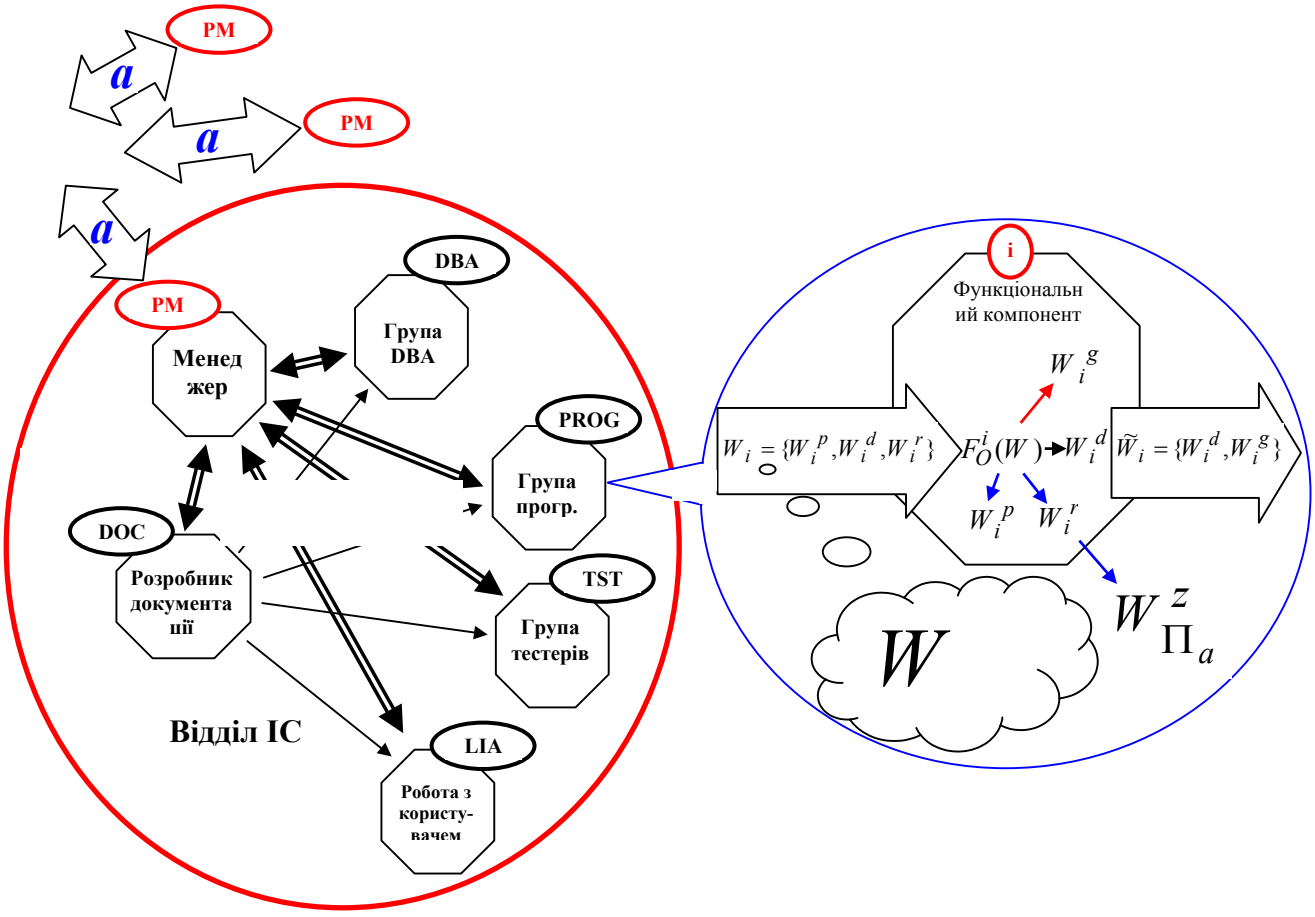


Рисунок 7.3 – Модель компоненти/функціональної системи [25].

суспільства більш високого рівня.

Суспільство агентів відділу ІС складається з таких функціональних компонентів: менеджер проекту (агент **PM**), група адміністратора баз даних (агент **DBA**), група програмістів (агент **PROG**), група створення документації (агент **DOC**), група тестерів (агент **TST**), група зв'язку із користувачами (агент **LIA**)

Якщо на деякому етапі виникає потреба розглянути один із функціональних компонентів детальніше, цей компонент можна промоделювати як функціональну систему – у вигляді суспільства агентів.

Процес починається в момент часу t_0 із зовнішнього впливу

$$W_a = \{w_0 = '\mathit{Propose\_IS\_Development\_Plan}', X_0, Y_0\},$$

з такими параметрами і описом результатів:

$$X_0 = \{\textbf{\textit{budget}} =< \mathit{figure} >, \textbf{\textit{duration}} =< \mathit{figure} >, \textbf{\textit{Proposal\_Template}} =< \mathit{file\_name} >,$$

$$\textbf{\textit{Proposal\_Descr}} =< \mathit{file\_name} >\};$$

$$Y_0 = \{\mathit{Possibility}(\textbf{\textit{budget}}, \textbf{\textit{duration}}), \mathit{Proposal}(\textbf{\textit{Proposal\_Template}})\}$$

що надходить до сенсорного входу агенту **PM. PM** виконує як найменш такі дії:

1. **Перевірка вхідного впливу.** На цьому етапі **завдання PM** - перевірити, чи узгоджується вхідний вплив, його параметри і опис результатів із роллю **PM** і його поточним станом. Далі поведінка **PM** (тобто, його макромодельна програма), може бути одна з двох: вхідний вплив приймається, або відхиляється. Не будемо розглядати ситуацію, коли зовнішній вплив відхиляється. Будемо надалі вважати, що всі вхідні впливи приймаються і виконуються відповідні макромодельні програми.

2. **Виконання роботи і зміна стану агента.** На цьому етапі макромодель **PM** $F_O^{PM}(W_a)$ (відповідно із онтологією планування програмного проекту, доступ до якою забезпечує агент онтології) розкладає вхідну роботу, тобто

генерує множину робіт, що відповідають етапам проекту, згідно із правилом (1.1а), де:

$$W_{PM}^{p} = \{w_0 = '\mathit{Propose\_IS\_Development\_Plan}', X_0, Y_0\}\ ,\ W_{PM}^{d} = \varnothing,\ W_{PM}^{r} = \varnothing,$$

$$W_{PM}^{g} = \{ w_1 = (\mathit{'Assemble\ Project\ Proposal'}, X_1, Y_1),$$

$$w_2 = (\mathit{'Choose\ best\ DB\ Schemata\ Plan\ Bid'}, X_2, Y_2),$$

$$w_3 = (\mathit{'Choose\ best\ Software\ Model\ Plan\ Bid'}, X_3, Y_3),$$

$$w_4 = (\mathit{'Choose\ best\ REQ\ Analyses\ Plan\ Bid'}, X_4, Y_4),$$

$$w_5 = (\mathit{'Analyse\ requirements'}, X_5, Y_5),$$

$$w_6 = (\mathit{'Design\ database\ schemata'}, X_6, Y_6),$$

$$w_7 = (\mathit{'Design\ software\ model'}, X_7, Y_7),$$

$$w_8 = (\mathit{'Program\ the\ software'}, X_8, Y_8),$$

.....

$$w_{15} = (\mathit{'Perform\ customers\ training\ '}, X_{15}, Y_{15})\}.$$

На цьому етапі виконання макромодельної програми зміна стану агента **PM** від s_0 до s_1 шляхом додання додаткових обмежень може бути зроблена, якщо це відповідає правилу поведінки, яке записано у відповідній онтології, але вбудоване в макромодельну програму.

3. Генерація компонентів матриці $\mathbf{K}_{PM}(t_0)$. На цьому етапі макромодель **PM** генерує $\mathbf{K}_{PM}(t_0)$, як показано на Рис. 7.4.

Матрицю $\mathbf{K}_{PM}(t_0)$ можна інтерпретувати так: на наступному етапі моделювання , в момент часу $t_0 + \Delta t$ роботи w_1, w_2, w_3, w_4 виконуються; роботи w_5, w_6, w_7 передаються декільком функціональним компонентам: w_5 - агентам **DBA** і **LIA**, w_6 - агентам **DBA**, **PROG** і **LIA**, w_7 - агентам **DBA**, **PROG** і **LIA**. Ціллю такої передачі може бути наприклад те, що агент **PM** точно не знає, у чому полягаю обов'язки **DBA**, **PROG**, **LIA**, і хоче отримати деякий досвід у цьому для використання у майбутньому. Однак, ми не вважаємо, що

$$\mathbf{K}_{PM} = \begin{array}{c|cccccccc|c} & w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 & w_8 & \dots & w_{15} \\ \hline \textbf{DBA} & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & & 0 \\ \textbf{PROG} & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & & 0 \\ \textbf{PM} & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \dots & 0 \\ \textbf{DOC} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & 0 \\ \textbf{TST} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & 0 \\ \textbf{LIA} & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & & 1 \end{array}$$

Рисунок 7.4 – Наміри і можливості **PM** до завдання $W_a = \{'developIS', X, Y\}$ в $]t_0, t_0 + \Delta t]$.

$$\Omega_a = \begin{array}{c|cccccccc|c} & & & & & & & & & & \\ \hline \Theta_{DBA} & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & & 0 \\ \Theta_{PROG} & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & & 0 \\ \Theta_{PM} & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \dots & 0 \\ \Theta_{DOC} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & & 0 \\ \Theta_{TST} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & 0 \\ \Theta_{LIA} & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & & 1 \end{array}$$

Рисунок 7.5 – Стан системи (відносно до W_a) в момент часу $t_0 + \Delta t$.

агент **PM** може передати роботи декільком виконавцям паралельно, щоб кожний із виконавців виконав частину роботи, тому що це суперечить принципу атомарності роботи. Роботи w_5, w_{15} були назначені явно агентам **PROG** і **LIA**.

Матриця станів системи $\Omega_a(t_0 + \Delta t)$, пов'язана із процесом $W_a = \{'Propose\_IS\_Development\_Plan', X, Y\}$ буде виглядати як на Мал.7.5, оскільки всі агенти, крім **PM** відбувають у «ледачому» стані у момент часу $t_0 + \Delta t$.

Розглянемо діяльності агентів **PM, DBA, LIA** на етапі $t_1 = t_0 + \Delta t$.

PM приймає множину робіт:

$$W_{PM} = \{ w_1 = ('Assemble\ project\ proposal', X_1, Y_1),$$
$$w_2 = ('Choose\ best\ DB\ Schemata\ Plan\ Bid', X_2, Y_2),\},$$
$$w_3 = ('Choose\ best\ Software\ Model\ Plan\ Bid', X_3, Y_3),$$
$$w_4 = ('Choose\ best\ REQ\ Analyses\ Plan\ Bid', X_4, Y_4)\}$$

із параметрами і описом результатів для роботи w_1:

$$X_1 = \{\textbf{\textit{budget}} =< figure >, \textbf{\textit{duration}} =< figure >,$$

Proposal_Template =< *file _ name* >,

Proposal_Descr =< *file _ name* >,

$\tilde{Y}_2, \tilde{Y}_3, \tilde{Y}_4, \tilde{Y}_8, ..., \tilde{Y}_{15}\}$,

$Y_1 = \{Possibility(\textbf{budget}, \textbf{duration}), Proposal(\textbf{Proposal\_Template})\}$

Оскільки в момент t_1 параметри $\tilde{Y}_2, \tilde{Y}_3, \tilde{Y}_4, \tilde{Y}_8, ..., \tilde{Y}_{15}$ ще не містять коректних значень, скінчено-станова машина F_X агента **PM** переходить у стан відхилення і класифікує роботу w_1 як приналежну до множини робіт W_{PM}^d, що передаються. Таким чином, робота w_1 реально відкладається як мінімум на період Δt макромодель **PM** додає 1 до елементу d_1 вектора затримок процесу D_a. Робота w_1 знову назначається агентові **PM** для виконання на наступному етапі моделювання t_2. Це саме повторюється для робіт w_2, w_3, w_4.

DBA приймає набір робіт $W_{DBA} = \{ w_5 = (\textit{'Analyse requirements'}, X_5, Y_5), w_6 = (\textit{'Design database schemata'}, X_6, Y_6), w_7 = (\textit{'Design software model'}, X_7, Y_7)\}$ з відповідними параметрами і описом результатів:

$X_5 = \{ x_5^1 = (\textbf{budget} = <b>), x_5^2 = (\textbf{duration} = <d>),$

$x_5^3 = (\textbf{Plan\_Template} = <file\_name>), x_5^4 = (\textbf{Step\_Descr} = <file\_name>) \},$

$Y_5 = \{ y_5^1 = (Possibility(\textbf{budget}, \textbf{duration})), y_5^2 = (Plan\_File\_Name(\textbf{Plan\_Template})) \};$

$X_6 = \{ x_6^1 = (\textbf{budget} = <b>), x_6^2 = (\textbf{duration} = <d>),$

$x_6^3 = (\textbf{Plan\_Template} = <file\_name>), x_6^4 = (\textbf{Step\_Descr} = <file\_name>) \},$

$Y_6 = \{ y_6^1 = (Possibility(\textbf{budget}, \textbf{duration})), y_6^2 = (Plan\_File\_Name(\textbf{Plan\_Template})) \};$

$X_7 = \{ x_7^1 = (\textbf{budget} = <b>), x_7^2 = (\textbf{duration} = <d>),$

$x_7^3 = (\textbf{Plan\_Template} = <file\_name>), x_7^4 = (\textbf{Step\_Descr} = <file\_name>) \},$

$Y_7 = \{ y_7^1 = (Possibility(\textbf{budget}, \textbf{duration})), y_7^2 = (Plan\_File\_Name(\textbf{Plan\_Template}))$

};

де *Possibility*(***budget***, ***duration***) - опис результату, зображений на Рис. 7.6; *Plan_File_Name* (***Plan_Template***) може бути або стрічкою «*DUMMY* », якщо план не було розроблено, або може бути стрічкою, що містить коректне ім'я файла, створене згідно із ***Plan_Template***.

	0.5*d*	0.8*d*	1.0*d*	1.5*d*	1.8*d*
0.5*b*	*p*	*p*	*p*	*p*	*p*
0.8*b*	*p*	*p*	*p*	*p*	*p*
1.0*b*	*p*	*p*	*p*	*p*	*p*
1.5*b*	*p*	*p*	*p*	*p*	*p*
1.8*b*	*p*	*p*	*p*	*p*	*p*

Possibility: значення $p \in [0,1]$

Рисунок 7.6 - Приклад опису результатів.

Обробка політики *'Analyse requirements'* скінченно-становою машиною F_A агенту **DBA**, веде до відхилення цієї політики, оскільки в нашому прикладі було прийнято, що ця політика не виконується агентом **DBA**. Виконавець політики відхилення генерує вектор результатів $\tilde{Y}_5 = \{\mathbf{O}, "DUMMY"\}$. Та же сама реакція буде проведена при обробці роботи *'Design software model'*: $\tilde{Y}_7 = \{\mathbf{O}, "DUMMY"\}$, де O - матриця, створена згідно правилу, що дано на Рис. 7.6, і містить нульові значення можливостей: $\mathbf{O}_{ij} = 0, i = 1,..,5, j = 1,...,5$. Робота *'Design database schemata'* виконується агентом **DBA**, і отриманий такий результат:

$$\tilde{Y}_6 = \{\begin{bmatrix} 0.1 & 0.1 & 0.4 & 0.6 & 0.35 \\ 0.3 & 0.35 & 0.4 & 0.8 & 0.35 \\ 0.3 & 1.0 & 1.0 & 0.8 & 0.35 \\ 0.35 & 1.0 & 1.0 & 0.85 & 0.35 \\ 0.4 & 1.0 & 1.0 & 0.9 & 0.35 \end{bmatrix}, "DBSCH\_PLAN.XLS"\} \quad (7.1)$$

Агент **LIA** приймає такий самий як у **DBA** набір робот $W_{LIA} = W_{LIA} = \{w_5 = ('Analyse\ requirements', X_5, Y_5),\ w_6 = ('Design\ database\ schemata', X_6, Y_6),\ w_7 = ('Design\ software\ model', X_7, Y_7)\}$. Виконується агентом LIA, і отриманий такий результат:

$$\Omega_a = \begin{matrix} \Theta_{DBA} \\ \Theta_{PROG} \\ \Theta_{PM} \\ \Theta_{DOC} \\ \Theta_{TST} \end{matrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \ldots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad W_{\Pi_a}^{z} = \begin{matrix} \text{DBA} \\ \text{PROG} \\ \text{PM} \\ \text{DOC} \\ \text{TST} \end{matrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \ldots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Рисунок 7.7 – Стан системи і невиконані роботи (W_a) в $t_1 + \Delta t$.

$$\Omega_a = \begin{matrix} \Theta_{DBA} \\ \Theta_{PROG} \\ \Theta_{PM} \\ \Theta_{DOC} \\ \Theta_{TST} \end{matrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \ldots & 0 \\ 0 & 0 & 0 & & & & & & \\ & & & & & & & & \end{bmatrix}$$

Рисунок 7.8 – Стан системи (відносно до W_a) в $t_2 + \Delta t$.

$$\Omega_a = \begin{matrix} \Theta_{DBA} \\ \Theta_{PROG} \\ \Theta_{PM} \\ \Theta_{DOC} \\ \Theta_{TST} \end{matrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \ldots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Рисунок 7.9 – Стан системи (відносно до W_a) в $t_3 + \Delta t$.

$$\tilde{Y}_5 = \{ \begin{bmatrix} 0.1 & 0.15 & 0.4 & 0.6 & 0.55 \\ 0.1 & 0.55 & 0.4 & 1.0 & 0.65 \\ 0.2 & 0.9 & 1.0 & 1.0 & 0.70 \\ 0.35 & 1.0 & 1.0 & 1.0 & 0.75 \\ 0.4 & 1.0 & 1.0 & 1.0 & 0.75 \end{bmatrix}, "REQ\_PLAN.XLS" \}, \qquad (7.2)$$

а роботи w_3, w_4 відхиляються як неналежні з результатами $\tilde{Y}_3 = \{\mathbf{O}, "DUMMY"\}$, $\tilde{Y}_4 = \{\mathbf{O}, "DUMMY"\}$. Матриця станів системи $\Omega_a(t_1 + \Delta t)$ і матриця відхилених робіт $W_{\Pi_a}^{z}(t_1 + \Delta t)$ мають вигляд такий, як на Рис. 7.7.

В момент часу t_2 агент **PM** отримує завдання:

$$W_{PM} = \{ w_1 = ('Assemble\ project\ proposal', X_1, Y_1),$$

$$w_2 = ('Choose\ best\ DB\ Schemata\ Plan\ Bid', X_2, Y_2), \},$$

$$w_3 = ('Choose\ best\ Software\ Model\ Plan\ Bid', X_3, Y_3),$$

$$w_4 = ('Choose\ best\ REQ\ Analyses\ Plan\ Bid', X_4, Y_4) \}$$

з

$X_2 = \{\widetilde{Y}_5^{DBA}, \widetilde{Y}_5^{LIA}\}$,

$Y_2 = \{ y_2^1 = (Possibility(\textbf{\textit{budget}}, \textbf{\textit{duration}})),\ y_2^2 = (Plan\_File\_Name(\textbf{\textit{Plan\_Template}}))\}$;

$X_3 = \{\widetilde{Y}_6^{DBA}, \widetilde{Y}_6^{PROG}, \widetilde{Y}_6^{LIA}\}$,

$Y_3 = \{ y_3^1 = (Possibility(\textbf{\textit{budget}}, \textbf{\textit{duration}})),$

$y_3^2 = (Plan\_File\_Name(\textbf{\textit{Plan\_Template}}))\}$;

$X_4 = \{\widetilde{Y}_7^{DBA}, \widetilde{Y}_7^{PROG}, \widetilde{Y}_7^{LIA}\}$,

$Y_4 = \{ y_4^1 = (Possibility(\textbf{\textit{budget}}, \textbf{\textit{duration}})),$

$y_4^2 = (Plan\_File\_Name(\textbf{\textit{Plan\_Template}}))\}$;

і в змозі виконати роботи w_2, w_3, w_4. Робота w_1 знову затримана і запланована на наступний момент моделювання. Тепер завдання для макромоделей агента **PM** w_2, w_3, w_4 - вибрати оптимальні плани із прийнятих заявок, згідно із запланованими відгуками *Possibility*. В нашому випадку завдання спрощене тим, що кожен агент виконує тільки одну роботу: LIA виконує w_2, **DBA** виконує w_3, і **PROG** виконує w_4. Результати цього етапу подані на Рис. 7.8. Зверніть увагу на те, що функціональні компоненті **DBA**, **PROG**, **LIA** є «ледачими» в інтервал часу $]t_2, t_2 + \Delta t]$, хоча вони здатні виконати ще деякі роботи.

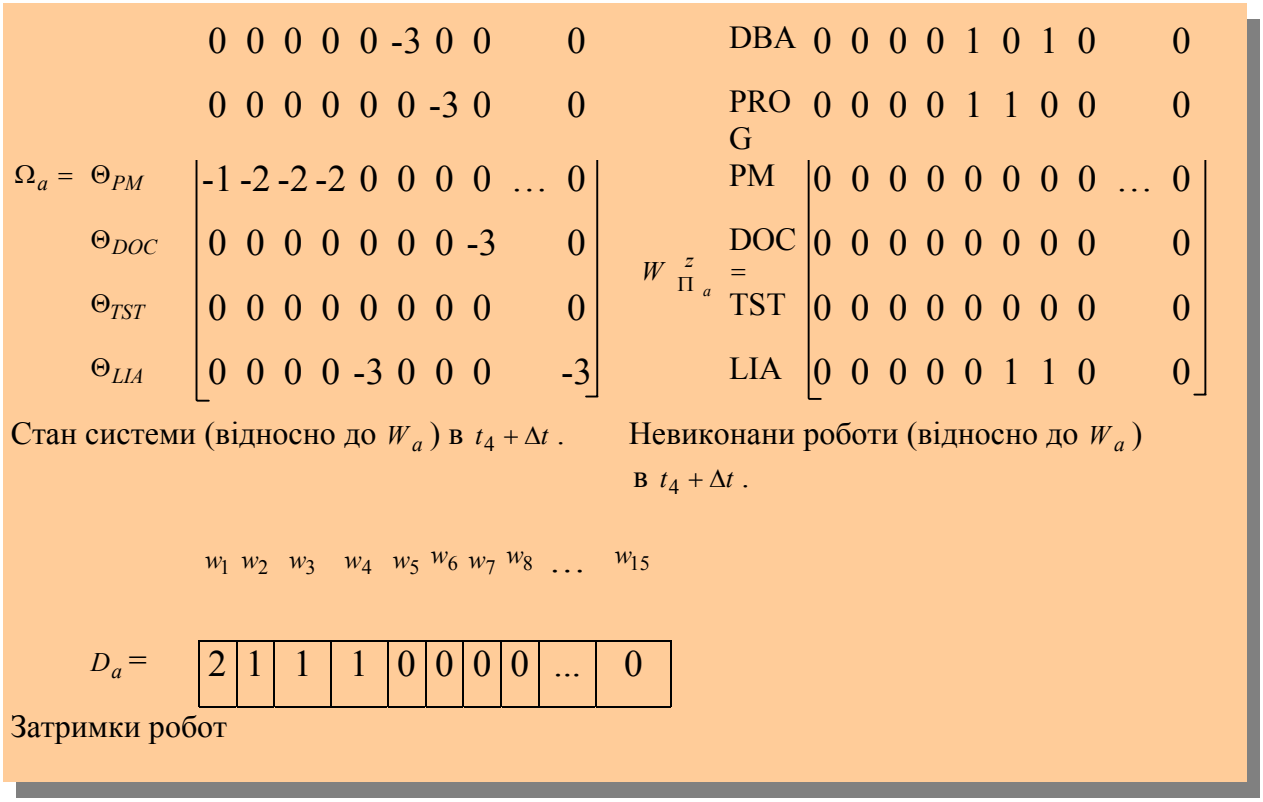


Рисунок 7.10 – Стан системи, невиконані роботи і затримки робот (відносно до W_a) після того, як W_a виконана.

Подібна оптимізація роботи w_1 виконується макромоделлю агента **PM** в інтервал часу $]t_3, t_3 + \Delta t]$. Ця робота полягає в тому, щоб об'єднати в один план

$$\tilde{y}_1^1 = \min(\tilde{y}_2^1, \tilde{y} \qquad \tilde{y}_{15}^1) = \begin{bmatrix} 0.1 & 15 & 0.4 & 0.6 & 0.3 \\ 0.1 & .5 & 0.4 & 0.75 & 0.35 \\ 0.2 & 75 & (1.0) & 0.8 & 0.35 \\ 0.3 & .9 & 1.0 & 0.8 & 0.35 \\ 0.35 & (.0) & 1.0 & 0.85 & 0.35 \end{bmatrix} \tag{7.3}$$

всі плани, запропоновані учасниками суспільства $\tilde{Y}_2, \tilde{Y}_3, \tilde{Y}_4, \tilde{Y}_8, ..., \tilde{Y}_{15}$. Стан системи представлено на Рис. 7.9. Результат – значення *Possibility* у найгіршому випадку буде таке:

Хоча завдання W_a в момент часу t_4 вже було виконане, потрібен ще один етап моделювання, щоб перевірити, чи всі агенти припинили абсорбувати вхідні роботи. Результати фінального етапу показані на Рис. 7.10.

Аналізуючи результати моделювання процесу планування, помітимо таке:

1. Параметричні відгуки для моделювання координації і торгівлі. Результуюче значення *Possibility* (7.3) свідчить про те, що, пропонований програмний проект може бути безумовно виконано з підвищеним бюджетом (1.8**b**) навіть швидше, ніж треба - 0.8d. У випадку, якщо бюджет проекту підвищувати не можна, проект може бути виконано за потрібний час, з бюджетом, який починається з 1.0**b**. Взагалі кажучи, параметричні відгуки, які забезпечуються нашим середовищем, роблять дуже простими моделювання протоколів координації (такої, як CNP або брокерство [27]), або аукціону через потоки робіт.

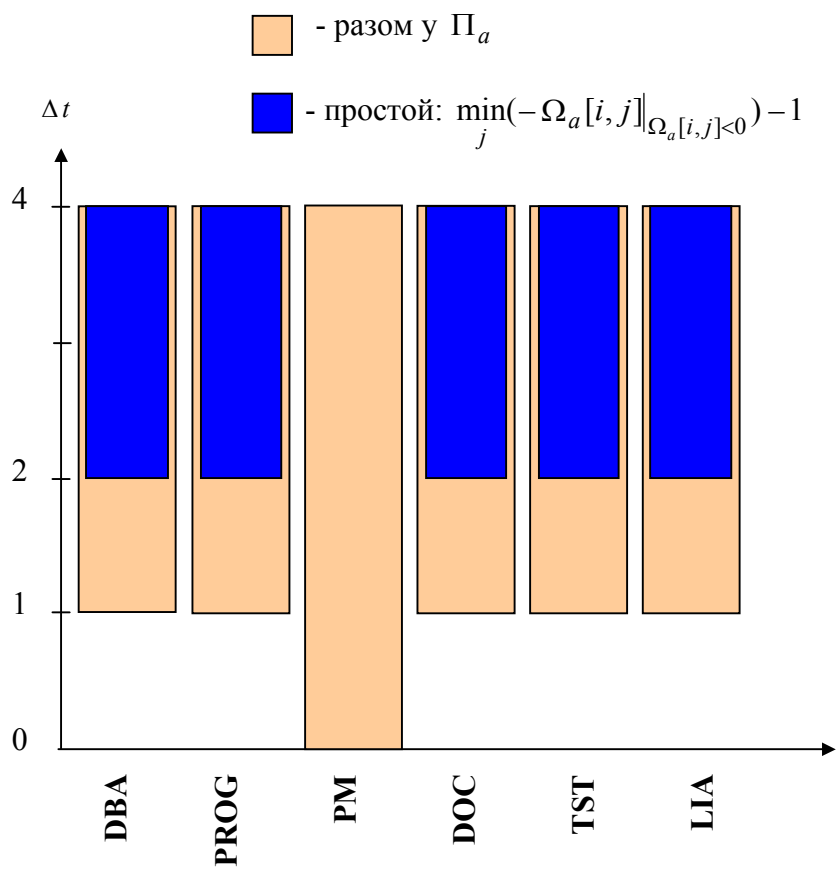


Рисунок 7.11 – Навантаження агентів в Π_a.

2. Навчання і еволюція поведінки. Аналіз $W^z_{\Pi_a}$ показав, що роботи w_5, w_6, w_7 були відхилені деякими агентами, хоча врешті-решт, всі роботи були виконані. Слідство таке: $W^z_{\Pi_a}$ може бути використано як приклад для більш акуратного призначення робіт в моделюванні процесу планування – агент **PM** вже вивчив можливості учасників суспільства, пов'язаних із виконанням загального завдання і адаптував свою поведінку. Тому ми будемо вважати роботу w_i такою, що не виконується суспільством агентів, пов'язаних до процесу Π_a тільки тоді, коли стовпчик i у $W^z_{\Pi_a}$ не містить нульових значень.

Task/Work	t_1	t_2	t_3	t_4
Tasks Specification	**PM**			
REQ Analyses Planning		**LIA**		
DB Schemata Design Planning		**DBA**		
Software Model Design Planning		**PROG**		
Programming Planning		**PROG**		
.....		...		
Customers' Training Planning		**LIA**		
REQ Analyses Plan Optimisation			**PM**	
Software Modelling Plan Optimisation			**PM**	
DB Schemata Design Plan Optimisation			**PM**	
Project Plan Assembly				**PM**

Рисунок 7.12 – Автоматично згенерований план процеса приготування пропозиції щодо проекту.

Однак, невідомо нічого про поведінку суспільства, якщо нові агенти – учасники з новими здатностями (наприклад, що частково перекривають роботи, розміщені в $W_{\Pi_a}^z$) приєднуються до команди. Поки що ми вважаємо, що з моменту початку процесу і до моменту його виконання контингент функціональних компонентів не змінюється.

3. Оцінка завантаженості агентів. Оцінка завантаженості агентів, пов'язаних із процесом Π_a - згідно з діаграмою, представленою на Рис. 7.11. – показує, що середня завантаженість агентів - 37.5 %.

4. Планування методом симуляції. Один із додаткових результатів, який може бути отримано при моделювання процесу планування за допомогою симуляції, є чорновий план створення проекту (див. Рис. 7.12).

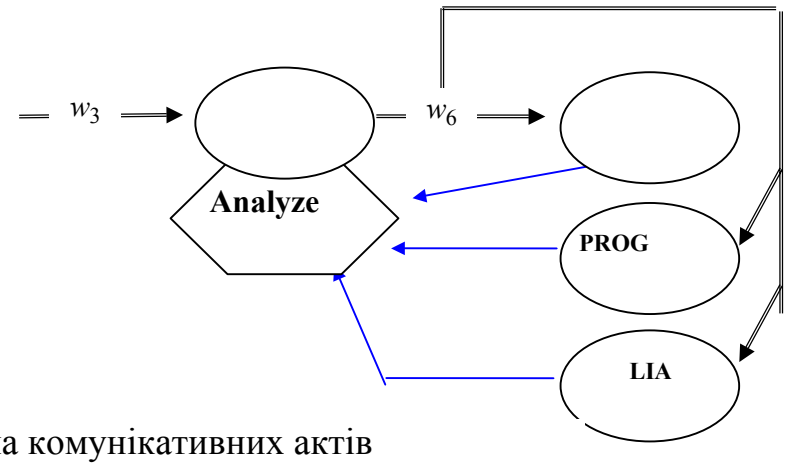
5. Акти комунікації як послідовності робіт. Акти комунікації між агентами, що виконують процес, моделюються як послідовності конкретних робіт, що виконуються у різні моменти часу. В нашому прикладі, виконання роботи w_3 було віднесено до (див. п.4.2) ***недетермінованих запитів із аналізом результатів*** w_6 до агентів **DBA**, **PROG** і **LIA** (мал.7.13 (а)), де аналіз результатів виконує агент **PM**. Наш приклад показав, що доцільно виконувати

роботи w_3 і w_6 у зворотному порядку, щоб зняти необхідність синхронізації вхідного впливу із відповідною реакцією. Як бачимо з мал.7.13 (б), ці роботи і були виконані у «інверсному порядку». Також слід відмітити механізм встановлення необхідних затримок для елементів, що знаходяться у «стеці» робіт. В нашому випадку, середовище використовує процедуру F_X верифікації вхідних параметрів для затримки виконання роботи w_3. Ця робота не буде виконана, доки в $\tilde{Y}_6^{DBA}, \tilde{Y}_6^{PROG}, \tilde{Y}_6^{LIA}$ не будуть розміщені коректні значення, тобто доки агенти **DBA**, **PROG** і **LIA** не завершать виконання роботи w_6, яку отримали від агента **PM** в момент часу $t_1 + \Delta t$ (див. Рис. 7.13b). У випадку, якщо роботи не мають параметрів, наше середовище забезпечує альтернативний механізм для створення послідовності - обмеження на стани агентів (див. Рис. 7.13). Процес зміни стану тоді – це використання деяких обмежень, що мають скінчений період життя, і мають такий вигляд:

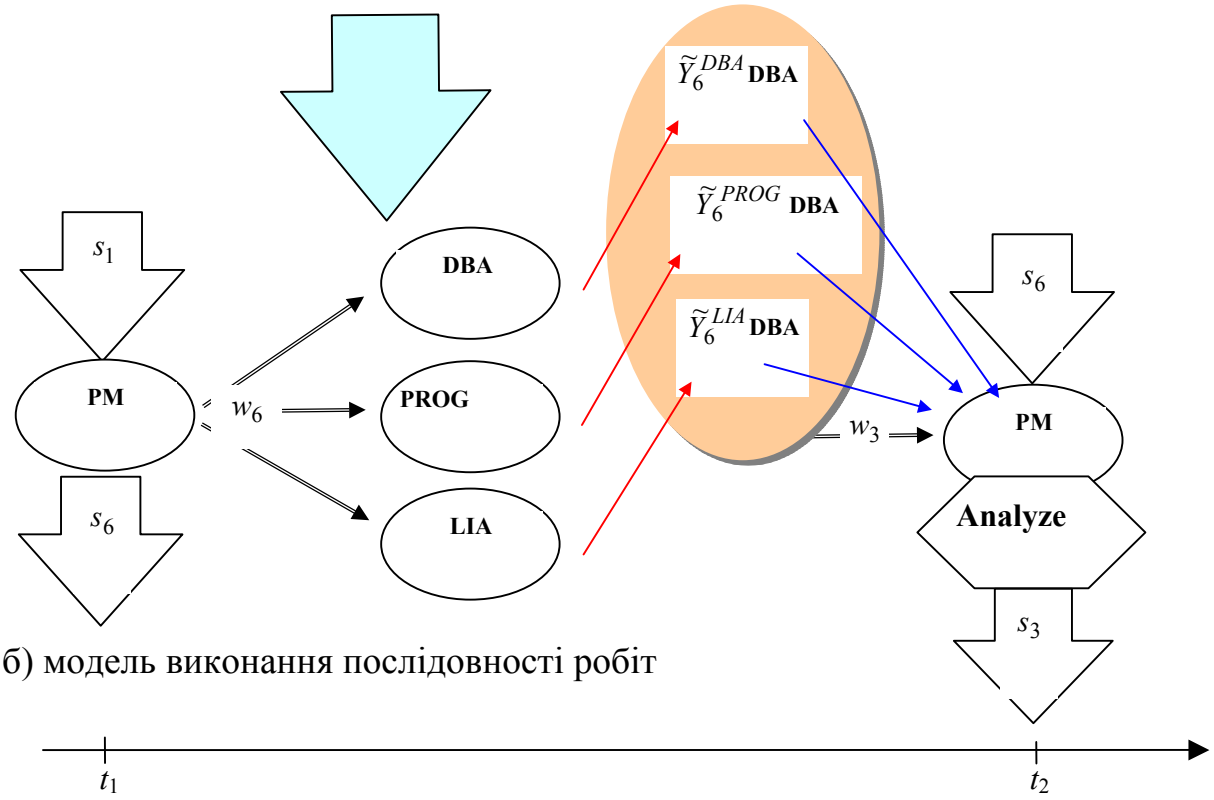
$$s_6 : \begin{cases} \ldots \\ \texttt{AddConstraint} = \text{'Reject } w_3\text{'}, \texttt{LifeTime} = 1 \\ \ldots \end{cases}, \tag{7.4}$$

якщо викликання виконавців роботи w_6 було комунікативним актом типу ***директива***.

6. Узгодженість із FIPA. Модель послідовності робіт, зображена на Мал.. 7.12, за нотацією подібна до Протоколу Мережі Контрактів FIPA [27]. Припущення були такі, що, по-перше, ми можемо моделювати комунікативні акти FIPA ACL за допомогою нашого підходу, а по-друге – ми можемо використати ACL як транспортну мову для комунікацій між агентами, в рамках суспільств, що будуть моделюватися в нашому середовищі.



а) діаграма комунікативних актів



б) модель виконання послідовності робіт

Рисунок 7.13 – Моделювання недетермінованого запиту із аналізом результатів.

6. **Підготовка роботи «ручним способом».** Навіть із розглянутого простого прикладу можна помітити, що на практиці треба виконати багато «ручної» роботи, до того, як функціональна системи буде готова до роботи. Макромоделі, онтології, обмеження на стан, описи параметрів, результатів, ролей повинні бути приготовлені заздалегідь і збережені у базі знань агента / суспільства. Щоправда, більшу частина такої роботи треба виконати один раз, і результат може потім бути використаний іншими агентами

/суспільствами (для різних завдань). Треба також помітити, що не всі атомарні роботи можна виконати за допомогою макромодельної програми. Агентів, що використовуються як персональні асистенти, забезпечують помітну допомогу у виконанні рутинної частини роботи. Наприклад, виконання роботи w_6 вимагає участі людини - спеціаліста, для виконання неформальної частини роботи по підготовці плану.

7.3 Моделювання процесу конкурсного відбору аспірантів

Для аналізу припустимості використання ЕІП к імплементації процедур дистанційного навчання і створення Віртуального Університету розглянемо одну із типових функцій -- процес конкурсного відбору аспірантів [36]. Головною причиною вибору саме цього прикладу було прийняття того факту, що ВУ повинен саморегулюватися для того, щоб бути успішним. Процеси управління ВУ потребують аналізу відгуків від процесів дистанційного навчання та викладання, для того, щоб виконати вимоги студентів. З другого боку, процедури доставки курсів і інших відомостей студентам, потребують зворотного зв’язку із блоками керування ВУ. Набір аспірантів можна розглянути як управлінську процедуру (наприклад, як набір співробітників). Нижче буде показано, що цей процес забезпечить нові відомості про курсі, які необхідно ввести,

Будемо вважати, що пошукачі (ті, що бажають стати аспірантами) можуть працювати з ЕІП, контактуючи із факультетами, які вони обирають, через агентів – посередників, і виражати свою потребу бути аспірантом.

Віртуальна кафедра: Заздалегідь зазначимо, що Віртуальна кафедра - це МАС, що складається принаймні з таких акторів (Рис. 7.14):

- Секретар – Агент-Посередник (**PA**)
- Професори (**PRA**), Асистенти (**AA**), розроблювачі курсів (**CMA**), Бібліотекар (**LA**) – агенти середнього рівня

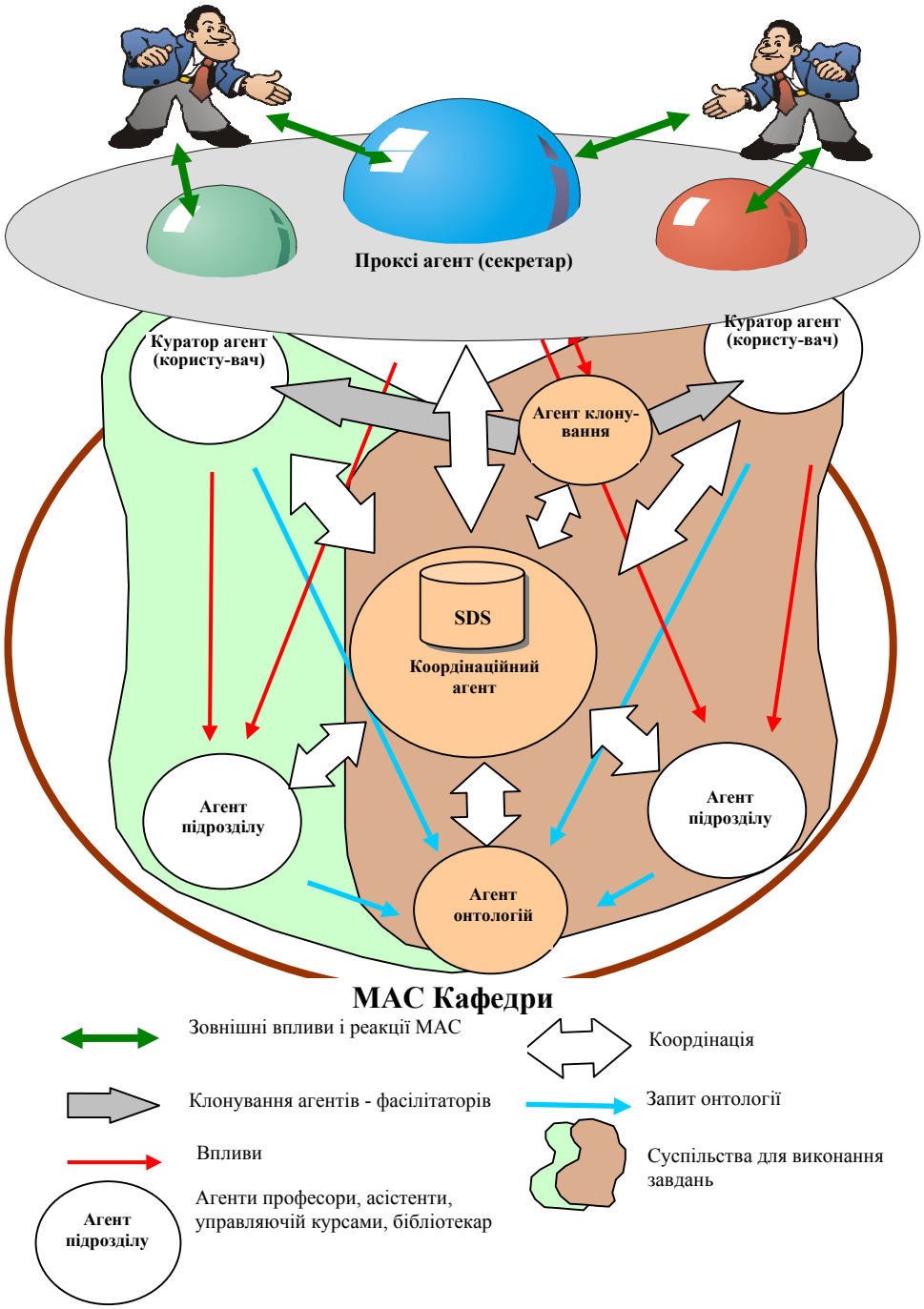


Рисунок 7.14 – Структура МАС віртуальної кафедри.

МАС факультету також містить сервісні блоки, що забезпечують масштабованість, координацію та спільне використання знань між функціональними акторами.: це – Агент Клонування (**СА**), Агент Координації (**СОА**) з його Простором Спільних Знань (SDS) і Агент Онтології (**ОА**).

Роль **СА** в даній задачі - клонувати Агентів Викладачів (**ТА**)кожний раз, коли агент-посередник “створює” нове завдання - опрацювати запит нового пошукача.

Сценарій прийому аспіранта: Сценарій прийому пошукача в аспірантуру було трохи адаптовано, щоб підкреслити переваги, які можна отримати, використовуючи підхід, поданий в розділах 4 і 5. Були зроблені такі припущення: усі учасники процесу прийому - пошукачі, професори, та ін. находяться на онлайновому зв'язку під час виконання всього сценарію, а також всі потрібні роботи виконуються за короткий час.

Ми вважаємо, що процедура прийому в аспірантуру складається із таких етапів:

- Пошукач надсилає своє резюме і підтверджує своє бажання навчатися у аспірантурі
- Резюме аналізується і знаходиться Професор, який більш за інших підходить цьому пошукачеві
- Пошукач складає тести, підготовлені обраним Професором
- Цей пошукач, що успішно склав тести, проходить інтерв'ю і приймається до роботи над дослідницьким проектом
- Професор і його асистент готують індивідуальний графік роботи аспіранта, а також, перелік літератури, що йому необхідно прочитати.

Діяльності агентів в цьому процесі можуть бути такими:

Етап 1. В*становити зв'язок і прийняти резюме.*:

PA приймає зовнішній вплив s генерує нове завдання. Перші атомарні роботи в рамках цього завдання такі: **CA** – клонувати Агента Викладача; **PA** – перенаправити зв'язок пошукача із факультетом на зв'язок із конкретним **TA**, **TA** – запросити резюме і виділити дані про кваліфікацію пошукача

Етап 2. *Аналіз резюме і пошук найкращого Професора - керівника*:

TA віддає дані про кваліфікацію пошукача **PRA факультету**. **PRA** відповідають, формуючи параметричний відгук, згідно з параметрами кваліфікації. **TA** знаходить найкращій результат, у тому випадку, якщо параметри відповідей професорів знаходяться в межах необхідних. В тому

випадку, якщо рівень кандидата не підходить, **TA** генерує завдання для Агента-Посередника відповісти пошукачеві і рекомендувати йому провести таку саму роботу на іншому факультеті. В тому випадку, якщо рівень кандидата є достатньо високим, кандидата кваліфікують і **TA** викликає новий етап – етап тестування.

Етап 3. *Тестування:*

TA запитує питання для тестів у **PRA**. **PRA** дає такі питання. **TA** пропонує пошукачеві відповісти на питання тесту і передає результати тестів **PRA**. **PRA** оцінює завдання і відповідає, ставлячи відмітки. **TA** проводить аналіз оцінок (подібно до етапу 2) і або кваліфікує пошукача і ініціює етап інтерв'ю, або просить **PA** повідомити пошукача про невдачу.

Етап 4. *Інтерв'ю:*

TA генерує завдання для **PRA** провести інтерв'ю з пошукачем. **TA** перенаправляє пошукача до **PRA**. **PRA** встановлює онлайновий зв'язок між своїм господарем (професором – людиною) і пошукачем. **PRA** вимагає від свого господаря – професора заповнити форму прийому в аспірантуру. **PRA** просить **TA** обробити цю форму із заповненими даними. **TA** аналізує форму і або передає її **PA** Відділу Кадрів для прийняття пошукача в аспірантуру, або просить **PA** повідомити пошукачу про невдалі результати інтерв'ю. Якщо пошукач успішно пройшов етап інтерв'ю, і тому став аспірантом, **TA** запускає етап створення індивідуальної програми.

Етап 5. *Розробка індивідуальної програми:*

TA генерує завдання для **PRA** приготувати індивідуальну програми роботи аспіранта на 1-ій семестр. **PRA** перенаправляє цей завдання своєму помічнику **AA**, щоб той добавив рекомендації щодо курсів у список параметрів. **AA** готує план роботи і запитує необхідні електронні курси у **CMA**. **CMA** аналізує запит, і, якщо треба, робить запит про додаткові, але ще недоступні, курси – приклад див. у [34].

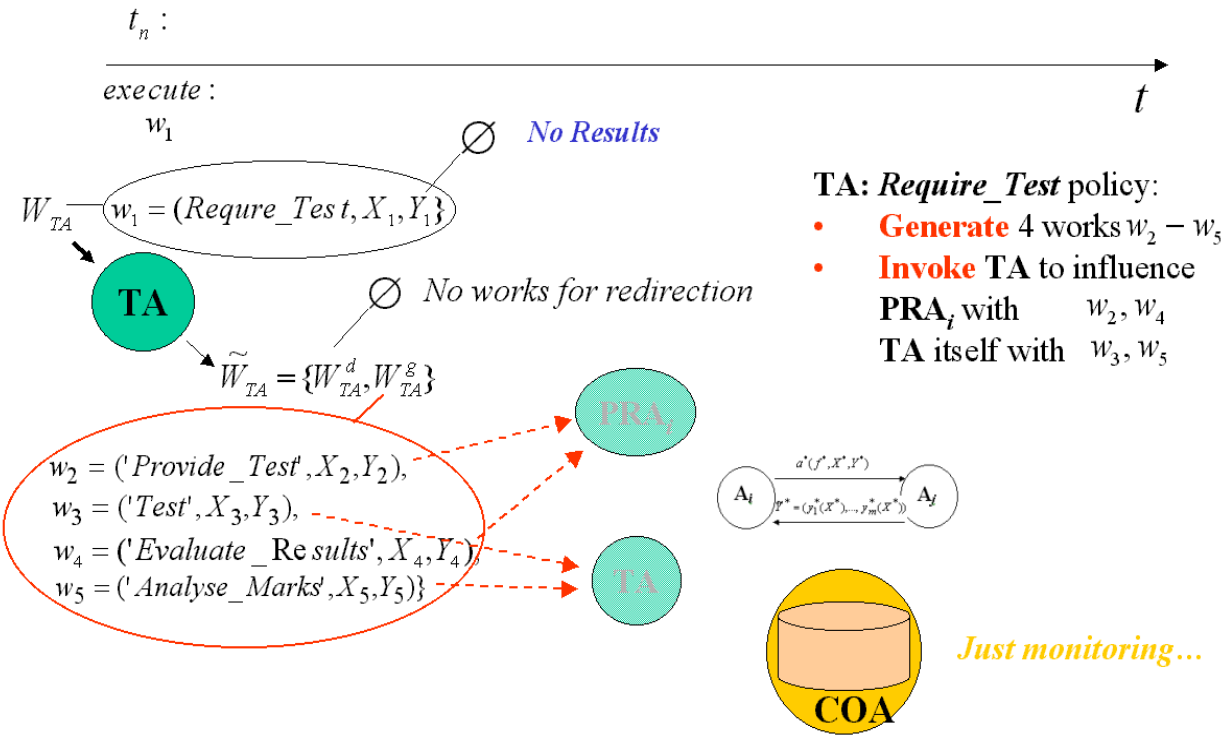


Рисунок 7.15 – Тестування, фаза 1, запит тесту (приведено по [54])

Моделювання етапу тестування: Нехай в момент $t = t_n$ **TA** ініціалізує етап тестування $t = t_n$. Розглянемо діяльності агентів **TA**, **PRA** і **PA** на цьому етапі.

В момент $t = t_n$ (див. Рис. 7.15) **TA** приймає таку множину робіт

$W_{TA} = \{ w_1 = ('Require\ the\ test', X_1, Y_1)\}$

з такими параметрами і результатами роботи w_1:

$X_1 = \{$***Edu_Rating*** $=<$ *structure*, **ontology** $=$ *Edu_Rating* $>$,
Q_E_Rating $=<$ *structure*, **ontology** $=$ *Qualif_Exp_Rating* $>$,
Pub_Rating $=<$ *structure*, **ontology** $=$ *Publication_Rating* $>$,
Professor $=<$ *Id*, **ontology** $=$ *Agent_Name* $>\}$,

$Y_1 = \varnothing$.

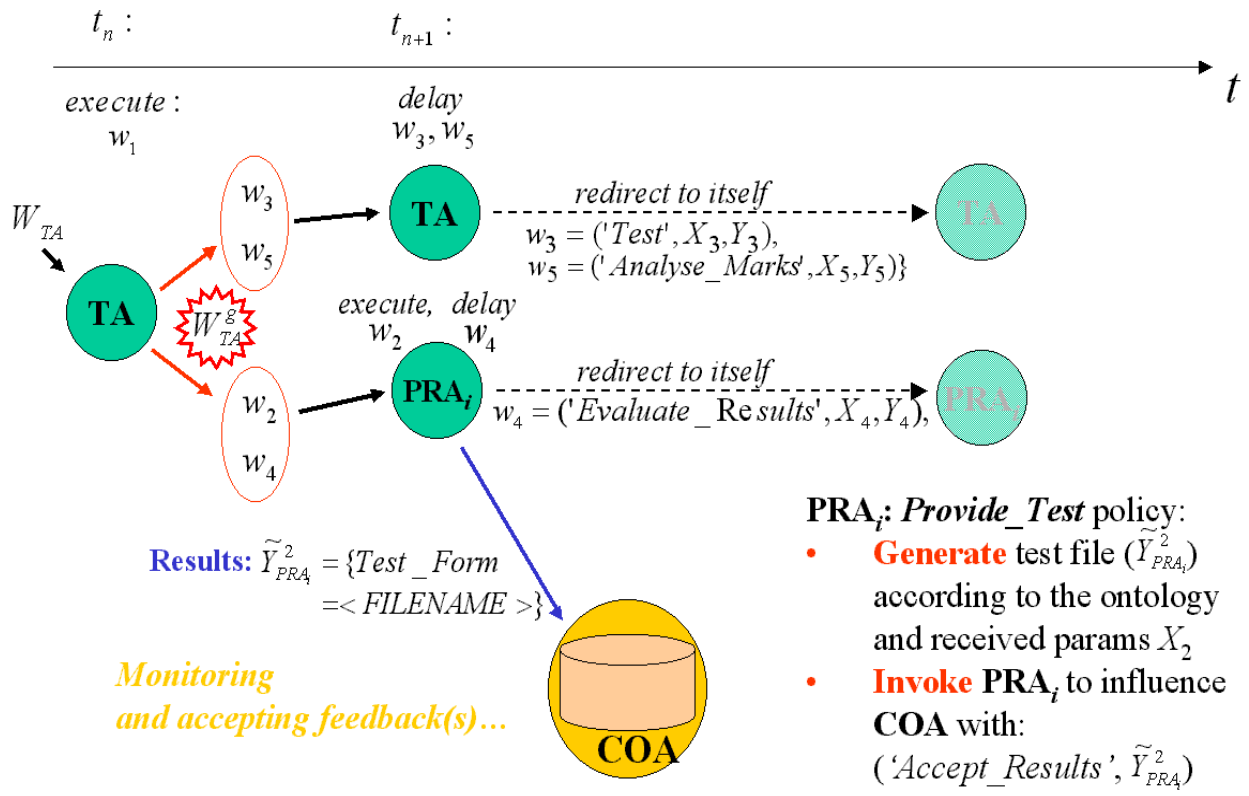


Рисунок 7.16 – Тестування, фаза 2, представлення тесту по [54].

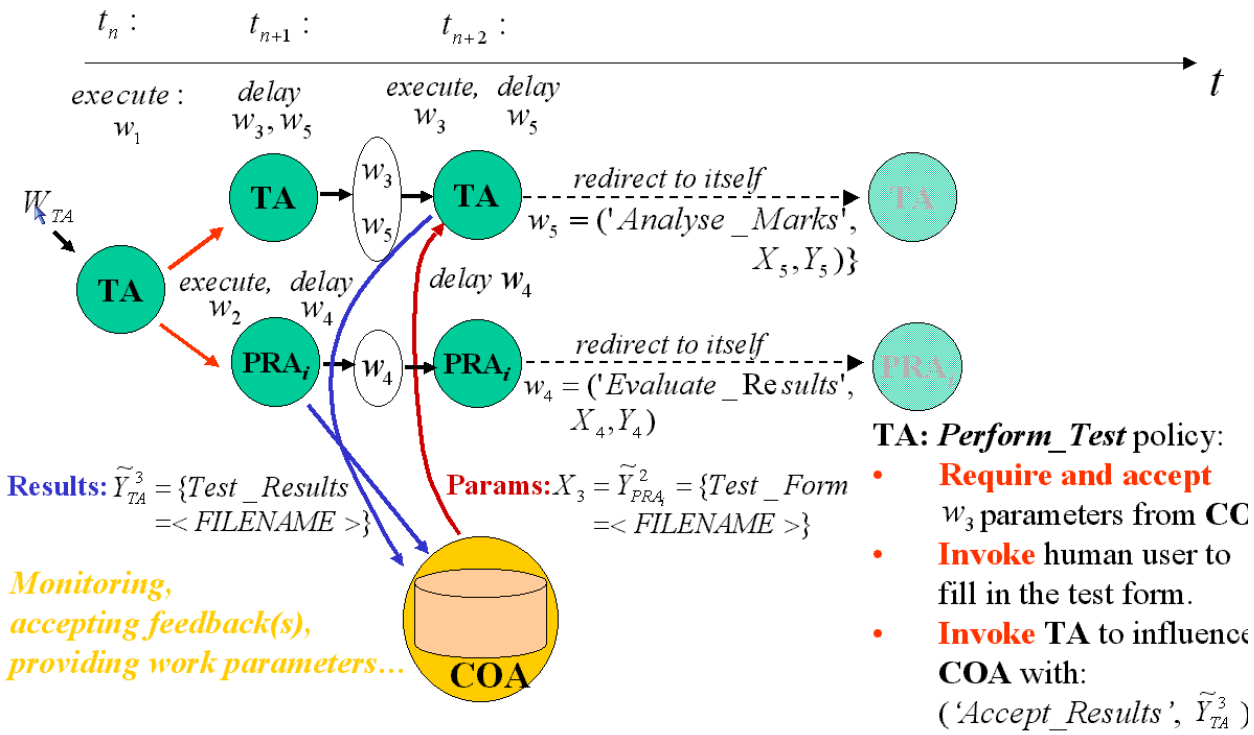


Рисунок 7.17 – Тестування, фаза 3, виконання тесту по [54].

Атомарна робота w_1 приймається і виконується, оскільки всі параметри X_1 (отримані як результати виконання робіт на попередніх етапах) є

доступними через SDS **агента COA**. В процесі виконання роботи w_1 агент **TA** генерує завдання $\widetilde{W}_{TA} = \{W_{TA}^d, W_{TA}^g\}$, де:

$W_{TA}^d = \varnothing$ тому що робота w_1 вже виконається і ніяких інших робіт не залишилося для передачі їх іншим агентам;

$$W_{TA}^g = \{ w_2 = ('Provide\_Test', X_2, Y_2),$$

$$w_3 = ('Test', X_3, Y_3),$$

$$w_4 = ('Evaluate\_\mathrm{Re}sults', X_4, Y_4),$$

$$w_5 = ('Analyse\_Marks', X_5, Y_5)\}.$$

Роботи w_2, w_4 сформують завдання W_{PRA} , а роботи w_3, w_5 сформують завдання W_{TA} в наступний момент часу $t = t_{n+1}$.

В момент $t = t_{n+1}$ (див. Рис. 7.16) **PRA** приймає

$$W_{PRA} = \{ w_2 = ('Provide\_test', X_2, Y_2),$$

$$w_4 = ('Evaluate\_Results', X_4, Y_4)\}.$$

Робота w_2 виконується і результати

$$\widetilde{Y}_2 = \{Test\_Form =< FILENAME >\}$$

передаються **COA** для подальшого використання. Тим часом, робота w_4 відправляється **PRA** на наступний момент часу - результати роботи w_3, які формують параметри роботи w_4, ще не повернулися від **COA**. Тим часом, **TA** приймає

$$W_{TA} = \{ w_3 = ('Test', X_3, Y_3),$$

$$w_5 = ('Analyse\_Marks', X_5, Y_5)\}$$

і перенаправляє обидві роботи собі на наступні проміжки часу, чекаючи на результати робіт w_2, w_4. В момент $t = t_{n+2}$ (див. Рис. 7.17) **TA** виконує w_3. В момент $t = t_{n+3}$ **PRA** виконує w_4 і передає вектор результату $\widetilde{Y}_{PRA} = \{\widetilde{y}_4^1 = (m_1,$

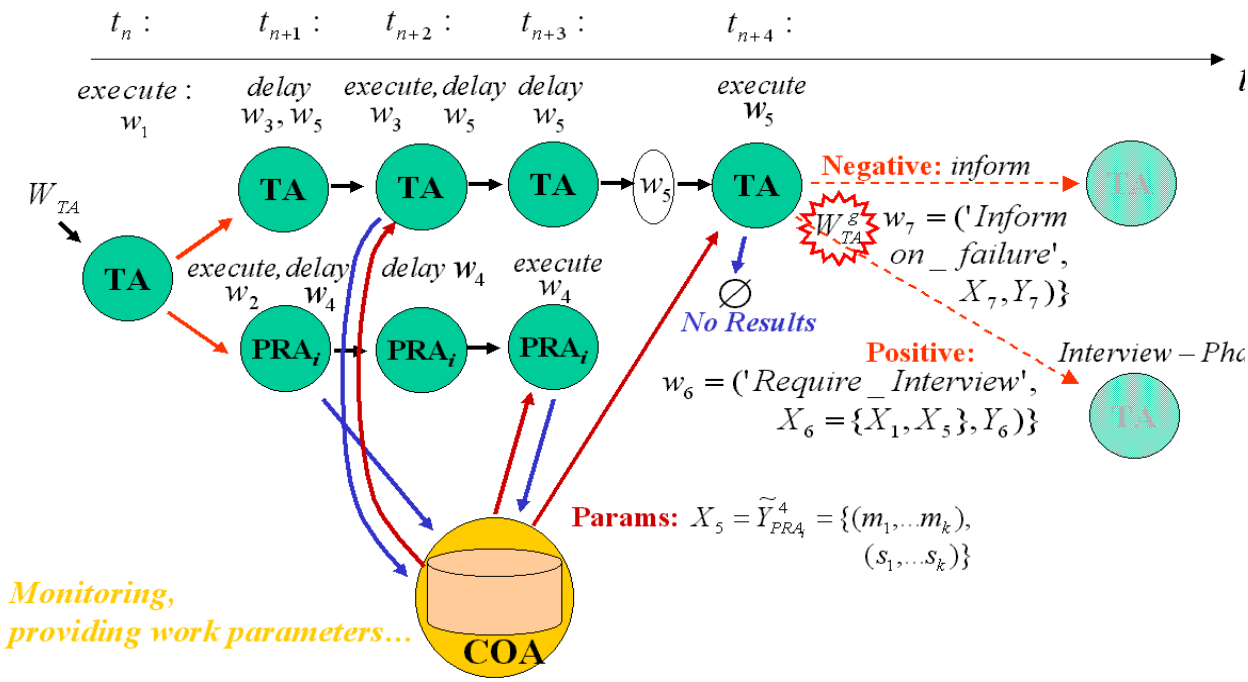


Рисунок 7.18 – Тестування, фаза 5 - остання, аналіз параметризованих оцінок (приведено по [54]).

$m_2,...,m_k), \tilde{y}^2_4 = (s_1, s_2,...,s_k)\}$ агенту **COA**. k - це кількість проектів, якими керує господар агента **PRA**, де потрібні аспіранти, m_i - це оцінка пошукача, якщо він претендує на роботу над проектом i, та s_i відображає найнижчу (за думкою професора) оцінку необхідного рівня пошукача.

В момент $t = t_{n+4}$ (див. Рис. 7.18) **TA** приймає

$$W_{TA} = \{w_5 = ('Analyse\_Marks', X_5, Y_5)\},$$

де

$$X_5 = \{\textbf{Marks} =< \tilde{y}^1_4, \textbf{ontology} = Mark\_per\_Project >,$$

$$\textbf{Scale} =< \tilde{y}^2_4, \textbf{ontology} = Positive\_Mark\_per\_Project >\}$$

і вирішує, чи може даний пошукач отримати місце в тому чи іншому проекті. Якщо це можливо, W^g_{TA} буде складатися із роботи $w_6 = ('Require\_the\_Interview', X_6 = \{X_1, X_5\}, Y_6)$,

в інших випадках **TA** генерує роботу.

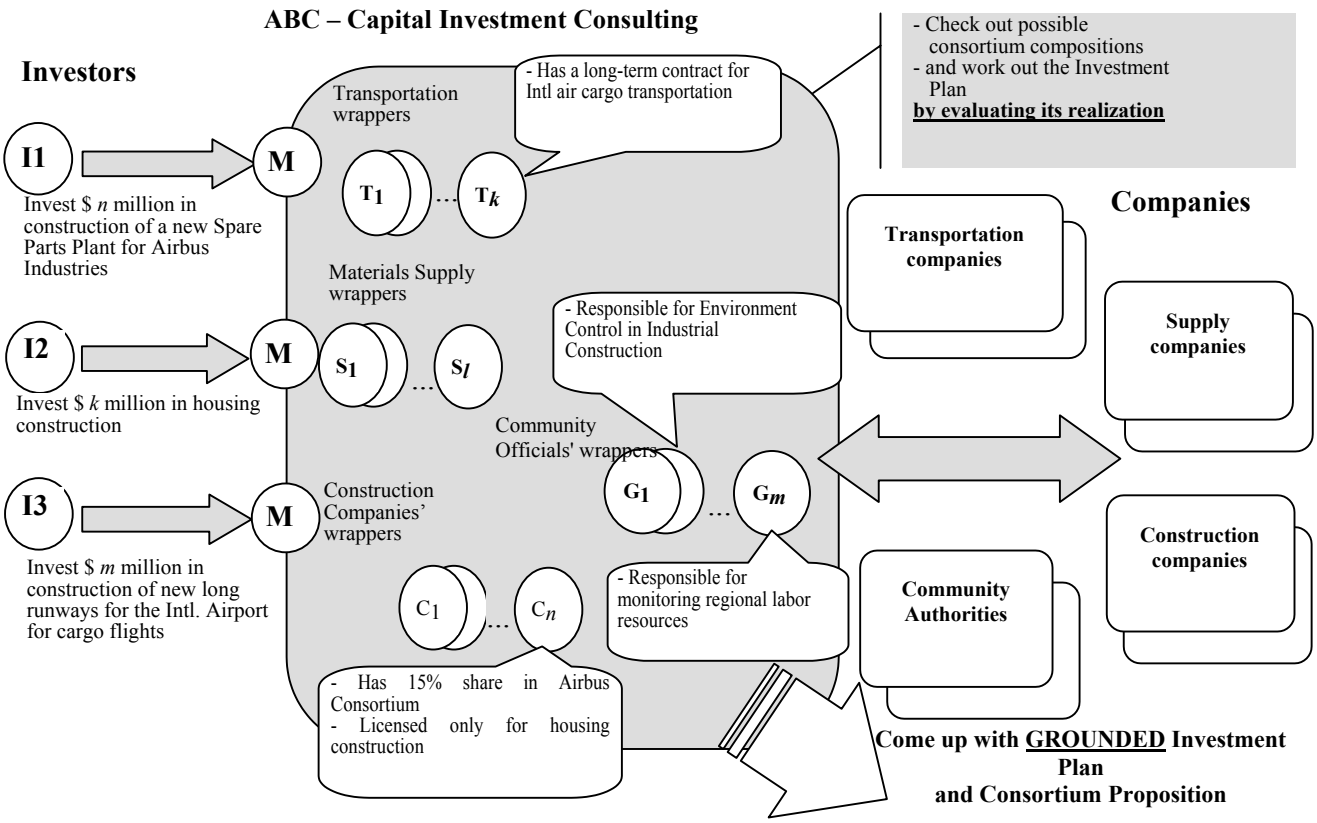


Рисунок 7.19 – Організація ABC, наміри інвесторів та інтереси акторів.

7.4 Семантичні аспекти моделювання електроній консалтінговій компанії

Попередні публікації авторів стосуються окремих моделей процесів: планування виконання проекту [18], прийом на роботу (вступ до аспірантури) [19], управління [20]. В цих окремих випадках використовувалась спрощена термінологія моделювання. Підсилення середовища для моделювання було по-перше викликано бажанням моделювати більш складні спільні взаємовідносини, поведінки, ніж погодження із 100% альтруїстичною поведінкою, для досягнення деякої організаційної цілі [28]. Наступний приклад додатка з галузі B2B Consulting E-може переконати в тому, що агенти повинні бути більш раціональними, а також і мати підвищену необхідність спільного виконання завдань.

Нехай ABC - успішна консультаційна компанія, що спеціалізується на інвестиціях у капітальне будівництво (див. Рис. 7.19). Будемо вважати, що ABC

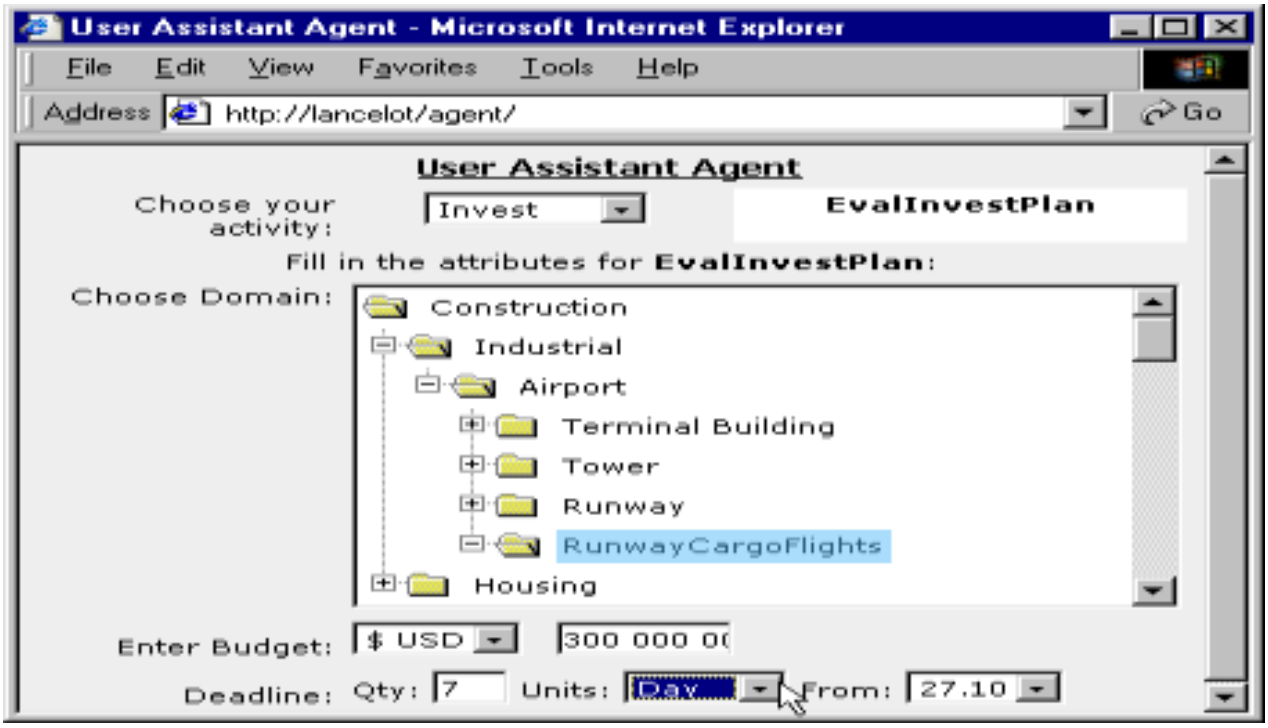


Рисунок 7.20 – І3 допомагає при формування завдання "Work out investment proposal ..."

складається із такого набору співробітників: менеджери проекту, представники будівельних компаній, представники компаній-постачальників матеріалів для будівництва, представники транспортних компаній, представники суспільних інститутів. Кожний із акторів представляє можливості і інтереси цих організацій. З іншого боку, актори є членами АВС і тому повинні дбати про успіх саме компанії і її прибутки. АВС і працює з перспективними інвесторами. Інвестори шукають як найвигідніше вкласти свої гроші.

Для досягнення успіху АВС треба забезпечити привабливі плани інвестиції з мінімальними ризиками. Такі плани інвестицій повинні підтримувати баланс взаємних інтересів як інвесторів, так і компаній – виконавців, в умовах обмежених ресурсів і обмежених можливостей виконавців. Інвестори і виконавці можуть також мати спільні або конфліктні інтереси, узгодження і наміри (Рис. 7.19).

Для забезпечення якісних пропозицій з інвестування у відповідь на запити інвесторів, АВС має моделювати (через імітацію і оцінку) процеси реалізації

проектів, аналізувати можливу поведінку підприємств – виконавців, і аналізувати ріски/прибутки в разі успішного або неуспішного завершення проекту. Також, ця компанія має забезпечити розумні рекомендації щодо впливу коректив на критичні ланки виконання проекту.

Інвестори – люди і їх асистенти - агенти **I1**, **I2**, **I3** представляють зовнішні впливи у вигляді завдань "Work out investment proposal ..." (“Розробити пропозицію на інвестиції...”). АВС моделюється у вигляді мультіагентної системи (МАС), яка складається із спеціалізованих агентів-виконавців: **M**, **T**, **S**, **G**, **C**. Комбінація цілей, уявлень і намірів таких агентів додає uncertainty, яка впливає на виконання завдань інвесторів і може проявитися в успішному або невдачному виконанні завдання. Екземпляри акторів моделюються як програмні агенти, розроблені на базі загального формального середовища [16]. Індивідуальні поведінки конкретних акторів T_1, ..., T_k, ..., C_n є функціями від чотирьох параметрів: ***capability (здатність)***, ***capacity(ємність)***, results ***desirability (бажаність результатів)***, ***credibility(міра довіри)*** (подробиці у розділі 5).

Наступний простий приклад буде використано для ілюстрації використання онтології завдання і переговорів в процесі оцінки плану інвестицій.

Нехай перспективний інвестор–людина звертається до агента – асистента користувача **I3** з метою сформувати таку пропозицію “Розробити пропозиції щодо вкладання $300 000 000 для створення нових довгих посадкових смуг для вантажних літаків в міжнародному аеропорту для розширення можливостей з прийому вантажних літаків. Результат потрібен через 1 тиждень починаючи з сьогодні.” Процес формулювання замовлення проходить з використанням онтології домену інвестицій, відому для **I3** (не розглядається в даній статті) як показано на Рис. 7.20.

```
(ask-one
:sender      "I3"
:receiver        "M"
:in-reply-to        Null
:reply-with      Null
:language        (XML)
:ontology        (Task)
:contents        (
<task>
 <activity>
  <name>EvalInvestPlan</name>
  <descr>Виробити план інвестиції: USD 3 000 мілліонів,
         побудова довгих посадкових сміг для вантажних
літаків.
         Час закінчення: 1 тиждень від сьогодня.</descr>
  <budget>   <value>USD, 300 000 000</value>
             <format>money</format> </budget>
  <deadline> <value>day, 7, today</value>
             <format>deadline</format> </deadline>
  <param> <name>Domain</name><value>
             Смуги для вантажних літаків</value>
             <format>Category</format> </param>
  <result-template><name>InvestPlan</name>
                   <spec>Investplan.dot</spec>
                   <format>MSWordDotFile</format>
  </result-template>
  <result-template><name>ConsortProp</name>
                   <spec>ConsortProp.dot</spec>
                   <format>MSWordDotFile</format>
  </result-template>
 </activity>
</task>             )  )
```

Рисунок 7.21 – KQML [21] повідомлення із *специфікацією діяльності* `EvalInvestPlan` на XML

Агент менеджера проекту компанії ABC (**M**) отримує це замовлення від **I3** у вигляді завдання, як показано на Рис. 7.21. Це завдання далі буде проаналізоване, декомпозоване і виконане у динамічно сформованій коаліції агентів-членів середовища ABC. Знання про те, як провести декомпозицію завдання і хто буде здатний виконати ці частини, розподілене і знаходиться в

базах знань агентів – виконавців. Це знання формалізоване за допомогою *Моделі Завдання* (розділи 4 і 5) і *Онтології Завдання* (розділ 6). Агенти приєднуються до коаліції виконавців завдання, беручи участь у торах на розподілення діяльностей. Механізм проведення таких переговорів реалізується у вигляді фази призначення роботи (розділ 5). Семантика, яку використовують агенти на фазі назначень формалізована *Онтологією Переговорів* (розділ 6).

7.4.1 Декомпозіція і виконання завдання

Вважаємо, що *завдання* $T = \{w^1, w^2, ..., w^k\}$ це множина із однієї або більше діяльностей. Діяльність w^k, що є атомарною для даного актора, іншими акторами може вважатися як завдання, у відповідності до їх баз знань, а саме до тих частин, які розпізнають зовнішній вплив.

Діяльність `EvalInvestPlan` , яку агент **I3** передає агенту **M** ,для **M** буде саме завданням[1]:

```
T = { w1 = (PerformArchitecturalDesign, …),
      w2 = (GetAirTrafficControlPermission, …),
      w3 = (CheckEnvironmentalRegulations, …),
      w4 = (BuildRunway, …),
          …,
      wk = (AssembleInvestPlan, …),}.
```

Актори також здатні створювати нові діяльності без зовнішнього впливу, у відповідь на деякі події, або під час виконання відомих діяльностей. Семантика завдання подається відповідно до *Онтології Завдання*. Ця онтологія використовується для:

Перевірки, чи знайомі актору діяльності із завдання

Декомпозиції діяльностей, які вважаються не атомарними

Перевірки, чи співпадають параметри і зразки результатів з уявленнями актора про параметри і результати даної діяльності

[1] Деталі опущені для стислості

Перевірки того, чи зможе актор виконати цю діяльність самостійно, або будуть проведені переговори на передачу виконання діяльності іншим акторам

Кожний із акторів, підключений до виконання завдання має власні уявлення про те, як, в якій послідовності виконувати атомарні діяльності, і скільки зусиль треба витратити на виконання діяльності. Тому будемо вважати (відповідно до моделі розділу 5), що кожний актор має відповідну робочу ємність (об'єм) для кожної відомої йому діяльності. Ці уявлення складають суб'єктивні Часткові Локальні Плани (ЧЛП,PLP) [105]. ЧЛП формалізовані в рамках онтології завдання і використані в макромодельних програмах під час виконання діяльності. ЧЛП відрізняються від, скажімо, GPGP [51] в тому, що в ЧЛП немає інформації про суб'єктивні уявлення про можливі діяльності інших акторів. Замість того актор використовує Матрицю Здатностей Партнерів (Fellows' ***Capabilities*** Matrix [106]) і Матрицю Довіри Партнерам (Fellows' ***Credibility*** Matrix [105]) для прийняття рішення про те, чи можуть інші актори виконати дану діяльність, або чи можна довіряти іншім акторам.

Вважається, що актор A характеризується своєю ємністю (***capacity***) $N_A(w^j)$ відносно до конкретної діяльності w^j у дискретному простору часу. Під ємністю розуміємо здатність актора виконати діяльність за інтервал часу τ. Якщо, наприклад, A доставляє будівельні матеріали в місто, тоді $N_A(w^j = (\text{"}deliver\_1\_tone\_of\_concrete\text{"},...)) = 4$ якщо A має одну вантажівку, яка може доставити за один раз до 4 тонн цементу і буде подвоєна, якщо A отримає ще одну таку ж вантажівку. Ємність актора може вважатися **нескінченною —** якщо в будь-який момент часу актор здатний доставити стільки цементу, скільки це потрібно. У випадку, коли термін виконання w^j буде залишатися $1 \times \tau$, оскільки актор оперувати тільки з одним типом вантажівок, мінімальний термін виконання діяльності не може бути менше за один інтервал часу. Такий випадок є ідеальним, коли немає інших діяльностей, що чекають на виконання. У випадку, коли ємність актора вважається **скінченною** (якщо в розпорядженні актора є тільки фіксована кількість готових вантажівок) $N_A(w^j)$ може бути

```
<Desirability>
  <Activity> <Name>DeliverConcrete</Name> </Activity>
  <Deadline> <Value>27.10.2001/20.00</Value>
             <Format>datetime</Format> </Deadline>
 <Time> <ZeroPoint>  <Value>27.10.2001/08.00</Value>
                     <Format>datetime</Format></ZeroPoint>
        <Granularity><Value>2</Value>
                     <Format>hours</Format></Granularity>
 </Time>
 <PointsNo>6</PointsNo>
 <TdfPoint> <TimeIncr>0</TimeIncr> <Incentive><Value>300</Value>
                   <Format>Money</Format></Incentive> </TdfPoint>
 …
 <TdfPoint> <TimeIncr>5</TimeIncr> <Incentive><Value>0</Value>
                   <Format>Money</Format></Incentive> </TdfPoint>
</Desirability>
```

Рисунок 7.22 – Функція бажаності (XML) для результатів діяльності `DeliverConcrete`

рівномірно, або нерівномірно, а, скажімо, відповідно до пріоритетів, які мають різні покупці, розподілена між діяльностями, що знаходяться у виконанні даного актора.

Діяльність w^j може бути обмежена часом закінчення (***deadline***) d_{w^j}. Час закінчення – це така точка в часі, після якої результати виконання w^j більше не потрібні агенту покупця. Наприклад, будівельної компанії цемент не буде потрібен у неділю, а лише у п`ятницю. Для конкретного прикладу доставки така функція бажаності (див. Розділ 5) може виглядати як на Рис. 7.22.

/

7.4.2 Фаза організації делегування робіт

Як було визначено в розділі 5, роль фази призначення робіт полягає в тому, щоб знайти виконавця конкретної діяльності, яку агент – ініціатор вирішує (згідно з ЧЛП, або в разі перенавантаження) передати іншим акторам. Призначення проводиться шляхом переговорів між ініціатором і групою акторів-учасників. Мета ініціатора полягає у оптимальному виконанні

```
<TdfFeedback>
 <activity> <name>DeliverConcrete</name> </activity>
 <PointsNo>2</PointsNo>
 <TdfPoint>
   <TimeIncr>t*wj </TimeIncr> <Incentive><value>tdf~(t*wj) </value>
                        <format>Money</format></Incentive>
 </TdfPoint>
 <TdfPoint>
   <TimeIncr>twj </TimeIncr> <Incentive><value>tdf~(twj) </value>
                          <format>Money</format></Incentive>
 </TdfPoint>
</TdfFeedback>
```

Рисунок 7.23 – Приклад відгуку з оцінками очікуваного компромісу ((XML) для діяльності `DeliverConcrete`.

діяльності. Вважається, що в процесі переговорів на призначення виконавців діяльності ініціатор намагається вирішити двокритеріальну задачу оптимізації. Першим критерієм оптимізації є термін отримання результатів діяльності. Другий критерій - це оптимальна ціна, яку треба буде заплатити виконавцям. Значення ціни вираховуються виходячи із бюджету для цієї діяльності (онтологія завдання).

Подивимось, які спільні концепції (помічені напівжирним) використовуються акторами ABC під час назначення діяльності `DeliverConcrete` activity. $\mathbf{C}_i$ стає ініціатором фази назначення після того, як він, генеруючи **діяльність** `DeliverConcrete` під час декомпозиції завдання `BuildRunway`, вирішив (згідно з **ЧЛП**) що він не здатен виконати цю діяльність самостійно і визначив, що $\mathbf{T}_1,\ldots\mathbf{T}_n$ є підходящими кандидатами для цієї діяльності, виходячи з даних в його матриці Можливостей партнерів (Fellows' Capabilities Matrix). $\mathbf{C}_i$ розповсюджує **специфікацію** діяльності `DeliverConcrete` і функцію бажаності результатів акторам $\mathbf{T}_1,\ldots\mathbf{T}_n$. Ці актори $\mathbf{T}_1,\ldots\mathbf{T}_n$ в свою чергу, після отримання зовнішнього впливу від $\mathbf{C}_i$ стають учасниками переговорів і відповідають відгуки на пропозицію у формі 2 – точкових відгуків з оцінками очікуваного компромісу (Рис. 7.23). Ці відгуки

базуються на оцінюванні доступної ємності акторів або на мірі зусиль, яких необхідно витратити для виконання діяльності.

До запланованої на майбутнє роботи належать: моделювання і концептуалізації альтернативних зразків переговорів (ітераційні переговори, аукціон …); подальше покращання онтології завдання шляхом забезпечення методів вирішення проблем і описом макромодельних процедур (що може вимагати використання розширень мови OIL); розробка прототипу B2B ринку консультацій по інвестиціям в агентській методології.

7.5 Підсумки

Розглянуті вище приклади моделювання реальних функціональних систем показують доцільність пропонованого підходу до моделювання складних функціональних систем.

ВИСНОВКИ

Дана науково-дослідницька робота присвячена розробці верифікації і часткової реалізації формальних математичних і алгоритмічних підходів, моделей і методів побудови Єдиного Інформаційного Простору Вузу, підприємства, організації.

В першому розділі пропоновано підходи і концепції до побудови проекцій ЄІП, а також формальних моделей об'єктів і агентів для проекцій ЄІП, що є використаними впродовж виконання роботи. Найбільш важливим результатом даного розділу є розроблена методика декомпозиції гіперкуба ЄІП на його базові складові - проекції. З методичної точки зору декомпозиція ЄІП на проекції підкоряється задоволенню двох аспектів: інтерфейсного і функціонального. Декомпозиція по інтерфейсним критеріям регулюється можливістю застосування тих або інших елементів УВІІ. Декомпозиція по функціональній приналежності визначається операційним призначенням структурованої сукупності компонент - проекції. Топологічна проекція містить компоненти відбивають у ЄІП природне розташування, форми, пропорції і взаємозв'язок по рівнях вкладеності моделей реальних елементів підприємства. Організаційна проекція моделює структуру й організаційну взаємодію підрозділів підприємства, представляючи тим самим відносини керування і взаємодії. Функціональна проекція моделює виконання процесів, що відбуваються усередині підприємства, підрозділу. Сформульовано також загальні припущення і підходи до розробки активних компонент проекцій ЄІП – агентів. Базовими принципами для цього обрано принципи діакоптики, побудови мультіагентських систем.

В другому розділі даного звіту представлено побудована архітектура розподіленої Інформаційної Системи (ІС) організації для представлення проекцій ЄІП. Моделі проекцій ЄІП будуються на основі операційних компонентів різних рівнів архітектури ІС ЄІП. Одним з важливих додаткових результатів є показана можливість реалізації проекцій і елементів УВІІ на базі

багаторівневої розподіленої інтегрованої інформаційної системи підприємства. Викладені способи реалізації дозволяють створювати гнучкі адаптивні і мобільні компоненти проекцій ЄІП на базі зазначених архітектурних принципів і методики застосування активного словника даних. Розроблена архітектура досить повна і має достатні засоби для застосування в практичній реалізації компонентів проекцій ЄІП.

В третьому розділі представлено розробка базових моделей і теоретичного апарату представлення активних функціональних об'єктів ЄІП. Розроблена модель функціональної системі/компоненти ЄІП. Дана модель розбудована на основі диакоптичного підходу до моделювання складних динамічних систем, з використанням принципу макромоделювання. Функціональні елементи моделі представлені інтелектуальними агентами. Для моделювання агента розроблена узагальнена модель на базі теорії кінцевих автоматів і загальноприйнятих стандартизованих принципів взаємодії інтелектуальних агентів у МАС. Модель процесу виконання завдань (упорядкованих множин робіт) побудована на базі властивостей моделі функціональної системі/компоненти ЄІП. Розроблено також архітектуру узагальненого агента.

Четвертий розділ звіту презентує розроблені моделі і апарат взаємодії автономних інтелектуальних компонент (агентів), що діють в функціональній проекції ЄІП. Розв’язаними є ключові проблеми організації взаємодії агентів, як автономних істот: реалізація засобів передачі впливів і повідомлень (комунікації), узгодження спільних робіт (координації) між агентами що діють спільно для вирішення спільної проблеми, або виконання спільного завдання, здатності агентів постійно поновлювати свої уявлення про партнерів (еволюціонувати) у процесі спільної роботи.

П’ятий розділ присвячений розробленим моделям і методам формування агентами динамічних коаліцій для спільного використання завдань в ЄІП. Презентована формальна модель для розподіленої координації формування динамічної коаліції між раціональними (такими, що мають істотний егоїзм в

намаганні отримати максимальну вигоду та такими, що кооперуються (намагаючись досягнути оптимального виконання діяльності у спільній праці один з одним) агентами. Такі агенти є функціональними акторами, які здатні виконувати спеціалізовані діяльності в організації. Співвідношення між доброзичливістю і раціоналізмом учасника коаліції регулюється узгодженням відносної кооперації, узгодженням при зазначенні діяльності та узгодженням доставки результатів

Переговори на залучення учасника до коаліції завдання проводяться кожний раз як в процесі виконання завдання з'являється нова діяльність. Процес переговорів розглядається як спеціальний тип координації і виконується під час так званої фази призначення виконавців, яка передує виконанню діяльності. Під час фази призначення, яка проводиться агентом – ініціатором з використанням спрощеного протоколу FIPA Contract Net Protocol.

Шостий розділ звіту презентує вирішення проблеми семантичної інтероперабельності в суспільствах і коаліціях взаємодіючих агентів ЄІП. В розділі конкретизовано стан сучасних систем інтеграції неоднорідних баз даних і баз знань, і підходів, використаних у цих системах та виділені узагальнюючі фактори, яки впливають на розв'язання завдань інтеграції неоднорідних ресурсів в таких складних системах, як ЄІП. Перший фактор – використання онтологій як специфікацій предметних областей, задач, методів рішення задач. Другий фактор – розвиток інтернет-технологій, що дозволяють зробити різнорідні інформаційні ресурси «ближче до користувача». Третій фактор – застосування систем інтелектуальних агентів для реалізації архітектури медіатора неоднорідних розподілених інформаційних ресурсів. В роботі запропоновано і розроблено класифікацію онтологій за виразною силою їх формальних теорій, за їх функцією в медіаторній ІС ЄІП – онтології ІР, загальна онтологія ІС і онтології предметних областей. Запропонована модель онтології $< O, \Omega >$ яка складається із двох частин: декларативної $O =< X_1, X_2, R, A >$ – для опису елементів концептуалізації, і маніпулятивної –

набору примітивів модифікації Ω. Дана структура загального типу онтології. Модифікаційні примітиві забезпечено для кожного структурного елементу онтології загального типу. Набір інваріантів модифікації і набір правил вирішення конфліктів сформульовано для таксономії. Також окреслені функції сервісів ІС для проведення моніторингу онтологій, матчінгу і вирівнювання онтологій.

Роль онтології завдання і переговорів, запропонованих в даному розділі, полягає в забезпеченні спільної концептуалізації концепції, структур і процедур, що їх використовують агенти в процесах аналізу діяльності, її декомпозиції, виконання або передачі іншим агентам ЄІП. Ці онтології формалізовані за допомогою мови OIL і переведені в нотації DAML (RDF), RDFS, SHIQ, таким чином стає можливим використання цих онтологій в стандартах розмітки сервісів, що розробляються у W3C.

Заключний сьомий розділ звіту присвячений практичним питанням верифікації розроблених підходів, моделей і методів шляхом їх використання для моделювання і управління в галузях планування, дистанційного навчання, побудови інтелектуальних програмних систем для е-комерції. Розглянуті приклади моделювання реальних функціональних систем показують доцільність розробленого підходу до моделювання і практичної реалізації таких складних динамічних функціональних систем, як ЄІП.

Таким чином, в результаті виконання роботи було отримано:

- розроблені підходи, моделі і методі декомпозиції єдиного інформаційного простору (ЄІП) на проекції, розроблені базові підходи до моделювання активних об'єктів і механізмів їхньої взаємодії в ЄІП;
- розроблені формальні принципі побудови моделей об'єктів для різних проекцій ЄІП на базі принципів диакоптики, архітектур типу майстер-агент і стандартів розподілених систем;
- розроблені формальні принципі моделі і методи взаємодії моделей функціональних об'єктів (агентів) для різних проекцій ЄІП.

- розроблені моделі і агенти топологічної, організаційної і функціональної проекцій ЄІП;
- розроблени моделі і методи кооперації агентів в ЄІП для спільного виконання завдань;
- формально вирішена проблема семантичної інтероперабельності між агентами ЄІП, яки спільно виконують завдання;
- формальні методи веріфіковані шляхом їх застосування до моделювання 4-х типових модельних задач із різних галузей: планування проекту, моделювання виробничого процесу, конкурсного відбору аспірантів для дистанційного навчання, моделювання електроного консалтингового сервісу в галузі інвестицій в капітальне будівництво.

Одним із суттєвих результатів роботи є демонстрація можливості застосування принципу макромоделювання для організації математичної моделі складної функціональної системи в термінах інформаційної взаємодії компонент і функціонування системи в цілому.

У результаті проведеного дослідження вдалося раціонально декомпозувати досить складну задачу, виділити доцільні для інформаційної моделі режимні параметри, розробити формальні методи організації математичної моделі системи в цілому і формальні принципи організації макромоделі – об'єкта системи.

Результати роботи апробовані на:

- міжнародній Науково - практичній конференції із програмування УкрПРОГ'2000, Кіев;
- міжнародній Науковій конференції "Інтелектуальна обробка інформації" - ІОІ'2000, Алушта;
- міжнародном конгресі "Інформаційне суспільство в Україні" - Конгрес'2000, Кіев;

- міжнародній науковій конференції "Information Society of the XXI century: Emerging technologies and New Challenges" - IS'2000, Аізу-Вакамацу, Япония;
- міжнародній науковій конференції “Information Systems Technology and Applications”- ISTA’2001, Харків Україна, 2001р.
- міжнародній науковій конференції “20-th International Conference on Entity-Relationship Approach” – ER’2001, Йокогама, Японія 2001р.

По результатам виконання даної роботи були опубліковані дві глави в колективних монографіях (видавництва ЗДУ і Springer Verlag) і п’ять статей в фахових наукових збірниках.

Принципова наукова новизна роботи полягає в тому, що вдалося розвити принципі диакоптики в застосуванні до моделей взаємодіючих інтелектуальних інформаційних агентів і розподілених архітектур, що побудована на базі підходів стандарту CORBA і методології майстер - агент.

ПЕРЕЛІК ПОСИЛАНЬ

1. Dewey et. al. The Impact of NIIIP Virtual Enterprise Technology on Next Generation Manufacturing. In Proc. of Conference on Agile and Intelligent Manufacturing Systems, 1996
2. Davulcu, H., Kifer, M., Pokorny, L. R., Ramakrishnan, C. R., Ramakrishnan I. V. Modeling and Analysis of Interactions in Virtual Enterprises. //In: Proc. of the 9-th International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises (RIDE-VE'99), March 23-24, 1999, Sidney, Australia.
3. Lupu, E., Milosevic, Z., Sloman, M. Use of Roles and Policies for Specifying, Building and Managing Virtual Enterprise. //In: Proc. of the 9-th International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises (RIDE-VE'99) , March 23-24, 1999, Sidney, Australia.
4. Baral, C., Lobo, J., Formalizing Workflows as Cooperative Agents In Proc. of DYNAMICS 97 (a workshop in ILPS 97).
5. Ермолаев В. А., Плецкий С. Ю., Толок В. А. Архитектура унифицированного информационного пространства виртуального университета. Вестник Запорожского госуниверситета, № 2, 1998 г., стр 44-53.
6. R. Otte, P. Patrick, M. Roy. Understanding Corba, 1/e. - Prentice Hall Professional Technical Reference, ISBN 0-13-459884-9, 1996, 288 pp.
7. Kron, G., Diacoptics. Macdonald, London, 1963.
8. Klusch, M. (Ed.) Intelligent Information Agents. Agent based Information Discovery and Management on the Internet, - Springer, ISBN 3-540-65112-8, 1999, 498 pp.
9. Storey, V. C., Dey, D., Sundaresan, S., Ullrich, H. and Yamakawa, S., Learning Accross Application Domains for Database Design Systems. Proc. of the 6-th Workshop on Information technologies and Systems (WITS'96), Cleveland, Ohio, 1996.

10. Глушков, В. М. Введение в кибернетику. Изд-во АН УССР, Киев, 1964, 323 стр.
11. Anghern, A Designing Mature Internet Business Strategies: The ICDT Model.// European Management Journal, Vol. 15, No 4, 1997, pp. 361-369.
12. I3Net European Research Initiative http://www.i3net.org
13. Papazoglou, M. P., Van der Heuvel, W.-J. From Business Processes to Cooperative Information Systems: An Information Agents perspective.//In: M. Klusch (Ed.) Intelligent Information Agents: Agent Based Information Discovery and Management on the Internet. Springer-Verlag, 1999, pp. 10-36.
14. OMG Unified Modeling Language. Specification. Version 1.3. June 1999. http://www.rational.com/uml/resources/documentation/
15. Sycara, K., Decker, K., Pannu, A. Williamson, M. and Zeng, D. Distributed Intelligent Agents. // IEEE Expert, Dec. 1996, pp. 36-45.
16. V. A. Ermolayev, S. U. Borue, V. A. Tolok, N. G. Keberle, Use of Diakoptics and Finite Automata for Modelling Virtual Information Space Agent Societies // "Lecture Notes of Zaporozhye State University", ISBN 966-599-058-4, Vol. 3, No 1, 2000, pp. 34-44.
17. Борю С.Ю., Ермолаев В.А., Толок В.А. О диакоптическом подходе к моделированию процессов во многофункциональных информационных системах // (Спец. выпуск) Доклады международной конференции KDS'99, Кацивели, 13-18.09.1999 / Искусственный интеллект №2, 1999, с. 211-219, ISSN 1561-5359.
18. V. A. Ermolayev, V. A. Tolok, Interfaces and Human - Agent Interaction in Virtual Information Spaces, Panel talk at ESPIRIT AgentLink SIG on Intelligent Information Agents meeting, London, GB, Apr. 21-23, 1999, 10 p.
19. Ermolayev, V. A., Tolok, V. A., Visual Intranet Interfaces and Architecture of Unified Information Space in the Concept of Virtual University at ZSU, Proc. ENCOM-98, Atlanta, USA, June 1998, 9 pp.
20. Demirchan, K. S., Rakitsky, U. V., Butuirin, P. A., Kartashev, E. N., Korovkin, I. V., Problems of Numerical Modeling of the Processes in Electric Circuits.

Lecture Notes of the Academy of Sciences of the USSR. Energetics and Transport. No 2, 1982, pp. 94-114.

21. Butuirin, P. A., Borue, S. U., Analytical Transformations of Characteristic Equations of Electrical Machines. Lecture Notes of the Academy of Sciences of the USSR. Energetics and Transport. No 2, 1986, pp. 24-34.
22. Schmitt, G. Saake, Merging Inheritance Hierarches for Schema Integration Based on Concept Lattices, Fakultät für Informatik, Otto-von-Guericke-Universität, Magdeburg, Preprint Nr. 2, 1997, 29 p.
23. Schmitt, Flexible Integration and Derivation of Heterogenious Schemata in Federated Database Systems, Fakultät für Informatik, Otto-von-Guericke-Universität, Magdeburg, Preprint Nr. 10, 1995, 32 p.
24. V. A. Ermolayev. Object Oriented Dynamic Data Modelling and Active Data Dictionaries - Some Crosspoints . Вісник Запорізького державного університету, №2, 1998, стр. 53-62.
25. Nwana, H. S. Software Agents: an Overview. // Knowledge Engineering Review, Vol. 11, No 3, pp. 205-244, Oct./Nov. 1996.
26. Adam, N., Atluri, V. and Huang, W. Modeling and analysis of workflows using petri nets. // Journal of Intelligent Information Systems, 10(2), pp. 131-158, March 1998.
27. Foundation for Intelligent Physical Agents (FIPA) Spec: DRAFT, Version 0.2, Agent Communication Language, 1999, http://www.fipa/org
28. Rao, A. S. and Georgeff, M. P. Modeling rational agents within a BDI-architecture. In Fikes, R. and Sandewall, E., editors, Proceedings of Knowledge Representation and Reasoning (KR&R-91), 1991, pages 473–484. Morgan Kaufmann Publishers: San Mateo, CA.
29. Yannis Labrou, Tim Finin, and Yun Peng: "Agent Communication Languages: The Current Landscape"
30. Object Managament Group, “The Common Object Request Broker: Architecture and Specifications”, OMG Document Number 91.12.1, December 1991.

31. Finin, T. and Fritszon, R. KQML - A language for protocol and information exchange. // Proc 13th DAI workshop, pp. 127-136, Seattle, WA, USA.
32. Sycara, K., Decker, K., Pannu, A. Williamson, M. and Zeng, D. Distributed Intelligent Agents. // IEEE Expert, Dec. 1996, pp. 36-45.
33. Hyacinth Nwana, Lyndon Lee, and Nick Jennings: "Coordination in Software Agent Systems", BT Technology Journal, 14(4), 1996, pp 79-88.
34. S. U. Borue, V. A. Ermolayev, V. A. Tolok: Application of Diakoptical MAS Framework to Planning Process Modelling // in: "Problems of Programming" Scientific Journal №1-2, 2000, ISBN 966-02-1244-5, Special Issue: the Proc. of the 2-nd Intl. Scientific - Practical Conference on Programming (UkrPROG'2000), Kiev, 23-26 May 2000, p. 488-500
35. Genesereth M.R., Fikes R.E., et.al. Knowledge Interchange Format Version 3.0 Reference Manual. Logic-92-1, Stanford University Logic Group, 1992.
36. Vadim Ermolayev: Dynamic Agent Communities Facilitating to Distant Learning in a Virtual University Information Space. // Proc. of Intl. Conf. Information Society in the 21st Century (IS2000), Spec Session on Virtual Universities and Distant Educartion (VUDE), Aizy-Wakamatsu, Japan, Nov. 5-8, 2000, pp. 488-495.
37. Lesser V.R.: Reflections on the Nature of Multi-Agent Coordination and Its Implications for an Agent Architecture, Journal of Autonomous Agents and Multi-Agent Systems, 1(1), 89–111, 1998.
38. M. Klusch (ed.): Intelligent Information Agents, Springer-Verlag, 1999.
39. Omicini, A., Zambonelli, F., Klusch, M., Tolksdorf, R. (Eds.) Coordination of Internet Agents: Models, Technologies, and Applications, Springer-Verlag, 2001, 523 pp.
40. Nwana H.S., Ndumu D.T., Lee L.C., Collis J.C.: ZEUS: A Collaborative Agents Tool-Kit, Proc. of 2nd UK Workshop on Foundations of Multi-Agent Systems (FoMAS’97), Warwick, 45–52, 1997.

41. Tambe M., Shen W., Mataric M., Goldberg D., Modi J., Qiu Z., Salemi B.: Teamwork in Cyberspace: Using TEAMCORE to Make Agents Team-Ready, Proceedings of AAAI Spring Symposium on Agents in Cyberspace, 1999.
42. Lesser V.R.: Reflections on the Nature of Multi-Agent Coordination and Its Implications for an Agent Architecture, Journal of Autonomous Agents and Multi-Agent Systems, 1(1), 89–111, 1998.
43. D. E. Neiman, D. W. Hildum, V. R. Lesser, T. W. Sandholm. Exploiting Meta-Level Information in a Distributed Scheduling System. // In Proc. of the Twelfth National Conference on Artificial Intelligence, USA, Seattle, Aug. 1994
44. Tsvetovat, M., Sycara, K., Chen, Y. and Ying, J. "Customer Coalitions in the Electronic Marketplace", Agents 2000 Conference, Barcelona, Spain, June 3-June 8, 2000.
45. http://www.cs.cmu.edu/~softagents/coala.html Last accessed on 10 Apr. 2001.
46. Thomas W. Malone and Kevin Crowston. Toward an Interdisciplinary Theory of Coordination. Center for Coordination Science. Technical Report 120, MIT Sloan School of Management, 1991.
47. Paolo Ciancarini, Andrea Omnichini, Franco Zambonelli, Coordination Models for Multi-Agent Systems // AgentLink News, No 3, July 1999, pp. 3-6.
48. Gasser, L., "DAI Approaches to Coordination" in Distributed Artificial Intelligence: Theory and Praxis (eds. N. M. Avouris and L. Gasser) Kluwer Academic Publishers pp 31-51.
49. Werner, E., "Cooperating Agents: A Unified Theory of Communication and Social Structure" in Distributed Artificial Intelligence Vol II (eds. L. Gasser and M. N. Huhns), 1989, pp 3
50. D. Fensel, M. Crubezy, F. van Harmelen, and I. Horrocks: OIL & UPML: A Unifying Framework for the Knowledge Web. In Proceedings of the Workshop on Applications of Ontologies and Problem-solving Methods, 14th European Conference on Artificial Intelligence ECAI'00, Berlin, Germany August 20-25, 2000.

51. Keith S. Decker, Environment Centered Analysis and Design of Coordination Mechanisms. Umass CMPSCI Tech. Rep. 95-69, 1995, 199 p.
52. N. R. Jennings: Coordination Techniques for Distributed Artificial Intelligence, in Foundations of Distributed Artificial Intelligence(eds. G. M. P. O'Hare and N. R. Jennings), Wiley, 1996, pp. 187-210.
53. W. Richard Scott. Organizations: Rational., Natural and Open Systems. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1987.
54. FOUNDATION FOR INTELLIGENT PHYSICAL AGENTS. FIPA Contract Net Interaction Protocol Specification. Version E. Ref. No XC00029E. 2001. http://www.fipa.org/specs/fipa00029/ Last accessed on 10 Apr. 2001
55. Plaksin, S. L. Co-ordination Models and Algorithms in Document Flow Scenaria. (Msci. Thesis), Technical Rep., ZSU, 2001.
56. Калиниченко, Л.А., Рывкин, В.М., Чабан, И.А. Принципы организации и архитектура СИЗИФ - системы организации интегрированных баз данных // Программирование. – Москва, 1975, № 4.
57. Калиниченко Л.А., Рывкин В.М., Чабан И.А. Основные особенности языка манипулирования данными в системе интегрированного запоминания информации СИЗИФ // Программирование. – Москва, 1975, № 6.
58. Калиниченко Л.А. Методы и средства интеграции неоднородных баз данных. – Москва: Финансы и Статистика, 1983. – 300 с.
59. Adiba M.et al. POLYPHEME:An Experience in Distributed Data Base System Design and Implementation.-In: Proc. of the International Symposium on Distributed Data Bases. Paris. Amsterdam: North-Holland, 1980.
60. Cardenas A.F, Pirahesh M.H. Data Base Communication in a heterogeneous data base management system network. -Information Systems, 1980, 5, p.55-79.
61. Программа исследований в области баз данных на следующее десятилетие (Асиломарский отчет о направлениях исследований в области баз данных) // Открытые системы. – Москва, 1999, №1.

62. Брюхов, Д.О., Задорожный, В.И., Калиниченко, Л.А., Курошев, М.Ю., Шумилов, С.С. Интероперабельные информационные системы: архитектуры и технологии. // СУБД. Москва, 1995, №4. – С.86-113
63. Object Managament Group, “Object Managament Architecture Guide”, OMG Document Number 91.11.1, September 1, 1992.
64. Object Managament Group, “Object Services Architecture”, Revision 8.0.
65. Object Managament Group, “The Common Object Request Broker: Architecture and Specifications”, OMG Document Number 91.12.1, December 1991.
66. Sheth A.P. Changing Focus on Interoperability in Information Systems: from System, Syntax, Structure to Semantics. In: Interoperating Geographic Information Systems. Goodchild M.F., Egenhofer M.J., Fegeas R. and Kottman C.A. (eds.). Kluver. 1998.
67. Guarino N. The Role of Ontologies in Information Systems Design. Proceedings of the First International Conference on Formal Ontologies, FOIS’98.
68. Uschold M., Gruninger M. Ontologies: Principles, Methods and Applications. Knowledge Engineering Review, 11(2), 1996.
69. Gruber T. A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 5:199-220, 1993.
70. Lenat D. et al. CYC: Toward programs with Common Sense, Communications of the ACM, Vol.33, No.8, august 1990, p. 30-49.
71. Guarino N., The Ontological Level. In: Casati R., Smith N. and White G.(eds.), Philosophy and the Cognitive Sciences, Vienna: Ho:lder-Pichler-Tempsky, 1994.
72. Arens Y., Knoblock C.A., Shen W. Query Reformulation for Dynamic Information Integration. Journal of Intelligent Information Systems. 1996.
73. Adali S., Subrahmanian V.S. Amalgamating knowledge bases, II. Distributed mediators. International Journal of Intelligent and Cooperative Information Systems 3(4): 349-383, 1994.
74. Bayardo et al. InfoSleuth: Semantic Integration of Information in Open and Dynamic Environment. In Proceedings of the 1997 ACM International

Conference on the Management of Data (SIGMOD), Tucson, Arisona, May 1997.

75. Garcia-Molino H. et. Al. The TSIMMIS Approach to Mediation: Data Models and Languages. In Proceedings of the NGITS (Next Generation Information Technologies and Systems), June 1995.
76. Levy A., Srivastara D., Kirk T. Data Model and Query Evaluation in Global Information Systems, Journal of Intelligent Information Systems, 5(2), September 1995.
77. Mena E., Kashyap V., Sheth A., Illaramendi A. OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-Existing Ontologies. In Proceedings of the First IFCIS International Conference on Cooperative Information Systems (CoopIS'96), Brussels (Belgium), June. IEEE Computer Society Press, 1996.
78. Lu J., Nerode A., Subrahmanian V.S. Hybrid Knowledge Bases, IEEE Transactions on Knowledge and Data Engineering, 1994.
79. Benjamins V.R. et al. IBROW3: An Intelligent Brokering Service for Knowledge-Component Reuse on the World – Wide Web. Proceedings of the 11th Workshop on Knowledge Acquisition, Modeling and Management, KAW'98.
80. Guarino N., Masolo C., Vetere G. Content-Based Access to the Web. IEEE Intelligent Systems, May/June 1999, p.70-80.
81. Sowa J. Conseptual Structures: Information processing in Mind and Machine. Addison-Wesley, Reading, Mass.,1984.
82. Sheth, A. et.al.: OBSERVER: An Approach for Query Processing in Global Information System based on interoperation across Pre-existing Ontologies. Distributed and Parallel Databases 8(2): 223-271 (2000).
83. Bayardo Jr. R. J., et al.: InfoSleuth: Agent-Based Semantic Integration of Information in Open and Dynamic Environments (Experience Paper). In (Peckham, J. Ed.): Proc. ACM SIGMOD Int. Conf. on Management of Data, Tucson, 1997. ACM Press, 1997; pp.195-206.

84. Oliver, D.E.; Shahar, Y.; Musen, M.A.; Shortliffe, E.H.: Representation of Change on Controlled Medical Terminologies. Artificial Intelligence in Medicine 15(1):53-76, (1999).
85. Van der Vet, P.E.; Mars, H.J.; Speel, P.H.: The Plinius Ontology of Ceramic Materials. In (Cohn, A.G. Ed.): Proc. of 11th European Conf. In Artificial Intelligence (ECAi'94), Amsterdam, 1994. John Wiley and Sons, 1994.
86. Bergamaschi, S.; Sartori, C.: On Taxonomic Reasoning in Conceptual Design. ACM TODS 17(3): 385-422 (1992).
87. Storey, V. C.; Chiang, R. H. L.; Dey, D.; Goldstein, R. C.; Sundaresan, S.: Database design with Common Sense Business Reasoner. ACM TODS 22(4): 471-512 (1997).
88. Uschold, M.: Creating, Integrating and Maintaining Local and Global Ontologies. In (Horn, W. Ed.): Proc. of 14th European Conf. on Artificial Intelligence (ECAI'00), Berlin, 2000. IOS Press, Amsterdam, 2000.
89. Goñi, A.; Illarramendi, A.; Mena, E.; Blanco, J.M.: Monitoring the Evolution of Databases in Federated Relational Database Systems. In (Conrad, S.; Hasselbring, W.; Heuer, A.; Saake, G. Eds.): Proc. of Int. CAiSE'97 Workshop on Engineering Federated Database Systems (EFDBS'97), Barcelona, 1997. Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, Preprint Nr. 6, 1997; pp.81-91.
90. Zicari, R.: A Framework for Schema Updates in an Object Oriented Database System. In: Proc. of the 7th Int. Conf. on Data Engineering (ICDE'91), Cobe, 1991. IEEE Computer Society, 1991; pp.2-13.
91. Benatallah, B.: A Unified Framework for Supporting Dynamic Schema Evolution in Object Databases. In (Akoka, J.; Bouzeghoub, M.; Comyn-Wattiau, I; Métais, E. Eds.): Proc. of the 18th Int. Conf. on Conceptual Modeling (ER 1999), Paris, 1999. Springer, 1999; pp.16-30.
92. Bernstein, P.A.; Rahm, E.: Data Warehouse Scenarios for Model Management. In (Laender, A.H.F.; Liddle, S.W.; Storey, V.C. Eds.): Proc. 19th Int. Conf. on Conceptual Modeling (ER2000), Salt Lake City, 2000. Springer, 2000; pp.1-15

93. OKBC standard. Version 2-0-3, April 1998. http://www.ai.sri.com/~okbc/ - URL was accessed 8 May. 2001.
94. http://www-ksl-svc.stanford.edu:5915/doc/chimaera/chimaera-docs.html - URL was accessed 8 May 2001
95. Noy, N. F.; Musen M.: PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In: Proc. of the 17th National Conf on AI / 12th Conf. on Innovative Applications of AI (AAAI/IAAI 2000); AAAI Press / The MIT Press, 2000; pp.450-455.
96. http://www.ontoknowledge.org/wpdes.shtml#wp1 - URL was accessed 8 May 2001.
97. Omelayenko, B.: Syntactic-level Ontology Integration Rules for E-commerce. In: Proc. of the 14th Int. Conf.(FLAIRS 2001), Key West, 2001. AAAI Press, 2001 (to appear).
98. Storey, V. C.: Understanding semantic relationships. The Very Large Data Bases (VLDB) Journal 2(4):455-488 (1993).
99. http://www.ksl.stanford.edu/software/ontolingua/project-papers.html - URL was accessed 8 May 2001.
100. Klein, M.; Fensel, D.; van Harmelen, F.; Horrocks, I.: The Relation between Ontologies and Schema languages: Translating OIL-specifications in XML-Schema. In (Horn, W. Ed.): Proc. of 14th European Conf. on Artificial Intelligence (ECAI'00), Berlin, 2000. IOS Press, Amsterdam, 2000.
101. Калиниченко, Л. А.: СИНТЕЗ: язык определения, разработки и программирования интероперабельных сред разнородных информационных ресурсов. Институт проблем информатики РАН, Москва, 1993.
102. Zaniolo, C.; Ceri, S.; Faloutsos, C.; Snodgrass, R.T.; Subrahmanian, V.S.; Zicari, R.: Advanced Database Systems. Morgan Kaufmann Publishers, San Francisco, 1997.
103. Bergamaschi, S.; Castano, S.; De Capitani di Vimercati, S.; Montanari, S.; Vincini, M.: An Intelligent Approach to Information Integration. In (Guarino, N.

Ed.): Proc. of the 1st Int. Conf. on Formal Ontologies in Information Systems (FOIS'98), Trento, 1998. IOS Press/ Ohmsha, 1998; pp. 253-268.

104. N. R. Jennings, P. Faratin, M. J. Johnson, P. O'Brien, M. E. Wiegand: "Using Intelligent Agents to Manage Business Processes", Proc. First Int. Conf. on The Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM96),pp. 345-360. London, UK.

105. Ermolayev, V. A., Borue, S. U., Tolok, V. A., Keberle, N. G., Plaksin, S. L.: Arranging Co-operative Activities in Dynamic Coalitions of Self-Interested Actors. Technical Report. Dept. of Math. Modelling and IT, Zaporozhye State University (2001)

106. Ermolayev, V. A, Tolok, V. A.: Modelling Distant Learning Activities by Dynamic Agent Task Coalitions. To appear In: (Noguchi, S., Yu, C. Eds.): Enabling Society with Information Technology. Springer-Verlag, Tokyo (2001)