НАЦІОНАЛЬНИЙ СТАНДАРТ УКРАЇНИ

Інформаційні технології

АРХІТЕКТУРА ВІДКРИТОГО РОЗПОДІЛЕНОГО КЕРУВАННЯ ТА ПІДТРИМКА ЗАГАЛЬНОЇ АРХІТЕКТУРИ БРОКЕРА ОБ'ЄКТНИХ ЗАПИТІВ (CORBA)

(ISO/IEC 13244:1998, Amd/1:1999, MOD)

ДСТУ 4071:2001

Видання офіційне

Київ
ДЕРЖСПОЖИВСТАНДАРТ УКРАЇНИ
2004

ПЕРЕДМОВА

1 РОЗРОБЛЕНО: Технічний комітет зі стандартизації «Інформаційні технології» (ТК 20) при Держспоживстандарті України та Інститут кібернетики ім. В. Глушкова НАН України

РОЗРОБНИКИ: **О. Демська-Кульчицька,** канд. філол. наук; **С. Зайцева,** канд. фіз.-мат. наук; **Б. Кульчицький; О. Перевозчикова,** проф., д-р фіз.-мат. наук (керівник розробки)

2 ЗАТВЕРДЖЕНО ТА НАДАНО ЧИННОСТІ: наказ Держстандарту України від 12 березня 2002 р. № 146 з 2003–07–01 зі зміною терміну чинності згідно з наказом № 60 від 31 березня 2004 р.

3 Цей стандарт відповідає ISO/IEC 13244:1998 | ITU-T Recommendation X.703 Information technology — Open Distributed Management Architecture (Інформаційні технології. Архітектура відкритого розподіленого керування) / Amd/1:1999 Support using Common Object Request Broker Architecture (CORBA) (Підтримка використання загальної архітектури брокера об'єктних запитів)

Ступінь відповідності — модифікований (MOD)

Переклад з англійської (en)

4 УВЕДЕНО ВПЕРШЕ

ЗМІСТ

С.

Національний вступ IV

1 Сфера застосування 1

2 Нормативні посилання 3

2.1 Ідентичні рекомендації | міжнародні стандарти 3

2.2 Додаткові джерела 4

2.3 Відкриті джерела специфікації 4

3 Терміни та визначення понять 4

3.1 ODP-RM визначення 4

3.2 Визначення OSI-керування 6

3.3 Додаткові визначення 6

4 Скорочення 7

5 Вимоги 8

6 Загальне середовище 9

6.1 Основи 9

6.2 Архітектура 9

6.2.1 Погляд з позицій предметної області 9

6.2.2 Інформаційний погляд 10

6.2.3 Обчислювальний погляд 10

6.2.4. Інжиніринговий погляд 14

6.2.5 Технологічний погляд 15

6.3 Багаторазове використовування ODMA-специфікацій 15

7 Підтримування OSI-керування для ODMA 15

7.1 Обчислювальний погляд 16

7.1.1 Зв'язні відгуки 18

7.1.2 Рівні обчислювальної абстракції 18

7.2 Інжиніринговий погляд 20

7.2.1 Створювання і видаляння керованих об'єктів 22

7.2.2 Обробляння запитів операцій 22

7.2.3 Нав'язування політики 23

7.2.4 Підтримування ділянки дії крізь множину систем 23

7.2.5 Підтримування фільтрації 24

7.2.6 Підтримування розповсюдження сповіщень 24

7.2.7 Керівна роль 24

7.2.8 Підтримування прозорості розміщення 25

7.2.9 Підтримування прозорості транзакцій 25
7.2.10 Підтримування прозорості сталості 25
7.2.11 Підтримування прозорості відгуку 26
8 CORBA-підтримування для ODMA 26
8.1 Обчислювальний погляд 26
8.1.1 CORBA-керовані об'єкти 26
8.1.2 Обробляння сповіщень 27
8.1.3 Обробляння зв'язних відгуків 27
8.1.4 Підтримування ділянки дії 27
8.1.5 Підтримування фільтрації 27
8.2 Інжиніринговий погляд 28
8.2.1 Підтримування прозорості доступу 28
8.2.2 Підтримування прозорості розміщення 28
8.2.3 Підтримування CORBA-реалізації 28
8.2.4 CORBA-керовані об'єкти з гетерогенними агентами 28
Додаток А Термінологія OSI-керування 29
Додаток В Функції ODMA 37
В.1 Функція диспетчеризації операцій 37
В.1.1 Інформаційний погляд 37
В.1.2 Обчислювальний погляд 38
В.2 Функція диспетчеризації сповіщень 38
В.2.1 Інформаційний погляд 38
В.2.2 Обчислювальний погляд 39
В.3 Функція нав'язування політики 40
В.3.1 Інформаційний погляд 40
В.3.2 Обчислювальний погляд 40
Додаток С Приклад OSI-керування, що використовує ODP-RM 42
С.1 Погляд предметної області 42
С.2 Інформаційний погляд 42
С.3 Обчислювальний погляд 43
С.4 Інжиніринговий погляд 44
С.5 Залежності між поглядами 44
Додаток D Приклад моніторингу метрики 46
D.1 Визначення об'єктів метрики 46
D.2 Визначення класу відношень 47
D.3 Визначення класів керованих об'єктів 47
D.4 Приклад обчислювальних об'єктів метрики 48

D.5 Приклад класу відношень 48
D.6 Приклад відображення відношень 48
D.7 Приклад класів керованих об'єктів 48
Додаток E Приклади обчислювальних шаблонів 49
E.1 *Обчислювальний шаблон ITU-T Rec. G.851.1* 49
E.2 Приклади використовування обчислювального шаблону 49
Додаток F Приклад специфікації спільності предметних областей 51
F.1 ODP-принципи погляду предметної області 51
F.2 Приклад специфікації спільності предметної області 52
F.2.1 Спільність керування простим підмережним з'єднанням 52
F.2.2 Опис дій спільності 52
Додаток G Термінологія CORBA 54
Додаток H Забезпечення міжмережного обміну CORBA з іншими стандартними протоколами керування 54
H.1 Міжмережний обмін між GDMO/ASN.1 і ODP IDL 55
H.2 Міжмережний обмін між SNMP і ODP IDL 55

НАЦІОНАЛЬНИЙ ВСТУП

Цей стандарт є прийнятий із змінами ISO/IEC 13244:1998 | ITU-T Recommendation X.703 Information technology — Open Distributed Management Architecture (Інформаційні технології. Архітектура відкритого розподіленого керування) / Amd/1:1999 Support using Common Object Request Broker Architecture (CORBA) (Підтримка використання загальної архітектури брокера об'єктних запитів)

Технічний комітет, відповідальний за цей стандарт, — ТК 20 «Інформаційні технології».

Стандарт містить вимоги, які відповідають чинному законодавству.

До стандарту внесено такі редакційні зміни:

— слова «цей європейський стандарт» замінено на «цей стандарт»;

— структурні елементи цього стандарту: «Обкладинку», «Передмову», «Національний вступ», «Вступ» — оформлено згідно з вимогами національної стандартизації України;

— до розділу 2 «Нормативні посилання» долучено переклад назв стандартів українською мовою.

Цей документ (надалі цей стандарт) згармонізований з ISO/IEC 13244:1998 | ITU-T Recommendation X.703 Information technology — Open Distributed Management Architecture/Amd/1:1999 «Support using Common Object Request Broker Architecture (CORBA) (Інформаційні технології. Архітектура відкритого розподіленого керування та підтримка використання загальної архітектури брокера об'єктних запитів).

Ступінь відповідності ISO/IEC 13244:1998 — модифікований стандарт.

До стандарту долучено поправку 1999 року, згідно з якою добавлено розділ 8 та додатки G і H. Крім того, до додатку А добавлено перелік визначень для термінів, запозичених з ISO/IEC-стандартів щодо взаємозв'язку відкритих систем; для цих міжнародних стандартів відсутні згармонізовані ДСТУ.

ДСТУ 4071:2001

НАЦІОНАЛЬНИЙ СТАНДАРТ УКРАЇНИ

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

АРХІТЕКТУРА ВІДКРИТОГО РОЗПОДІЛЕНОГО КЕРУВАННЯ ТА ПІДТРИМКА ЗАГАЛЬНОЇ АРХІТЕКТУРИ БРОКЕРА ОБ'ЄКТНИХ ЗАПИТІВ (CORBA)

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

АРХИТЕКТУРА ОТКРЫТОГО РАСПРЕДЕЛЕННОГО УПРАВЛЕНИЯ И ПОДДЕРЖКА ОБЩЕЙ АРХИТЕКТУРЫ БРОКЕРА ОБЪЕКТНЫХ ЗАПРОСОВ (CORBA)

INFORMATION TECHNOLOGY

OPEN DISTRIBUTED MANAGEMENT ARCHITECTURE AND SUPPORT USING COMMON OBJECT REQUEST BROKER ARCHITECTURE (CORBA)

Чинний від 2004–07–01

1 СФЕРА ЗАСТОСУВАННЯ

У цьому стандарті розглянуто архітектуру відкритого розподіленого керування (Open Distributed Management Architecture — ODMA). ODMA обумовлює архітектуру для специфікації і розробки як безпосередньо відкритого розподіленого застосування, що здійснює керування системами, так і саме керування відкритими розподіленими застосуваннями. Також ODMA обумовлює архітектурне середовище (framework) для розробляння необхідних стандартів. Керування має розподілену природу, що означає:

— розподіл активності керування;

— керування розподіленими застосуваннями;

— керування ресурсами, які потрібно розподілити.

ODMA узгоджується з ODP-RM, тобто у розподіленому середовищі керування OSI-системами ODMA можна використовувати разом з іншими технологіями, що реалізують ODP-принципи. Цей стандарт є базовим документом серії стандартів і рекомендацій, розроблюваних щодо ODMA. На рисунку 1 зображені залежності між цим стандартом та іншими стандартами. До стандартів, які можна розробляти в рамках ODMA, належать такі:

— підтримка ODMA. Ґрунтуючись на загальному середовищі ODMA, ці стандарти дають опис специфічних систем, які підтримують ODMA. Наприклад, тотожні керування OSI-системами і CORBA-підтримка ODMA;

— нотації опису ODMA. Ці складові стандарти формалізують стандартизовані нотації для опису ODP-поглядів на ODMA (наприклад, додаток D). Ці нотації описують в окремих документах ODMA;

— ODMA-функції. Ці складові стандарти описують функції, необхідні для створювання відкритої системи з розподіленим керуванням. Деякі з функцій, наприклад, функції диспетчера операцій і диспетчера повідомлень, розглянуто у цьому стандарті;

Видання офіційне

— міжгалузеві ODMA-функції. Ці складові стандарти описують забезпечування міжмережного обміну різних парадигм, що підтримують ODMA, наприклад, між керуванням OSI-системами і CORBA.

Згідно з рисунком 1, у цьому стандарті розглянуто лише підмножину, яка підтримує ODMA-системи, допускаючи розвиток інших положень. Тому цей стандарт складається з двох частин:

1) загальне середовище (framework). У цій частині ODMA розглянуто як інтерпретацію для цілей керування Еталонною моделлю відкритого розподіленого обробляння. Запроваджено основну термінологію відкритого розподіленого керування. Тут також ідентифіковано засоби опису відкритих застосувань, керованих розподілено;

2) підтримка OSI-керування для ODMA. У цій частині описано підтримку OSI-керування ODMA. Описано взаємозв'язки між сучасними принципами керування OSI-системами і ODMA. Однак цей стандарт розширює сучасні стандарти системного керування, щоб підтримати розподіл активності керування і розподіл керованих ресурсів. Оскільки специфічна інтерпретація стосується OSI-стандартів, то допустимі додаткові обмеження, наприклад, механізми OSI-керування можуть підтримувати лише ряд видів розподіленої прозорості.

У таблиці 1 відбито, які подання є суттєвими і для яких документів. Знак плюс вказує, що документ розглядає дане подання.

Хоча цей стандарт складається з розділів, концепція ODMA всеохопна і її використовують як інтегровану архітектуру для різних парадигм, що підтримують ODMA.

На рисунку 1 фраза «стандарти і специфікації, засновані на ODMA», означає всі стандарти і специфікації, що будуть розроблені з використанням ODMA-стандартів. У прямокутниках із суцільним контуром розміщено ODMA-документи у межах цього стандарту; у прямокутниках з пунктирним контуром — застосування до цього стандарту, у прямокутниках із заокругленням — ODMA-документи поза рамками цього стандарту.

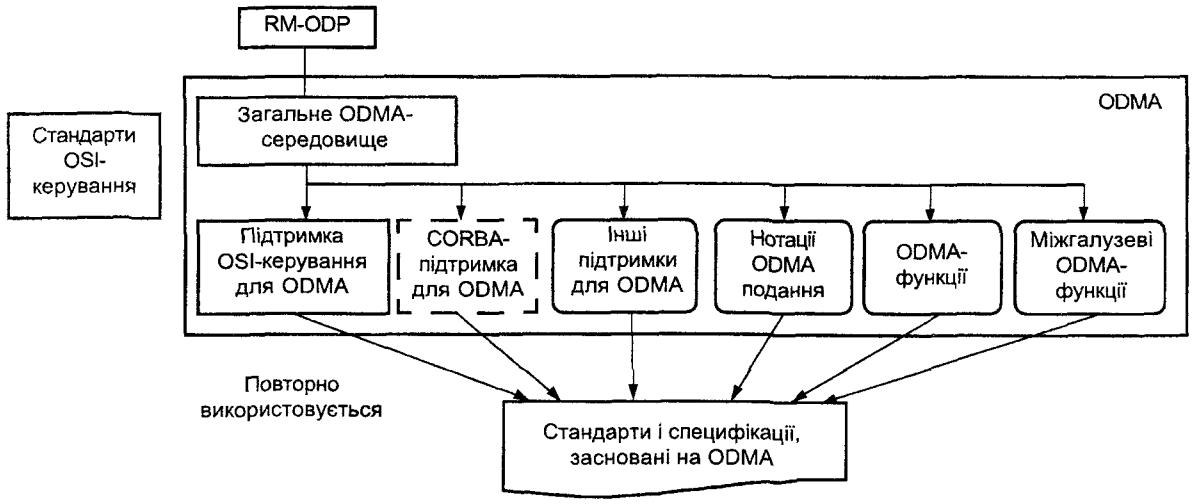


Рисунок 1 — Зв'язки між ODMA-документами

Таблиця 1 — Організація ODMA-документів

Види опису	Загальне середовище	Підтримка OSI-керування	Підтримка CORBA	...	ODMA-функції
Предметна область	+				+
Інформаційний	+				+
Обчислювальний	+	+	+		+
Інжиніринговий	+	+	+		+
Технологічний					

2 НОРМАТИВНІ ПОСИЛАННЯ

У цьому стандарті використано посилання на такі джерела, стандарти[1] і чинні рекомендації ITU-T Бюро стандартизації телекомунікацій ITU (наведено останні публікації документів).

2.1 Ідентичні рекомендації | міжнародні стандарти

— ITU-T Recommendation X.500 (1993) | ISO/IEC 9594-1:1995 Information technology — Open Systems Interconnection — The Directory: Overview of concepts, models and services (Інформаційні технології — Взаємозв'язок відкритих систем — Каталог: Огляд принципів, моделей і сервісу)

— ITU-T Recommendation X.701 (1997) | ISO/IEC 10040:1998 Information technology — Open Systems Interconnection — Systems management overview (Інформаційні технології — Взаємозв'язок відкритих систем — Огляд керування системами)

— ITU-T Recommendation X.702 (1995) | ISO/IEC 11587:1996 Information technology — Open Systems Interconnection — Application context for systems management with transaction processing (Інформаційні технології — Взаємозв'язок відкритих систем — Прикладний контекст для керування системами з обробляннням транзакцій)

— ITU-T Recommendation X.710 (1997) | ISO/IEC 9595:1998 Information technology — Open Systems Interconnection — Common management information service (Інформаційні технології — Взаємозв'язок відкритих систем — Загальний інформаційний сервіс керування)

— CCITT Recommendation X.720 (1992) | ISO/IEC 10165-1:1993 Information technology— Open Systems Interconnection — Structure of management information: Management information model (Інформаційні технології — Взаємозв'язок відкритих систем — Структура керівної інформації: Інформаційна модель керування)

— CCITT Recommendation X.721 (1992) | ISO/IEC 10165-2:1992 Information technology — Open Systems Interconnection — Structure of management information: Definition of management information (Інформаційні технології — Взаємозв'язок відкритих систем — Структура керівної інформації: Визначання керівної інформації)

— CCITT Recommendation X.722 (1992) | ISO/IEC 10165-4:1992 Information technology — Open Systems Interconnection — Structure of management information: Guidelines for the definition of managed objects (Інформаційні технології — Взаємозв'язок відкритих систем — Структура керівної інформації: Настанова щодо визначання керованих об'єктів)

— ITU-T Recommendation X.725 (1995) | ISO/IEC 10165-7:1996 Information technology — Open Systems Interconnection — Structure of management information: General relationship model (Інформаційні технології — Взаємозв'язок відкритих систем — Структура керівної інформації: Узагальнена реляційна модель)

— CCITT Recommendation X.734 (1992) | ISO/IEC 10164-5:1993 Information technology — Open Systems Interconnection — Systems Management: Event report management function (Інформаційні технології — Взаємозв'язок відкритих систем — Системи керування: Функція керування звітами подій)

— CCITT Recommendation X.735 (1992) | ISO/IEC 10164-6:1993 Information technology — Open Systems Interconnection — Systems Management: Log control function (Інформаційні технології — Взаємозв'язок відкритих систем — Системи керування: Функція контролю журналу реєстрації)

— ITU-T Recommendation X.739 (1993) | ISO/IEC 10164-11:1994 Information technology — Open Systems Interconnection — Systems Management: Metric objects and attributes (Інформаційні технології — Взаємозв'язок відкритих систем — Системи керування: Вимірювані об'єкти і атрибути)

— ITU-T Recommendation X.749 (1997) | ISO/IEC 10164-19:1997 Information technology — Open Systems Interconnection — Systems management domain and management policy management function (Інформаційні технології — Взаємозв'язок відкритих систем — Сфера керування системами і функція керування політикою)

— ITU-T Recommendation X.750 (1996) | ISO/IEC 10164-16:1997 Information technology — Open Systems Interconnection — Systems Management: Management knowledge management function (Інформаційні технології — Взаємозв'язок відкритих систем — Системи керування: Функція керування знаннями)

— ITU-T Recommendation X.901 (1997) | ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview (Інформаційні технології — Відкрите розподілене обробляння — Еталонна модель: Огляд)

---

[1] Копії міжнародних стандартів можна отримати у Національному фонді нормативних документів.

— ITU-T Recommendation X.902 (1995) | ISO/IEC 10746-2:1996 Information technology — Open Distributed Processing — Reference model: Foundations (Інформаційні технології — Відкрите розподілене обробляння — Еталонна модель: Основи)

— ITU-T Recommendation X.903 (1995) | ISO/IEC 10746-3:1996 Information technology — Open distributed processing — Reference Model: Architecture (Інформаційні технології — Відкрите розподілене обробляння — Еталонна модель: Архітектура)

— ITU-T Recommendation X.920 (1997) | ISO/IEC 14750:1998 Information technology — Open Distributed Processing — Interface Definition Language (Інформаційні технології — Відкрите розподілене обробляння — Мова опису інтерфейсу)

— ITU-T Recommendation X.950 (1997) | ISO/IEC 13235-1:1998 Information technology — Open distributed processing — Trading function: Specification (Інформаційні технології — Відкрите розподілене обробляння — Функція пошуку компромісів: Специфікація)

— ITU-T Recommendation X.770 (199X) | ISO/IEC 15427-1 Information technology — Open Distributed Management Architecture — Notification Selection and Dispatch Function (Інформаційні технології — Архітектура відкритого розподіленого керування — Відбір повідомлень та функція пересилання).

2.2 Додаткові джерела

— ITU-T Recommendation G.805 (1995) Generic functional architecture of transport networks (Загальна функціональна архітектура транспортних мереж)

— ITU-T Recommendation G.851.1 (1996) Management of the transport network— Application of the RM-ODP framework (Керування транспортною мережею — Застосування RM-ODP середовища)

— ITU-T Recommendation G.852.1 (1996) Management of the transport network— Enterprise viewpoint for simple subnetwork connection management (Керування транспортною мережею — Погляд з рівня підприємства на керування з'єднанням простих підмереж)

— ITU-T Recommendation G.853.2 (1996) Subnetwork connection management information viewpoint (Інформаційний погляд на керування з'єднанням підмереж)

— ITU-T Recommendation M.3100 (1995) Generic network information model (Загальна мережна інформаційна модель)

— ITU-T Recommendation Q.821 (1993) Stage 2 and stage 3 description/or the Q3 interface — Alarm surveillance (Етап 2 і етап 3 опису або Q3-інтерфейс — Спостерігання за подіями).

2.3 Відкриті джерела специфікації

CORBA: The Common Object Request Broker: Architecture and Specification (Загальна архітектура брокера об'єктних запитів. Архітектура і специфікації) Revision 2.1, Object Management Group (Група Керування Об'єктами), August 1997 (OMG Doc Number: Formal/97-09-01)

— CORBA Services; Common Object Services Specification, Object Management Group (Послуги CORBA: Специфікація спільних послуг об'єктів, Група Керування Об'єктами) November 1997 (OMG Doc Number: Formal/97-12-02)

— CORBA Facilities: Common Object Facilities Specification, Object Management Group (Властивості CORBA: Специфікація спільних властивостей об'єктів, Група Керування Об'єктами), Revision 4, November 1995 (OMG Doc Number: Formal/97-06-15).

3 ТЕРМІНИ ТА ВИЗНАЧЕННЯ ПОНЯТЬ

Згідно з ITU-T Rec. X.901 | ISO/IEC 10746-1 термін «об'єкт», додатково не уточнений зворотом «керований», у цьому стандарті означає ODP-об'єкт.

3.1 ODP-RM визначення

У цьому стандарті використано терміни, визначені у ITU-T Rec. X.902 | ISO/IEC 10746-2 (визначення термінів див. у додатку А):

Abstraction	— абстракція;	— абстракция;
Action	— операція, дія;	— операция, действие;
Activity	— активність, дія;	— активность, действие;
Architecture	— архітектура;	— архитектура;
Behaviour	— поведінка;	— поведение;
Binding	— зв'язування, прив'язка (мови);	— связывание, привязка (языка);

Class	— клас	— класс
client object	— клієнтський об'єкт	— клиентский объект
Communication	— взаємодія	— взаимодействие
Compliance	— погодженість узгодженість	— согласованность
Composition	— композиція	— композиция
Configuration	— конфігурація	— конфигурация
Conformance	— відповідність	— соответствие
Contract	— контракт	— контракт
Creation	— створення	— создание
Data	— дані	— данные
Decomposition	— декомпозиція	— декомпозиция
Deletion	— видалення знищення	— удаление
distributed processing	— розподілене обробляння	— распределенная обработка
distribution transparency	— розподілена прозорість	— распределенная прозрачность
Entity	— сутність	— сущность
environment contract	— контракт середовища	— контракт среды
Failure	— збій, відмова	— сбой, отказ
Fault	— помилка, збій	— ошибка, сбой
Identifier	— ідентифікатор	— идентификатор
Information	— інформація	— информация
Instance	— екземпляр примірник	— экземпляр
Interaction	— взаємодія	— взаимодействие
Interface	— інтерфейс	— интерфейс
interface signature	— сигнатура інтерфейсу	— сигнатура интерфейса
Invariant	— інваріант	— инвариант
management information	— керівна інформація	— управляющая информация
Name	— ім'я	— имя
naming action	— операція іменування найменування	— операция именования
Object	— об'єкт	— объект
Obligation	— зобов'язання	— обязательства
ODP system	— ODP-система	— ODP-система
open distributed processing	— відкрите розподілене обробляння	— открытая распределенная обработка
Permission	— припустимий допустимий	— допустимый
Persistence	— усталеність постійність	— постоянство
Policy	— політика	— политика
Portability	— мобільність	— мобильность
Postcondition	— постумова	— постусловие
Precondition	— передумова	— предусловие
Prohibition	— заборона	— запрет
quality of service	— якість сервісу (служби, послуг, обслуговування)	— качество сервиса (службы, услуг, обслуживания)
Role	— роль	— роль
Server object	— серверний об'єкт	— серверный объект
State	— стан	— состояние
System	— система	— система
Type	— тип:	— тип
Viewpoint	— подання, погляд	— представление

У цьому стандарті використано терміни, визначені ITU-T Rec. X.903 | ISO/IEC 10746-3 (визначення термінів див. у додатку А):

Announcement	— оголошення	— объявление
basic engineering object	— базовий інжініринговий об'єкт	— базовый инжиниринговый объект

Binder	— редактор зв'язків	— редактор связей
channel	— канал	— канал
cluster	— кластер	— кластер
communication domain	— сфера (ділянка) комунікації	— сфера коммуникаций
dynamic schema	— динамічна схема	— динамическая схема
engineering interface	— інжиніринговий інтерфейс	— инжиниринговый интерфейс
invariant schema	— інваріантна схема	— инвариантная схема
Node	— вузол	— узел
Operation	— операція	— операция
operation interface signature	— сигнатура інтерфейсу операцій	— сигнатура интерфейса операций
Parameter	— параметр	— параметр
protocol object	— об'єкт протоколу	— объект протокола
reactivation	— реактивація	— реактивация
static schema	— статична схема	— статическая схема
stub	— посередник (заглушка)	— посредник (заглушка)

3.2 Визначення OSI-керування

У цьому стандарті використано термін **трейдер** (trader), визначений у ITU-T Rec. X.950 | ISO/IEC 11235-1. Крім того, використано терміни **агент** (agent), **менеджер** (manager), **клас керованого об'єкта** (managed object class), **MIS-користувач** (MIS-user), визначені в ITU-T Rec. X.701 | ISO/IEC 10040. У стандарті також використано термін **каталог** (directory), визначений у ITU-T Rec. X.500 | ISO/IEC 9594-1. Визначення термінів див. у додатку А

3.3 Додаткові визначення

3.3.1 обчислювальний об'єкт керування *(computational management object)*

Специфічне ім'я ODP-сумісних обчислювальних об'єктів, які підтримують керівний та (або) керований інтерфейс

3.3.2 інжиніринговий об'єкт керування *(engineering management object)*

Специфічне ім'я ODP-сумісних інжинірингових об'єктів, які підтримують керівний та (або) керований інтерфейс.

Національна примітка. Термін «інжиніринговий об'єкт» обрано для уніфікації познак об'єктів, які проектують чи модифікують

3.3.3 операція зв'язного відгуку *(linked reply operation)*

Послідовність операцій між обчислювальними об'єктами керування, які виступають у керівній та керованій ролях. Першу операцію ініціює керівний об'єкт, який перебуває у керівній ролі. Наступні операції ініціюють об'єкт у керованій ролі, а відповідь переспрямовують керівному об'єкту

3.3.4 клієнтський інтерфейс зв'язного відгуку *(linked reply client interface)*

Інтерфейс операцій обчислювального об'єкта керування, який може породжувати множинні відповіді

3.3.5 серверний інтерфейс зв'язного відгуку *(linked reply server interface)*

Інтерфейс операцій обчислювального об'єкта керування, який може приймати множинні відповіді від численних повідомлень операцій керування системами

3.3.6 об'єкт керування *(management object)*

Об'єкт, який може виступати у керівній та(або) керованій ролях

3.3.7 керівна роль *(managed role)*

Поведінка обчислювального об'єкта керування під час виконування операцій керування системами та породженні повідомлень керування системами за взаємодії з іншим обчислювальним об'єктом керування

3.3.8 керована роль *(managing role)*

Поведінка обчислювального об'єкта керування в процесі обробляння повідомлень керування системами й ініціалізації операцій керування системами за взаємодії з іншим обчислювальним об'єктом керування

3.3.9 сповіщення *(notification)*

Взаємодія, коли контракт між об'єктом (клієнтом), який викликає, і приймальним об'єктом (сервером) обмежений здатністю приймати сервером обсяг інформації, відправленої клієнтом

3.3.10 клієнтський інтерфейс сповіщення *(notification client interface)*

Інтерфейс операцій обчислювального об'єкта керування, що може лише породжувати повідомлення керування системами

3.3.11 серверний інтерфейс повідомлення *(notification server interface)*

Інтерфейс операцій обчислювального об'єкта керування, що може лише сприймати сповіщувальні повідомлення керування системами.

Примітка. Терміни клієнт і послуга використовують в ODP-значенні

3.3.12 серверний інтерфейс операцій керування *(management-operation server interface)*

Інтерфейс операцій обчислювального об'єкта керування, що може лише приймати повідомлення операцій керування системами

3.3.13 клієнтський інтерфейс операцій керування *(management-operation client interface)*

Інтерфейс операцій обчислювального об'єкта керування, що може лише породжувати повідомлення операцій керування системами.

4 СКОРОЧЕННЯ

У цьому стандарті використовано такі абревіатури:

ACID (Atomic Consistent Isolated Durable) — атомарна сумісна ізольована стійкість

ACSE (Association Control Service Element) — серверний елемент асоціативного керування

AE (Application Entity) — прикладна сутність

API (Application Programming Interface) — інтерфейс прикладних програм

ASN.1 (Abstract Syntax Notation.1) — нотація.1 абстрактного синтаксису

bmos (base management-operation server) — базовий сервер операцій керування

CMIP (Common Management Information Protocol) — загальний інформаційний протокол керування

CMIS (Common Management Information Service) — загальний інформаційний сервіс керування

CMISE (Common Management Information Service Entity) — сутність загального інформаційного сервісу керування

CORBA (Common Object Request Broker Architecture) — загальна архітектура брокера об'єктних запитів

DII (Dynamic Invocation Interface) — інтерфейс динамічного виклику

GDMO (Guidelines for the Definition of Managed Objects) — керівні вказівки для визначання керівних об'єктів

GIOP (General Inter Orb Protocol) — загальний внутрішній Orb-протокол

GRM (General Relationship Model) — узагальнена реляційна модель

IDL (Interface Definition Language) — мова визначання інтерфейсу

IR (Interface Repository) — репозитарій інтерфейсу

Ir (linked reply) — зв'язний відгук

Ire (linked reply client) — клієнт зв'язного відгуку

Irs (linked reply server) — сервер зв'язного відгуку

JIDM (X/Open-NMF Joint Inter-Domain Management) — X/Open-NMF об'єднане міжгалузеве керування

MOC (Managed Object Class) — клас керованого об'єкта

moc (management-operation client) — клієнт операцій керування

mos (management-operation server) — сервер операцій керування

nc (notification client) — клієнт сповіщення

ns (notification server) — сервер сповіщення

ODMA (Open Distributed Management Architecture) — архітектура відкритого розподіленого керування

ODP (Open Distributed Processing) — відкрите розподілене обробляння

ODP-RM (Reference Model for Open Distributed Processing) — еталонна модель відкритого розподіленого обробляння

OMG (Object Management Group) — група керування об'єктами

ORB (Object Request Broker) — брокер об'єктних запитів

OSI-SM (OSI Systems Management) — керування OSI-системами

Qo (Quality of Service) — якість сервісу (служби, послуги, обслуговування)

RC (Relationship Class) — реляційний клас

RPC (Remote Procedure Call) — виклик віддаленої процедури

SMA (Systems Management Architecture) — архітектура керування системами

SMASE (Systems Management Application Service Element) — сервісний елемент застосування керування системами

SNC (Subnetwork Connection) — підмережне з'єднання

SNMP (Simple Network Management Protocol) — простий мережний протокол керування

TP (Transaction Processing) — обробляння транзакцій

5 ВИМОГИ

У цьому розділі визначений набір вимог, які потрібно виконувати для забезпечування відкритого розподіленого керування. ODMA повинна підтримувати:

— керування ресурсами, зокрема ресурсами, необхідними для самого керування;

— модульність з ідентифікацією частин, які можуть розподілятися;

— делегування обов'язків від одного менеджера до іншого для вихідних операцій керування;

— координацію активності розподіленого керування;

— керування системами будь-якого масштабу;

— моделювання відкритих систем з розподіленим керуванням, з огляду на прозорість розподіляння;

— інструментарій для обраного виду прозорості розподіляння;

— специфічну нотаційну техніку керування для об'єктно-орієнтованого моделювання;

— передачу обов'язків керування від однієї системи до іншої;

— механізми визначання обов'язків керування відповідно до компонентів керованої системи;

— деякі форми прозорості розподіляння, визначені ODP-RM. Не всіх їх використовують для застосувань керування чи базових систем, але всі види прозорості потрібно забезпечувати цими системами;

— прозорість доступу (наприклад, заснована на CMIP чи RPC), що допускає декілька реалізацій специфічних частин одного й того самого застосування у певному застосуванні для міжмережного обміну, реалізованого різними шляхами (які розрізняються API і протоколами взаємодії);

— гарантований міжмережний обмін з наявними OSI-SM-застосуваннями та системами з вказуванням, як саме принципи і нотації OSI-SM можна використовувати для специфікації ODMA-систем і застосувань;

— міжмережний обмін застосувань керування та некерування (наприклад, мережного керування і застосувань інтелектуальних мереж);

— мобільність застосувань керування;

— використовування в ODMA з мінімальною адаптацією наявних інформаційних моделей керування;

— керівні вказівки щодо розробляння нових інформаційних моделей керування, заснованих на ODMA.

Застосування керування, заснованого на ODMA, повинні бути здатними адаптуватися до змін середовища. До них належать, але не обов'язково обмежуються ними:

— внутрішня адміністративна організація: різні адміністративні організації предметних областей мають різні статичні конфігурації функціональності керування. Адміністративні рішення можуть призвести до реконфігурації функціональності керування. За ODMA варто подавати стабільні специфікації застосування керування незалежно від варіантів допустимих організацій керування;

— якість послуг застосування керування: часові обмеження, надійність, обмеження доступу тощо;

— ріст розмірів мережі керування: ODMA повинна підтримувати еволюцію від централізованого керування до розподіленого, оскільки маленькі мережі, збільшуючись, виходять за рамки ефективного керування централізованими системами керування;

— еволюція служб керування: ODMA повинна підтримувати еволюцію наявних служб керування. Специфікації нових служб, якщо це можливо, не повинні посилатися на розміщення наявних служб;

— технологічні зміни: специфікації системи керування повинні бути стабільні щодо зміни технології реалізації.

6 ЗАГАЛЬНЕ СЕРЕДОВИЩЕ

Еталонна модель відкритого розподіленого обробляння є об'єднаним стандартом ISO/ITU, який надає середовище (framework) для масштабних специфікацій гетерогенних розподілених систем. Він визначає архітектуру як набір з п'яти поглядів (уявлень), концентруючись на різних аспектах проблеми розподіленості, і як множину функцій і механізмів прозорості, що підтримують розподіленість. Підсумкове середовище (framework) наповнюється деталізованими стандартами специфічних аспектів конструкцій та операцій розподілених систем. ODMA задає еталонну модель спеціалізованої архітектури для розподіленого керування розподіленими ресурсами, системами і застосуваннями. Наступний опис ODMA фокусується на специфічних особливостях або вимогах керування, які вже не відображаються в ODP-RM. Якщо немає додаткових уточнень, то термін «об'єкт» відноситься до абстракції ODMA-об'єктів, що узгоджуються з ODP-поданням у цьому розділі. Якщо не обумовлене інше, то принципи з цього розділу відповідають зазначеним у ODP-RM принципам.

6.1 Основи

Загальне середовище ODMA ґрунтується на такому базисі:

— обчислювальний об'єкт керування;

— інжиніринговий об'єкт керування;

— керівна роль;

— керована роль;

— серверний інтерфейс операцій керування;

— клієнтський інтерфейс операцій керування;

— клієнтський інтерфейс сповіщень;

— серверний інтерфейс сповіщень.

6.2 Архітектура

Далі кожний ODM-погляд розглядають з використанням принципів ODP-архітектури, необхідних для цілей керування. Додаткові ODMA-принципи, не визначені в ODP, але необхідні для опису керування та перераховані у 6.1, слід розглядати щодо відповідних поглядів.

6.2.1 *Погляд з позицій предметної області*

Погляд з позицій предметної області визначає погляд на систему та її середовище, сфокусований на призначенні, ділянці дії і політиці системи.

ODMA-опис цього погляду не відрізняється від аналогічних описів інших ODP-RM-застосувань. Однак, оскільки особливо цікавими для ODMA є варіанти керованих і керівних ролей, то підвищені вимоги накладаються на збирання даних у реальному масштабі часу.

Специфікація предметної області повинна визначати контракти між об'єктами щодо керованих і керівних ролей. Об'єкт, що виступає у керівній ролі, може вимагати один або більше об'єктів, які виконують керовану роль для здійснення керування, підлеглого конкретному контрактові.

Зараз ODMA не приписує використання конкретної нотаційної техніки для специфікації подання предметної області (тобто нотації предметної області). Однак опис повинен однозначно ідентифікувати (іменувати) різні складові, наприклад, як проілюстровано у додатку F. Такими складовими є:

— контракт;

— роль предметної області;

— спільність;

— політика;

— дія;

— діяльність.

6.2.2 *Інформаційний погляд*

Інформаційний погляд є поглядом на систему та її середовище, сфокусованим на змісті інформації, яка маніпулює системою і її зберігає. Детальний опис можна знайти у документах ODP-RM (розділи 1 і 3).

ODMA-опис інформаційного погляду не відрізняється від аналогічних описів інших ODP-RM-застосувань. Винятком є частіша вимога відповідності між задіяною інформацією і реальними значеннями, що відносяться до устаткування, яке задають інформаційні об'єкти.

Інформаційна специфікація повинна забезпечувати несуперечливість інтерпретації інформації, оброблюваної об'єктом, незалежно від способу розподілу функцій інформаційного обробляння (визначеного щодо обчислювального погляду). Це вимагається специфікацією інваріантної, статичної і динамічної схем.

Інформаційні об'єкти, разом з їхніми зв'язками, задають за допомогою статичної схеми. Твердження засвідчують початковий стан кожного об'єкта у визначений момент часу. Взаємозв'язки між інформаційними об'єктами повинні відбивати інваріантну схему, яка виражає інваріанти.

Динамічну схему використовують для вираження змін інформації з часом і застосовують для специфікації допустимих змін станів інформаційних об'єктів. Специфікація з інформаційного погляду, узгоджуючись з ODP-RM з розділу 3, може охоплювати визначання динамічної схеми як конкретизацію переходів у допустимі стани одного або більше інформаційних об'єктів. На противагу цьому, операції на інтерфейсах, що можуть тригерувати перехід стану, специфікуються щодо обчислювального погляду.

Приклад специфікації статичної схеми з використанням OMT-бази[2)] (Object Modeling Technique — техніка об'єктного моделювання) і поліпшеної об'єктної моделі наведено у додатку С.

6.2.3 *Обчислювальний погляд*

Обчислювальний погляд — це погляд на систему і середовище, що відображає розподіл способом функціональної декомпозиції системи на об'єкти, які взаємодіють через інтерфейс. Детальний опис можна знайти в ODP-RM-документах (розділи 1 і 3).

6.2.3.1 *Специфікація шаблону обчислювального об'єкта керування* охоплює набір обчислювальних інтерфейсів, які можна конкретизувати об'єктом, специфікацією поведінки і специфікацією контракту середовища.

Обчислювальний інтерфейс характеризується сигнатурою, поведінкою і контрактом середовища. Сигнатура операцій інтерфейсу визначає набір операцій, підтримуваних інтерфейсом, та охоплює операції керування системами чи сповіщення, а також роль інтерфейсу (клієнт чи сервер).

Специфікацію поведінки об'єкта задають обмеженнями порядку, синхронізації та розпаралелювання, що застосовують до об'єкта. Цим визначають повну поведінку об'єкта, що обмежується специфікацією поведінки кожного інтерфейсу, підтримуваного об'єктом.

Специфікацію **контракту середовища** для шаблона об'єкта застосовують до об'єкта як неподільної сутності, включаючи підтримуваний інтерфейс. Прикладами аспектів, специфікованих у контракті середовища об'єкта, можуть бути:

— розміщення об'єкта тільки у визначеному домені (обмеження безпеки, обмеження розміщення);

— наявність у об'єкта максимальної імовірності збою (обмеження надійності).

У зв'язку з цим для будь-якого обчислювального об'єкта (охоплюючи об'єкти прив'язки) шаблони повинні специфікувати розглянуті вище наявні елементи.

6.2.3.2 *Обчислювальні інтерфейси керування.* Існує три види обчислювального інтерфейсу керування:

— операції керування;

— сповіщення;

— зв'язні відгуки.

Примітка. Якщо термін **операція** не уточнено додатково, то його використовують у контексті визначання ODP-RM.

[2)] Техніка об'єктного моделювання (OMT — Object Modeling Technique) як метод розроблена J. Rumbaugh.

Обчислювальний інтерфейс керування — це інтерфейс операцій, що може мати одну з таких ролей:

— керівного клієнта: ініціює операції клієнтського інтерфейсу операцій керування;

— керованого сервера: приймає операції від серверного інтерфейсу операцій керування;

— клієнта: ініціює повідомлення на клієнтському інтерфейсі повідомлення;

— керівного сервера: приймає повідомлення від серверного інтерфейсу повідомлень;

— керованого клієнта: ініціює операції над клієнтським інтерфейсом зв'язного відгуку;

— керівного сервера: приймає операції від серверного інтерфейсу зв'язного відгуку.

З метою керування, операції (наприклад, одержати, замінити, операції дії, визначені в GDMO) можуть бути оголошеннями або запитами. Операції, породжені клієнтським інтерфейсом операцій керування, приймає серверний інтерфейс операцій керування. Сповіщення, породжені клієнтським інтерфейсом сповіщень, приймає серверний інтерфейс сповіщень.

З обчислювального погляду інтерфейс специфікує сигнатура інтерфейсу ODP-операцій, яка складається з:

— вказівки ролі інтерфейсу керування;

— сповіщень або сигнатур операцій (ім'я виклику сповіщення чи операції, імена, число і типи параметрів, шаблони дії для можливого завершення).

Примітка. У цьому стандарті параметр використано в ODP-контексті і він не суперечить термінології GDMO.

Приклади нотаційних шаблонів, які можна використовувати для подання інтерфейсу керування, розглянуто у додатку Е.

Обчислювальний об'єкт керування може мати множинні інтерфейси, які керують і якими керують (таблиця 4 і рисунок 2). Для підтримування OSI-керування придатне використання наявних описів класів об'єктів керування для відображення різноцільового інтерфейсу керування; наприклад, один для керування конфігурацією (згідно ITU-T Rec. M.3100) та один для керування збоями (згідно ITU-T Rec. Q.821). Можливість визначати множинні інтерфейси керування спрощує опис взаємозв'язків між різним інтерфейсом об'єкта керування. Отже, для керованої системи існують альтернативи моделювання різноцільової інформації керування за допомогою множинних інтерфейсів керування об'єкта керування, або введення для різних цілей різних об'єктів керування.

Таблиця 4 — Типи обчислювальних інтерфейсів, асоційовані з ODMA-ролями

Роль інтерфейсу	Тип ODMA обчислювального інтерфейсу операцій
Роль керівного клієнта	Клієнтський інтерфейс операцій керування (moc)
Роль керівного сервера	Серверний інтерфейс сповіщень (ns)
Роль керованого клієнта	Клієнтський інтерфейс сповіщень (nc)
Роль керованого сервера	Серверний інтерфейс операцій керування (mos)

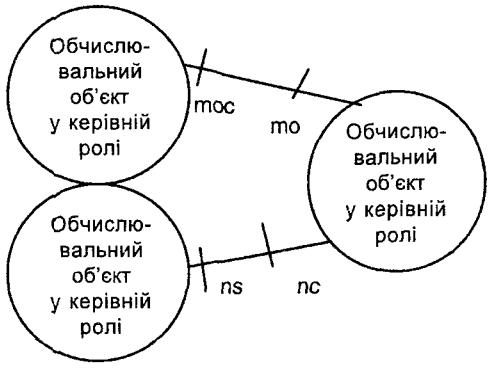


Рисунок 2 — Приклад залежностей між ролями і типами операцій та сповіщень

Примітка. Завдяки природі операцій сповіщення, їх може викликати об'єкт керування на об'єкті розподілу сповіщень, який здатний направляти вміст багатьом адресатам.

Можна використовувати різні нотації визначання обчислювального інтерфейсу, значно оптимізованіші для виразного інтерфейсу, передавання конкретними інжиніринговими об'єктами з протоколу. Це корисно для нейтральних нотацій визначання обчислювального інтерфейсу, що посилаються на інформаційні об'єкти і поведінку, які специфіковано з інформаційного погляду. Такі нейтральні нотації за допомогою трансляції специфікацій можуть відображатися в різні нотації визначення інтерфейсу.

Щодо перспектив керування, необхідно призначати ідентифікатори (однозначні імена) інтерфейсу об'єктів, що виступають у керованій ролі. Іменування інтерфейсу для цих об'єктів у керованій ролі може виявитися необхідним для прив'язки до них. Наприклад, іменування інтерфейсу об'єкта, що виступає у керованій ролі, необхідне, якщо диспетчер сповіщень може доставити подію до такого об'єкта.

6.2.3.2.1 Операція зв'язного відгуку. Розглянемо приклад операцій зв'язного відгуку. Його використовують для негайного повертання доступних даних. З операцією зв'язного відгуку асоційована фінальна відповідь завершування, що засвідчує закінчення операції.

Примітка. Операція зв'язного відгуку — спеціальна операція OSI-керування, що не має відображення в ODP-RM.

Для забезпечування такої можливості ODMA необхідно ввести два нових інтерфейси. У таблиці 5 описано асоціації інтерфейсу зв'язного відгуку з ODMA-ролями. За ODP-RM-принципами ці операції зв'язного відгуку можна промоделювати операціями, до яких потрібно виконати прив'язку. Інтерфейс зв'язного відгуку — це інтерфейс, у якому всі взаємодії є операціями, що відносяться до клієнтського інтерфейсу операцій.

Таблиця 5 — Типи обчислювального інтерфейсу, асоційовані з ролями

Роль інтерфейсу	Тип ODMA обчислювального інтерфейсу операцій
Роль керованого клієнта	Клієнтський інтерфейс зв'язного відгуку (Irc)
Роль керівного сервера	Серверний інтерфейс зв'язного відгуку (Irs)

Інтерфейс зв'язного відгуку сервера в керівній ролі відноситься до інтерфейсу операцій клієнта у керівній ролі, який ініціював дану дію. Обидва належать одному обчислювальному об'єкту у керівній ролі. Інтерфейс операції сервера в керованій ролі належить до Ir-інтерфейсу клієнта в керованій ролі, який реагує на певну дію і належить тому самому обчислювальному об'єктові у керованій ролі.

Зв'язний відгук специфікується у сигнатурі операцій виклику з ідентифікатором множинних операцій.

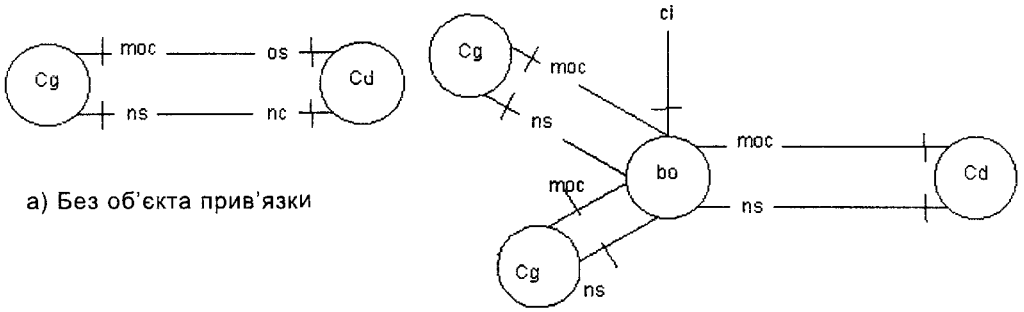
Примітка. Інтерфейс операції може підтримувати операцію для зупинки породження відгуку.

Для всіх операцій із сигнатури інтерфейсу зв'язного відгуку необхідний додатковий параметр для індикації зв'язку з необхідною операцією. На рисунку 3 показано, як працювати з декількома зв'язними відгуками, які використовують ODP-принципи.

Рисунок 3 — Приклад множинних зв'язних відгуків з єдиного об'єкта

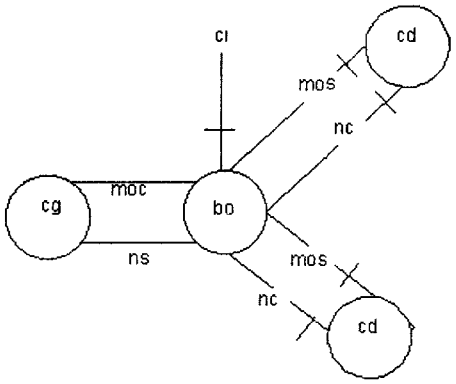
6.2.3.3 *Обчислювальні прив'язки.* Два об'єкти можуть взаємодіяти за допомогою з'єднання, встановлюваного між керівним і керованим інтерфейсом. У найпростішому випадку інтерфейс об'єктів у керівній і керованій ролях зв'язується без об'єктів прив'язки (рисунок 4). Однак іноді бажано ввести об'єкт прив'язки, що дозволяє контролювати зв'язок між об'єктами. Наприклад, у випадку не-

простого двоточкового з'єднання обчислювального інтерфейсу керування необхідний об'єкт прив'язки для контролю зв'язування обчислювальних об'єктів керування. Приклад зв'язування з використанням об'єкта прив'язки наведено у додатку В, де зв'язування інтерфейсу підтримують ODMA-функції диспетчеризації операцій і сповіщень.

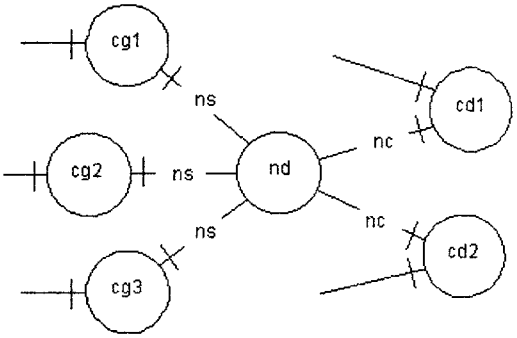


a) Без об'єкта прив'язки

b) Прив'язка до двох обчислювальних об'єктів у керованій ролі



c) Прив'язка до двох обчислювальних об'єктів у керованій ролі



d) Приклад об'єкта прив'язки диспетчера сповіщень

Рисунок 4 — Обчислювальні об'єкти керування з і без об'єктів прив'язки

На рисунку 4 використано позначки: ci — керівний інтерфейс; bo — об'єкт прив'язки; cg — керівний обчислювальний об'єкт; cd — керований обчислювальний об'єкт. Приклади таких ситуацій керування доступом одного чи декількох об'єктів керування:

— у керівній ролі до єдиного керівного об'єкта у керованій ролі (рисунок 4, b);

— у керованій ролі до єдиного об'єкта керування, у керівній ролі (рисунок 4, c).

Диспетчер сповіщень на рисунку 4, d є прикладом специфічного об'єкта прив'язки. Цей об'єкт прив'язки може керувати доступом одного чи декількох об'єктів керування в керівній ролі до одного чи кількох об'єктів керування у керованій ролі.

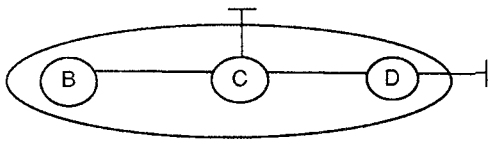


Рисунок 5 — Приклад композиції обчислювального об'єкта

6.2.3.4 *Композиція.* Принципи ODP-композиції використовують для групування ODMA обчислювальних об'єктів керування. Складений об'єкт має множинні інтерфейси керування. Інтерфейс між об'єктами керування усередині складеного об'єкта невидимий зовні. Цей механізм проілюстрований на рисунку 5. Показано три об'єкти керування, згруповані у складений обчислювальний об'єкт. Інтерфейс керування не може поєднуватися чи декомпозуватися з обчислювального погляду. Композиція об'єктів із множинним інтерфейсом може привести до об'єкта з множинними інтерфейсами, за винятком інтерфейсу, що існує між об'єктами в композиції (рисунок 5).

Примітка. Складений об'єкт і його компоненти не співіснують у єдиній обчислювальній специфікації. Навпаки, вони належать до двох різних обчислювальних специфікацій на різних рівнях абстракції.

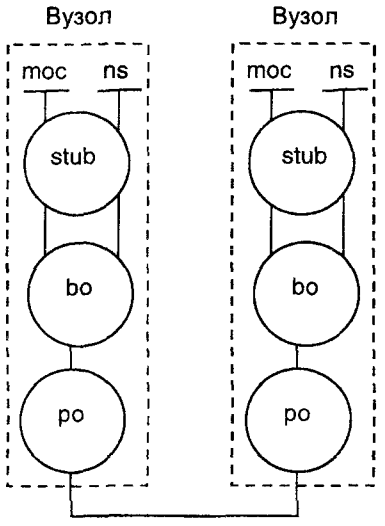


Рисунок 6 — Приклад інжинірингової моделі зв'язування з рисунка 4а

6.2.4 *Інжиніринговий погляд*

Інжиніринговий погляд є поглядом на систему та її середовище, сфокусованим на механізми і функції, необхідні для підтримування розподіленої взаємодії між об'єктами системи. Далі описана функціональність об'єктів, що підтримують прозорість розподіляння. Детальний опис можна знайти в ODP-RM-документах (розділи 1 і 3). Для цілей ODMA загальна модель повинна бути уточнена.

Як приклад використання інжинірингового погляду, на рисунку 6 показана можлива специфікація зв'язування керування, раніше подана на рисунку 4а. Цей приклад акцентує увагу тільки на описі каналів (і не розглядає конфігурації кластерів і капсул) та не залежить від специфіки реалізаційного рішення. У розділі 7, де розглянуто OSI-SM-підтримку ODMA, показано, як можна використовувати OSI-SM для специфікації такого зв'язування керування.

На рисунку 7 подано приклад складнішої системи, яка можливо, використовує різні технології. Елементи у пунктирних прямокутниках задають стек, що складається з stub-посередника, редактора зв'язків і об'єкта протоколу, що детальніше проілюстровано на рисунку 6.

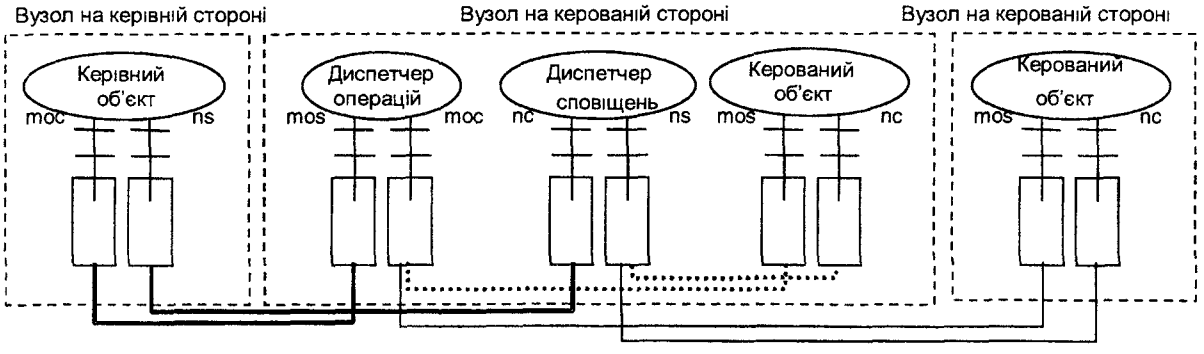


Рисунок 7 — Складніший приклад інжинірингової моделі множинних прив'язок

6.2.4.1 *Підтримування прозорості розподіляння.* Якщо прозорість розміщення і доступу потрібно підтримувати завжди, то прозорість розподіляння можна підтримувати, але не обов'язково у всіх випадках.

Прозорість розміщування маскує місце інтерфейсу в просторі: прозорість розміщення інтерфейсу вимагає приховувати ідентифікаторами інтерфейсу інформацію про місце розташування інтерфейсу. Це також забезпечує прозорість розміщування об'єктів керування.

Інколи може знадобитися знання про розміщування (тобто конкретна адреса для доступу до керованого об'єкта). Передбачено, що інформація про розміщення повинна бути доступною для інших об'єктів.

Прозорість переміщування маскує переміщування інтерфейсу з інших інтерфейсів, з ним пов'язаних. Вона застосовується до об'єктів у кластерах і досягається за допомогою кооперації між менеджерами кластера і редакторами переміщення.

Прозорість міграції маскує від об'єкта здатність функції міграції до зміни місця розташування самого об'єкта. Міграцію об'єкта використовують, наприклад, для:

— мобільних мереж за оптимізації швидкості;

— балансування динамічного завантажування.

Відповідальність за дозвіл міграції лежить на базовій архітектурі.

Прозорість доступу маскує від об'єкта відмінності у механізмах подання даних і виклику. Це дає змогу інтегрувати гетерогенні середовища і використовувати різні технології. У ODP-RM прозорість доступу забезпечують stub-посередники.

Йдеться про те, що ODMA може надати механізми міжмережного обміну між OSI-керуванням і такими парадигмами, як IDL, SQL тощо. Однак повна прозорість доступу неможлива. Замість цього ідентифікація повинна складатися з невеликого набору наявних у конкретний момент видів прозорості доступу, наприклад, GDMO-IDL міжмережний обмін.

Прозорість збою приховує від об'єкта встановлену для нього толерантність до збоїв.

Прозорість сталості приховує від об'єкта використання функцій деактивації і реактивації для зміни ресурсів обробляння, пам'яті і комунікаційних ресурсів, наданих об'єктові.

Прозорість відгуку приховує використання за підтримки інтерфейсу групи сумісних за поведінкою об'єктів.

Прозорість транзакцій приховує координацію дій між конфігураціями об'єктів для досягнення несуперечливості даних.

6.2.5 *Технологічний погляд*

Технологічний погляд виражає, як саме реалізовані специфікації ODP-систем. Він має справу з програмним та апаратним забезпеченням, інсталяціями тощо. Технологічний погляд стосується верифікації реалізованих систем на відповідність стандартним специфікаціям. Технологічний вибір апаратного і програмного забезпечення перебуває поза межами розгляду ODMA.

ODMA надає керівні принципи для погодженості операторів специфікацій стандартів у ODMA і специфікацій систем, що відповідають ODMA.

Примітка. Погодженість ODMA розглянуто далі.

6.3 Багаторазове використовування ODMA-специфікацій

ODP-подання, вжиті для опису специфікацій і стандартів, відносяться до специфічної предметної області. Ці предметні області можуть перекриватися, як показано на рисунку 8. Це може призвести до ситуації, коли об'єкти з обчислювального погляду на предметну область будуть визначені раніше stub-посередниками, як частина іншої області (що перекривається). Як показано на рисунку 8, може існувати кілька інформаційних поглядів (тобто кілька стандартів, рекомендацій, специфікацій, що узгоджуються з ODMA), кожен з яких належить до специфічної предметної області.

На рисунку 8 подано два stub-посередники, що можуть бути визначені двома ODMA-специфікаціями. Припустимо, що специфікація-1 була розроблена до написання специфікації-2. У цьому випадку та частина обчислювального погляду-2, що перекриває обчислювальний погляд-1, уже визначена в специфікації-1. Специфікація-2 може посилатися на специфікацію-1, а не повторювати її. Отже, частину інформаційного погляду-1 використовують повторно.

У кожному з поглядів можна повторно використовувати ту їхню частину, яка визначена в інших специфікаціях. Крім того, погляд також може посилатися на інші погляди в іншій документації, наприклад, обчислювальний погляд специфікації-1 посилається на інформаційний погляд специфікації-2.

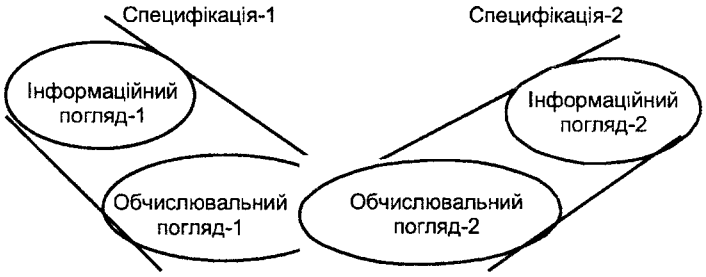


Рисунок 8 — Предметні області, що перекриваються

7 ПІДТРИМУВАННЯ OSI-КЕРУВАННЯ ДЛЯ ODMA

У цьому розділі розглянуто, як саме сучасні принципи керування OSI-системами можна використовувати для підтримування ODMA, тобто розподіляння застосувань керування і розподіл керованих ресурсів.

Підтримка OSI-керування для ODMA є додатковою для архітектури (OSI Systems Management Architecture — SMA). Розглянуто, як SMA можна використовувати у розподіленому середовищі.

Використання ODMA-середовища показує, як можна вбудовувати керування OSI-системами. У тривіальному випадку взаємодії між поодинокою системою, яка приводить до активності керування, й іншою поодинокою системою, в якій локалізовані керовані ресурси, можна застосовувати OSI-керування згідно з ITU-T Rec. X.701 / ISO/IEC 10040. Якщо керівні або керовані застосування чи ресурси розподіляються, то треба використовувати ODMA-архітектуру. Описи подання предметної області й інформаційного погляду на підтримування OSI-керування для ODMA відповідають погляду на предметну область й інформаційному погляду з розділу 6.

7.1 Обчислювальний погляд

У цьому підрозділі показано, як принципи OSI-керування підтримують обчислювальний погляд, розглянутий у розділі 6.

Явна ідентифікація керівної і керованої ролей з обчислювального погляду є розширюванням сучасної архітектури керування OSI-системами. Якщо SMA визначає менеджера й агента як специфічні ролі MIS-користувача, то ODMA визначає керівну і керовану ролі як специфічні ролі обчислювального об'єкта керування. Об'єкт може мати множинні інтерфейси, деякі з них є інтерфейси (OSI) керування.

MIS-користувач залишається принципово необхідним в ODMA; однак існують принципи інжинірингу, які потрібно приховувати з обчислювального погляду. Наприклад, агент (тобто MIS-користувач у керованій ролі) зовсім невидимий з обчислювального погляду у будь-якій формі.

Іншим важливим базовим елементом SMA є керований об'єкт, який видимий з обчислювального погляду. У ODMA керований об'єкт задає серверний інтерфейс операції керування і клієнтський інтерфейс сповіщень (або один з них). Інтерпретація керованого об'єкта, залишається тією самою, тобто об'єкт — це «погляд» менеджера на ресурс. Однак погляд на керований об'єкт є керований інтерфейс до об'єкту. Керований інтерфейс описується з використанням GDMO-шаблону.

Серверний інтерфейс операції керування і клієнтський інтерфейс сповіщень одного керованого об'єкта належать тій самій одиниці розподіляння.

Обчислювальний об'єкт керування може також мати керівний інтерфейс. Сучасна SMA не надає інструментарій для опису керівних інтерфейсів. Однак, оскільки керівний інтерфейс є протилежністю до керованого інтерфейсу, то немає потреби у такому додатковому інструментарії. Для опису залежностей між інтерфейсом сервера і зв'язаного клієнта обчислювального об'єкта керування може використовуватися GRM. Приклад наведено у додатку D.

На рисунку 9 показано приклад чотирьох обчислювальних об'єктів керування, задіяних у системі керування. Об'єкт Ланцюг активізований у трьох відношеннях. У відношенні з об'єктом у керівній ролі об'єкт Ланцюг виступає у керованій ролі. У відношенні з двома об'єктами Гілка об'єкт Ланцюг виконує керівну роль. Обчислювальну поведінку об'єкта керування Ланцюг можна визначити тим, як операції, прийняті керованим інтерфейсом, впливатимуть на операції, що передаються керівним інтерфейсом.

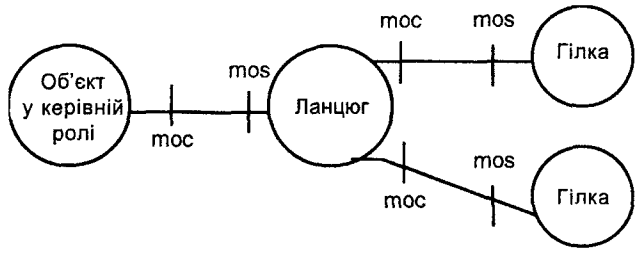


Рисунок 9 —Приклад обчислювального погляду для ODMA

Тут для опису інтерфейсу і поведінки обчислювальних об'єктів керування Ланцюг і Гілка можна використовувати визначання класів GRM-відношень (GRM Relationship Class — RC). Інтерфейс обчислювального об'єкта керування відповідає ролі класу відношень. Так, клас відношень Ланцюг виконує дві ролі, наприклад, сервер операції Ланцюг і клієнт операції Гілка. Клас керованого відношення Гілка виконує одну роль, наприклад, сервер операції Гілка. Поведінка класу відношень описує взаємодію між інтерфейсами (ролями) обчислювальних об'єктів керування (класу відношень), що показано на рисунку 10, причому поведінка RC Ланцюг описує взаємодію між двома інтерфейсами (ролями).

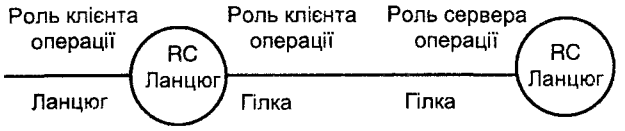


Рисунок 10 — Опис класу відносин обчислювальних об'єктів керування

Отже, інтерфейс обчислювальних об'єктів керування відповідає GRM-ролям. Оскільки ролі треба визначати термінами сумісних класів керованих об'єктів, то інтерфейс обчислювальних об'єктів керування описується термінами класів керованих об'єктів. Визначення керованих класу об'єктів (Managed Object Class — MOC), описує характеристики серверного інтерфейсу операцій керування і клієнтського інтерфейсу сповіщень. Крім того, клас керованих об'єктів можна також використовувати для опису клієнтського інтерфейсу операцій керування і серверного інтерфейсу сповіщень для екземпляра класу іншого об'єкта. У цьому методі класи керованих об'єктів використовують для опису керівної сторони інтерфейсу за допомогою дзеркального відображення серверного інтерфейсу операцій керування і клієнтського інтерфейсу сповіщень. Останній принцип є суттєвим для методів, що використовують GRM разом з GDMO.

Поведінка класу керованих об'єктів усе ще описує тільки поведінку керованої сторони. Нехай є клас керованого об'єкта Інтерфейс Гілки, який використовують для характеризації обох інтерфейсів — сервера і клієнта операції Гілка. Тоді поведінка Інтерфейсу Гілки застосовуватиметься лише до ролі сервера операції Гілка. Асоційована поведінка ролі клієнта операції Гілка описується поведінкою класу відношень Ланцюг, що показано на рисунку 11.

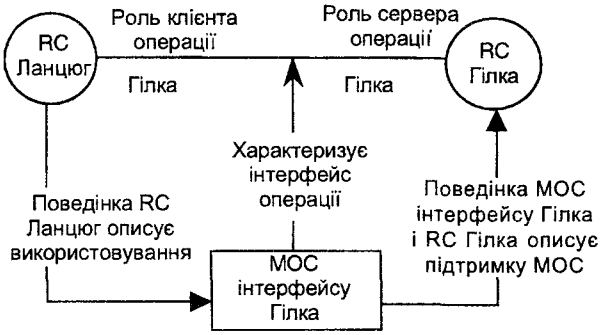


Рисунок 11 — Характеризація класом керованого об'єкта інтерфейсу операцій

Зазначимо, що поведінка MOC Інтерфейсу Гілка, аналогічна поведінці RC Гілка, може описати виконання операцій одержання атрибутів керованого об'єкта Інтерфейс Гілка. Однак поведінка RC Ланцюг може специфікувати лише виклик операції одержання атрибута. Причина в тому, що від подання обчислювального об'єкта керування Ланцюг операція реально виконується на віддаленій стороні інтерфейсу обчислювального об'єкта керування Гілка.

Використання тільки GDMO не дає змогу сформувати повні специфікації обчислювальних інтерфейсів керування. Оскільки не існує єдиної (стандартної) сигнатури CMIS-інтерфейсу операцій, можливі різні обчислювальні реалізації інтерфейсу операцій.

Існує два способи вирішування цієї проблеми: або розширення GDMO з CMIS параметрами (наприклад, для ділянки дії і фільтрації), або поповнення сигнатури інтерфейсу операцій обчислювального інтерфейсу керування, заснованого на GDMO-специфікації (частини) стандартизованою сигнатурою CMIS-операцій.

Примітка 1. Обидві можливості вивчено далі. Рішення особливо важливе для міжмережного обміну з наявними системами OSI-керування.

Отже, GDMO, GRM і CMISE можна використовувати разом для формування повної обчислювальної реалізації. Така реалізація буде оптимізована для роботи з CMIP, що підтримує сферу комунікації.

Примітка 2. Щоб розробити стандартизовану обчислювальну модель, засновану на CMISE, важливо використовувати GDMO і GRM для специфікації обчислювального інтерфейсу. Не йдеться про використання CMIP як протоколу щодо інжинірингового погляду.

Примітка 3. GDMO, GRM і CMISE-специфікації можуть бути успадковані з більш абстрактної обчислювальної нотації, як показано в 6.2 3

7.1.1 *Зв'язні відгуки*

В OSI-керуванні зв'язні відгуки використовують CMIP.

7.1.2 *Рівні обчислювальної абстракції*

Існують різні типи інтерфейсу операцій, можливо, залежні від базових механізмів. Крім того, для сигнатури інтерфейсу, зв'язаного з предметною областю (наприклад, клас керованого об'єкта для OSI-керування), можна ввести додаткові CMIS-параметри. Прикладами можуть служити параметри.

— ділянки дії і фільтрації;

— просування подій.

Ці параметри можуть візуалізуватися щодо обчислювального погляду, якщо:

— існує необхідність підтримування цих параметрів певним узагальненим способом. Наприклад, на рисунку 12 є специфічний об'єкт NW, що розподіляє операцію встановлювання часу на множині адресатів — керованих об'єктів E1, E2 і E3. У цьому випадку немає потреби у обчислювальній візуалізації ділянки дії і фільтрації;

— система може надавати необхідні механізми для підтримування цих параметрів, наприклад, підтримування узагальненої фільтрації й ділянки дії. На рисунку 13 подано обчислювальний аспект ділянки дії і фільтрації за підтримки узагальненого перегляду.

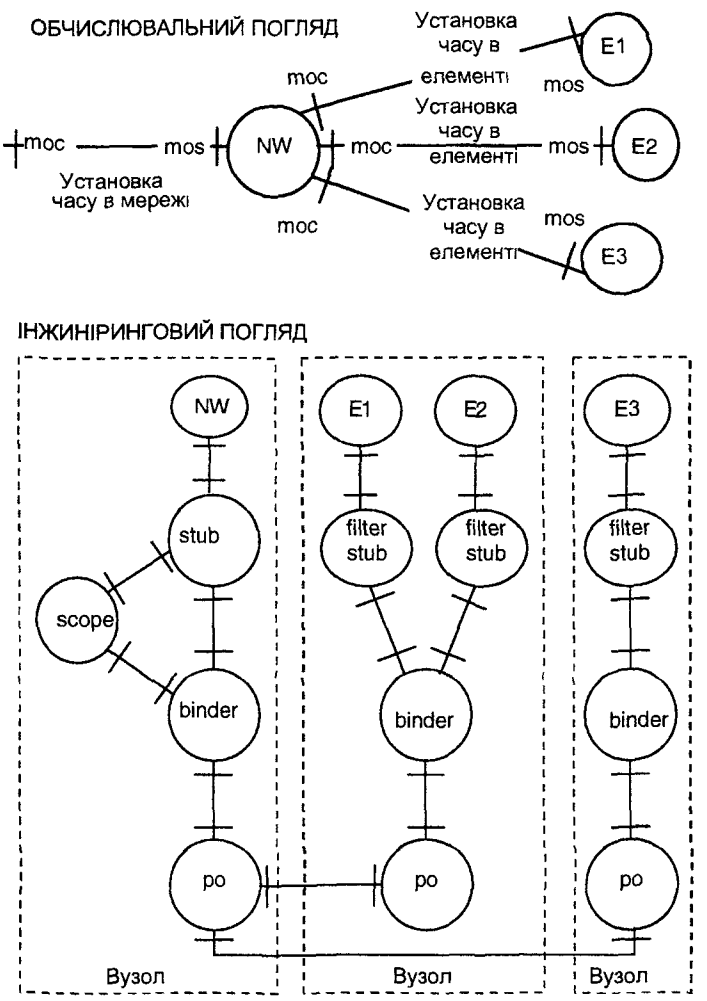


Рисунок 12 — Ділянка дії і фільтрація як інжинірингова оптимізація (без обчислювальної візуалізації)

На рисунках 12 і 13 вжито наступні позначки: F — фільтр; filter stub — заглушка фільтра; binder — редактор зв'язків; po — об'єкт протоколу; scope — контекст.

У верхній частині рисунка 12 показана обчислювальна реалізація, яка задовольняє специфічну предметну область встановлювання часу в мережі. У цій обчислювальній реалізації керівний об'єкт в лівій частині рисунка 12 також передає інтерфейс іншому керівному об'єкту (не показаний на рисунку). До того ж у цій реалізації не відображені параметри ділянки дії і фільтрації, оскільки об'єкти E1, E2, E3 приховані об'єктом NW з обчислювального погляду.

Діаграма обчислювального погляду, розташована у верхній частині рисунка 12, показує мережний об'єкт NW (Network Object), що розподіляє операцію встановлювання часу на множину об'єктів E1, E2 і E3. Цей об'єкт NW є абстрактною нотацією «параметрів ділянки дії і фільтрації». Зазначимо, що ділянка дії і фільтрація — дві різні функції, змодельовані двома інжиніринговими об'єктами: контексту *scope* (для керівного об'єкта) і фільтра (для керованого об'єкта).

Примітка. Для об'єднання фільтрації і керованих об'єктів можна використовувати композицію.

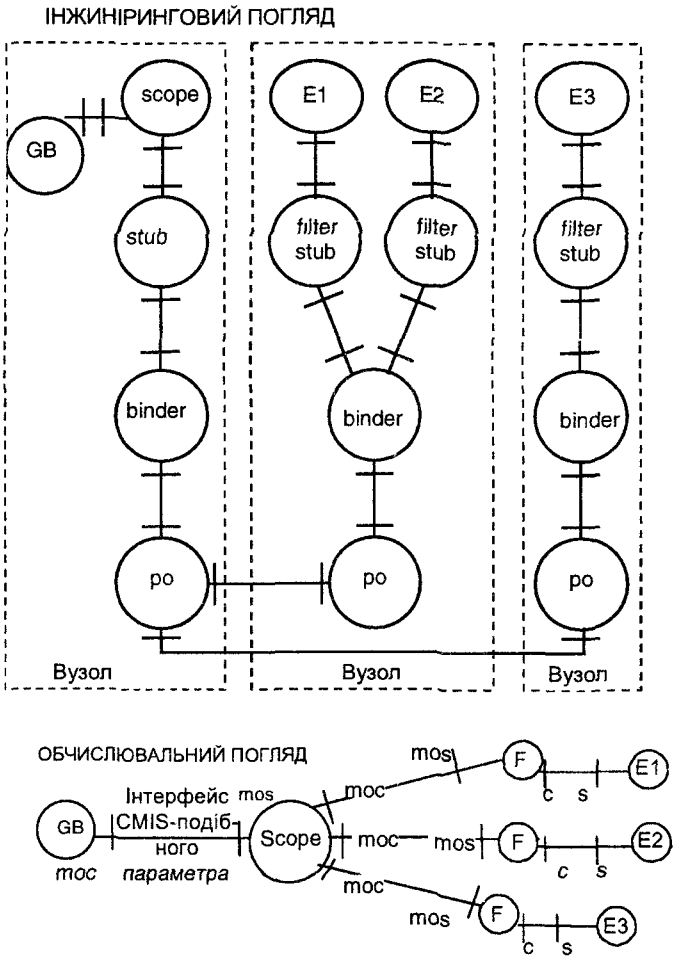


Рисунок 13 — Візуалізована ділянка дії і фільтрація

На рисунку 13 показано обчислювальну реалізацію, яка задовольняє іншу предметну область стосовно можливостей узагальненого браузера. У цьому випадку параметри ділянки дії і фільтрації мають обчислювальне вираження, а кожен об'єкт на рисунку 13 є кандидатом на розподіляння. Діаграма обчислювального погляду, розташована у верхній частині рисунка 13, показує одержування об'єктів узагальненого браузера, що використовує об'єкти ділянки дії і фільтрації.

Оскільки не існує стандартного CMIS-API, то для ділянки дії і фільтрації можна визначити множину сигнатур.

Обчислювальна реалізація, показана на рисунку 12, є прикладом того, як можна специфікувати стандартне рішення ODMA-компонента для однієї предметної області, тоді як на рисунку 13 на-

ведено приклад того, як можна специфікувати стандартне рішення ODMA-компонента для іншої предметної області. Обидві обчислювальні реалізації мають різних кандидатів на розподіляння, тому від них не можна очікувати інтрооперабельності.

Крім ділянки дії і фільтрації, інші CMIS-параметри можуть візуалізуватися з обчислювального погляду, наприклад, як параметри просування події. Допускаємо, що існує обчислювальний об'єкт керування, який може породжувати сповіщення за допомогою одного із своїх клієнтських інтерфейсів. У цьому випадку Дискримінатор просування подій можна використовувати для опису керованого інтерфейсу обчислювального об'єкта керування щоб керувати потоком звітів подій, як показано на рисунку 14.

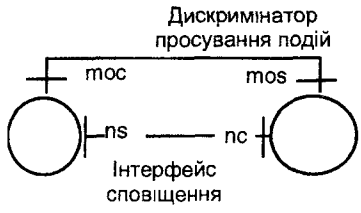


Рисунок 14 — Обчислювальна візуалізація просування подій

Такого роду рисунки викликають припущення, покладені в основу реалізації. У цьому стандарті використано тільки просту сигнатуру, описувану класом керованих об'єктів. Крім того, ODMA створює таку саму абстракцію з керівної сторони. Потрібно абстрагувати від деталей реалізації обчислювального об'єкта керування, зв'язаної за допомогою композиції з кількома механізмами підтримки.

Для стандартизації сигнатур інтерфейсу необхідно розробити додаткові стандарти в рамках ODMA. Наприклад, сигнатури функцій просування подій необхідно визначити для ділянки дії, фільтрації тощо.

7.2 Інжинірінговий погляд

У цьому підрозділі обговорюються, як різні форми прозорості розподіляння підтримуються механізмами OSI-керування.

Спочатку показано, яким чином сучасна архітектура керування системами забезпечує потреби інжинірингу. На рисунку 15 наведено приклад інжинірингу MIS-користувача в ролях менеджера й агента. У цьому випадку два інтерфейси об'єкта інжинірингу з агентної сторони відповідають класу керованого об'єкта. На рисунку 15 показано, що обидва — клієнтський і серверний інтерфейси керування — прив'язані в єдиній асоціації керування до об'єкта протоколу.

Та частина керування OSI-системами, яку необхідно розглядати з позицій відкритого розподіленого керування, є роллю MIS-користувача. Агент завжди ідентифікується у відкритій системі. Ідентифікацію виконують за допомогою AE-заголовка. На рисунку 15 показано, що обидва інтерфейси зв'язуються в єдину асоціацію керування. Для цілей відкритого розподіленого керування агент повинен бути розділений на (керовані) складові. У випадку відкритого розподіленого керування немає єдиного об'єкта *агент*. Як показано на рисунку 15, за функціональні складові агента можна розглядати об'єкти протоколу, stub-посередника і редактора прив'язки. На рисунку 15 прийняті наступні позначки: bo — редактор зв'язків, po — об'єкт протоколу, eo — інжинірінговий об'єкт, eso — інжинірінговий об'єкт підтримки, mg — керівний, md — керований, MOC — клас керованого об'єкта.

Інжинірінгові об'єкти можна визначати, виходячи з багатьох причин, наприклад:

— контроль за сесією керування (ініціалізацією і завершенням);

— створення і знищення керованих об'єктів;

— обробляння операцій, що вимагають контролювання доступу, синхронізації, ділянки дії і фільтрації;

— координація між об'єктами керування;

— поширення повідомлень.

Для керування OSI-системами інжинірінговий погляд описує функціональність, необхідну для комунікації між об'єктом у керівній ролі й об'єктом у керованій ролі, використовуючи CMISE, можливо, разом з іншими протоколами, наприклад, обробляння транзакцій. Крім того, описуються прикладні сутності, що існують між системами розподіленого керування.

Об'єкт протоколу задає стек протоколів, наприклад, 7-рівневий стек OSI-протоколів, що надає обслуговування прикладного рівня.

З редакторами зв'язків асоціюється більшість проблем, пов'язаних з розподілянням. Вони відповідають за наскрізне підтримування цілісності каналу і мають справу зі збоями об'єктів. Тому редактори зв'язків зобов'язані підтримувати зміни конфігурації і комунікацій. Редактор зв'язків пови-

нен встановити прив'язку, якщо канал створений, і простежити кінцевий пункт під час руху об'єкта, заміни об'єкта чи збою. Об'єкт редактора зв'язків задіяний у процесі переміщування об'єкта і забезпеченні більшості з видів прозорості розподіляння, описаних раніше.

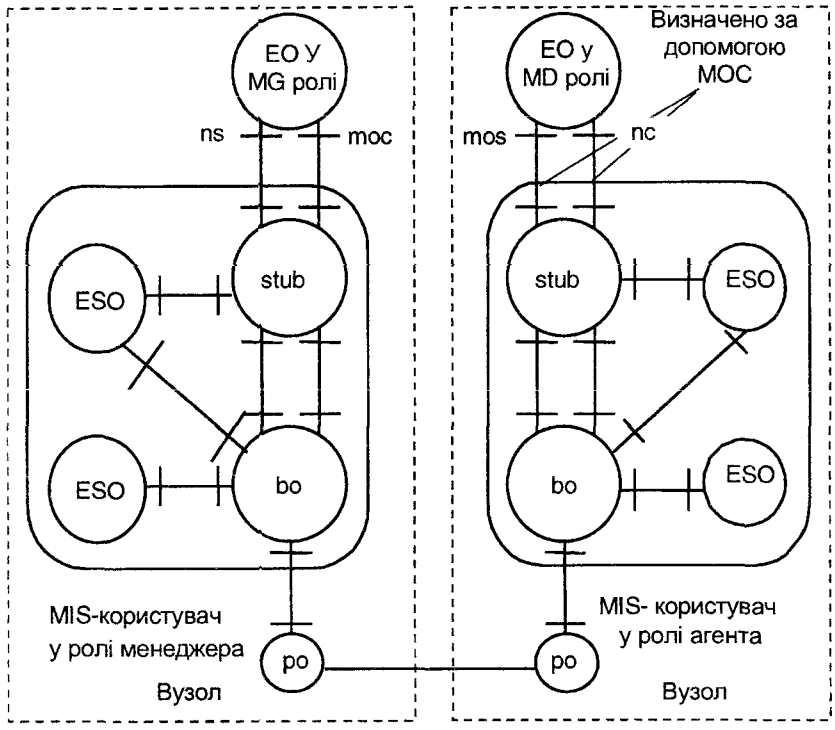


Рисунок 15 — Модель керування OSI-системами (1991) з інжинірингового погляду

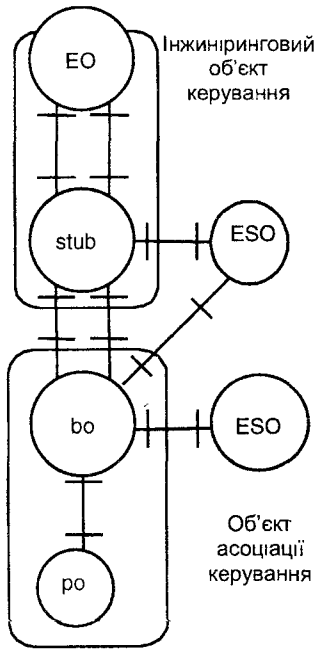


Рисунок 16 — Приклад ODMA базових інжинірингових об'єктів

Керований об'єкт, що має як клієнтський інтерфейс сповіщень, так і серверний інтерфейс операцій керування, повинен у деяких випадках мати можливість відобразити усі взаємодії в єдиній CMIP-асоціації. Таким чином, редактор зв'язків повинен мати змогу відобразити множинні інжинірингові інтерфейси в єдину асоціацію об'єкта протоколу.

Об'єкт stub забезпечує функції адаптації для підтримування взаємодії між інтерфейсами на різних вузлах. Така адаптація може означати трансляцію операцій у кодування, зрозумілу інжиніринговим об'єктам (так зване **упорядкування/розупорядкування** — marshaling/ unmarshalling).

На рисунку 16 наведено складені інжинірингові об'єкти, використані далі в рисунках. Інжиніринговий об'єкт керування є спрощеним варіантом взаємодії між базовим інжиніринговим об'єктом і stub'ом та допустимим варіантом інженерії у випадку, якщо не потрібно прозорості доступу з інжинірингового погляду на обчислювальний об'єкт керування. Об'єкт асоціації керування поєднує функціональність об'єктів протоколу і редактора зв'язків у єдиний інжиніринговий об'єкт, що відноситься до наявної асоціації керування. Це допустимий варіант інжинірингу, якщо немає потреби використовувати множинні протоколи в інжиніринговому погляді на обчислювальний об'єкт керування. Нарешті, інжинірингові об'єкти підтримки надають такі додаткові функції системного керування, як, наприклад, сервер повідомлення.

На рисунку 16 прийнято такі позначки: bo — редактор зв'язків, po — об'єкт протоколу, eo — інжиніринговий об'єкт, ESO — інжиніринговий об'єкт підтримки.

Служби, пропоновані об'єктами асоціації керування (їхнім інтерфейсом), відповідають CMIS-примітивам. Те саме стосується інжинірингових об'єктів підтримки: вони використовують CMIS-примітиви з об'єкта асоціації керування і надають підмножину CMIS-примітивів для інших інжинірингових об'єктів.

Далі детально описано, як підтримують специфічні інжинірингові об'єкти функції керування і прозорості розподілу.

7.2.1 *Створювання і видаляння керованих об'єктів*

Для створювання єдиного керованого об'єкта виконують такі кроки (на рисунку 17 вжито такі позначки: bo — редактор зв'язків, po — об'єкт протоколу, eo — інжиніринговий об'єкт, NM — вузол керування):

1. Керівний об'єкт одержує інформацію (наприклад, AE-заголовок-адресу) на бажаному інтерфейсі (керованого об'єкта), можливо, за допомогою запиту інформації з довідника.

2. Трейдер видає керівний об'єкт з необхідною інформацією (посиланням інтерфейсу) Трейдер розуміється у ODP-RM контексті.

3. Керівний об'єкт направляє запит створювання до відповідного AE.

4. Керування вузлом створює інтерфейс (керованого об'єкта).

Каталог може розташовуватися на іншому вузлі. У цьому випадку необхідний окремий канал зв'язку. ODP-керування вузлом відповідає за створювання і знищування інтерфейсу. Воно існує на кожному вузлі. Деталі див.: ITU-T Rec. X.903/ISO/IEC 10746-3 (12.1).

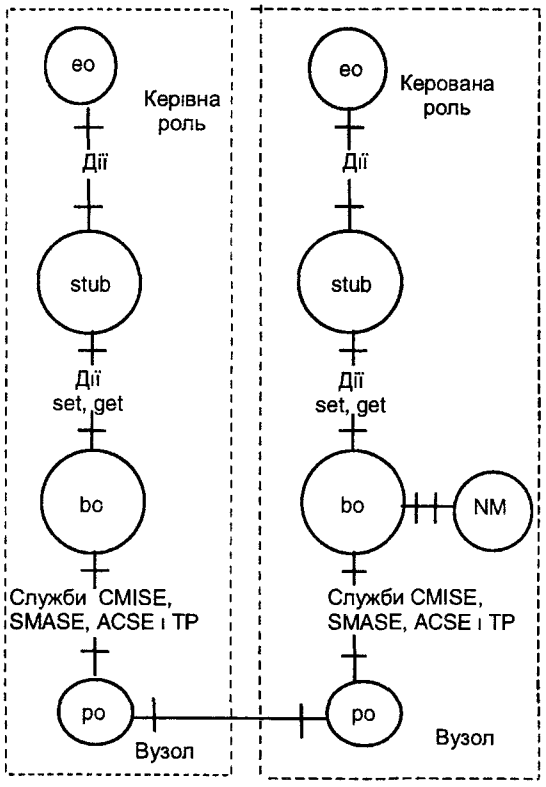


Рисунок 17 — Приклад інжинірингового погляду для підтримки створювання і знищування

7.2.2 *Обробляння запитів операцій*

Керований об'єкт надає операції на атрибутах (GET, REPLACE), діях і сповіщеннях. Частиною функціональності агентів є трансляція CMIS-примітивів у локальне подання цих операцій (сповіщень) на керованих об'єктах, наприклад, відображення M-GET у GET і відображення сповіщень у M-EVENT-REPORT. Природно, цю трансляцію потрібно виконувати з інжинірингового погляду.

Об'єкти у керівній і керованих ролях взаємодіють за допомогою об'єктів протоколів. Їхня комунікація здійснюється не безпосередньо через об'єкт протоколу, а через об'єкт stub з використанням операцій упорядкування (розупорядкування). Об'єкт stub може транслювати виклики операцій з їхніх локальних подань у CMIS-примітиви.

Керівний об'єкт викликає обчислювальні операції керування. Ці виклики транслює stub-посередник з їхнього локального подання в CMIS-примітиви, а на іншому кінці каналу CMIS-примітиви транслюються в локальні подання операцій. Наприклад, об'єкт у керівній ролі викликає операцію GET, що його stub-посередник транслює з локального подання в CMIS M-GET. Stub-посередник на протилежному кінці транслює M-GET у GET, що і є результатом GET для об'єкта у керованій ролі.

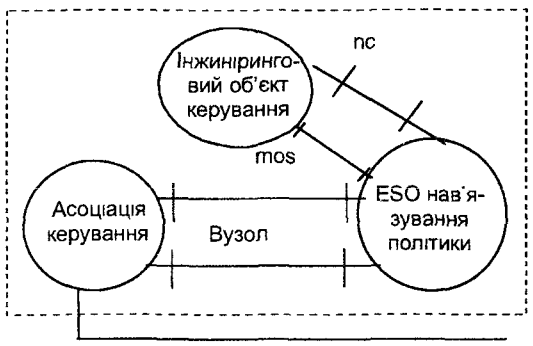


Рисунок 18 — Інжиніринговий об'єкт підтримки нав'язування політики зі сторони здійснення керування

7.2.3 *Нав'язування політики*

Нав'язування політики можна організувати за допомогою використовування допоміжного інжинірингового об'єкта Нав'язування Політики з керованої сторони, який перехоплює всі операції керування і взаємодії сповіщень між усіма контрольованими інжиніринговими об'єктами керування і середовищем, як показано на рисунку 18, де ESO — інжиніринговий об'єкт підтримки.

Примітка. Контроль доступу є окремим випадком нав'язування політики.

7.2.4 *Підтримування ділянки дії крізь множину систем*

Це можливе рішення для підтримування ділянки дії крізь множину систем.

Об'єкти OSI-керування називають за схемою іменування, що використовується шаблонами зв'язування імен, як специфіковано GDMO. У випадку ODMA дерево глобальних імен є найпрактичнішим рішенням для керованих об'єктів. У цьому дереві необхідно застосовувати добір об'єкта з використанням ділянки дії. В ділянці дії ідентифіковано об'єкти, до яких застосовують фільтрацію, наприклад, за допомогою ідентифікації піддерева. Фільтрацію використовують для добору підмножини об'єктів, ідентифікованих ділянкою дії.

ODMA може підтримувати ділянку дії у поєднанні з каталогом. У цьому випадку каталог повинен порціями зберігати інформацію з дерева інформації керування, що доволі стабільно зберігається у каталозі. Функцію керування знаннями надають об'єктам зі складу каталогу, щоб підтримувати таке збереження. Репозитарій повинен підтримувати ділянку дії і повертати список обраних керованих об'єктів. Список керованих об'єктів (посилання на них) скеровується інжиніринговому об'єкту підтримки диспетчера операцій (рисунок 19, де ESO — об'єкт підтримки). Інжиніринговий об'єкт підтримки диспетчера операцій задає функціональність для розподілу операцій (виклику і відгуку) для всіх застосувань AE-заголовків і керованих об'єктів, які з'являються після AE-заголовка.

Схема іменування складається з відношень між класами об'єктів керування. Ці відношення виражаються за допомогою шаблонів імен, релевантних з інформаційного погляду.

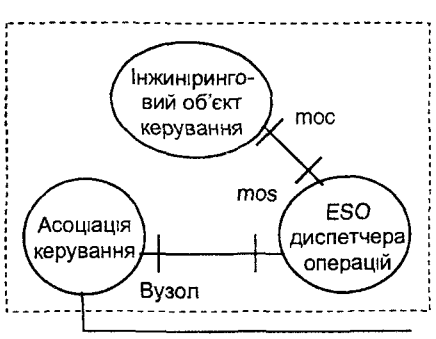


Рисунок 19 — Інжиніринговий об'єкт підтримування диспетчера операцій з керівної сторони

Примітка 1. Довідник можна використовувати як репозитарій для керівних знань. Приклад — об'єкт, що робить видимими керівні знання за допомогою інтерфейсу (керованих об'єктів).

Примітка 2. Об'єкт підтримки диспетчера операцій є потенційною ODMA-функцією. Приклад такого об'єкта розглядається у додатку В.

7.2.5 *Підтримування фільтрації*

Примітка. Фільтрацію вивчають далі.

Фільтрацію потрібно підтримувати керованим інтерфейсом (приклад на рисунку 12).

7.2.6 *Підтримування розповсюдження сповіщень*

Тепер існує два підходи до розповсюдження сповіщень:

— як окрема служба (ODP-функція сповіщення подій), через яку об'єкти можуть підписатися або на відправлення сповіщень (у ролі ресурсу), або на прийом звітів подій у керівній ролі;

— як частина іжинірингового об'єкта, що відноситься до обчислювального об'єкта керування, тобто функція просування події є частиною цього об'єкта.

У першому випадку існує єдина служба, що має інтерфейс керування керованим об'єктом і функціональністю дискримінатора просування подій.

Таку службу надає об'єкт підтримування диспетчера сповіщень. Цей об'єкт використовують для просування і фільтрації подій, які генерують об'єкт, безпосередньо зареєстровані об'єктом підтримки диспетчера сповіщень. Об'єкт, що генерує події, називають **емітером**, а об'єкт, що приймає сповіщення про події, — **реципієнтом**. Зазначимо, що сповіщення від емітера варто розглядати як операції об'єкта підтримування диспетчера сповіщень. Після того, як емітер створений, він може інформувати об'єкт підтримування диспетчера сповіщень про бажання генерувати події визначеного типу. Тоді об'єкт підтримування диспетчера сповіщень надає інтерфейс, здатний прийняти такі події.

Коли реципієнт бажає отримати сповіщення про деякі події, то він інструктує об'єкт підтримування диспетчера сповіщень про необхідність просування таких подій. Це дає змогу множині реципієнтів приймати подію від одного емітера. Об'єкт підтримування диспетчера сповіщень також може бути проінструктований про виконання функцій фільтрації (включно з перевіркою безпеки для санкціонованих реципієнтів) для подій, які приймаються, з метою визначання, які події просувати.

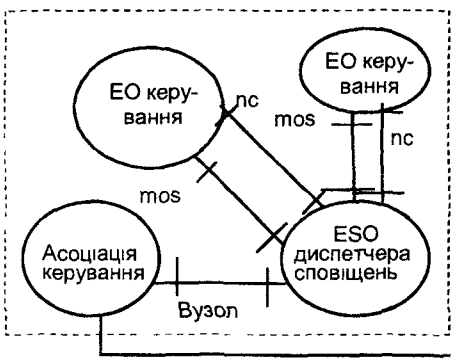


Рисунок 20 — Інжиніринговий погляд на об'єкт підтримування диспетчера сповіщень з керованої сторони

На рисунку 20 наведено іжиніринговий погляд на об'єкт підтримування диспетчера сповіщень щодо об'єктів у керованій ролі. Тут ЕО — проектний об'єкт, ESO — проектний об'єкт підтримування.

Об'єкти у керованій ролі можуть видавати сповіщення, що відправляються об'єкту підтримування диспетчера сповіщень. Через його асоціацію керування об'єкт підтримування диспетчера сповіщень буде переміщувати сповіщення до тих сайтів, у яких розміщені об'єкти у керівній ролі, що підписалися на ці сповіщення. Отже, об'єкт підтримування диспетчера сповіщень має адміністрацію, до якої об'єкти асоціацій керування повинні надсилати сповіщення. Детальніше об'єкт підтримування диспетчера сповіщень щодо ролі керування розглянуто в 7.2.7.

Примітка. Така функціональність може бути зайвою для різних середовищ (транспортних) протоколів.

Інтерфейс об'єкта підтримування диспетчера сповіщень розглядають як функцію звіту подій OSI-керування (ITU-T Rec. X.734 | ISO/IEC 10164-5).

В іншому випадку кожен об'єкт повинен мати власний дискримінатор просування подій у формі керованого об'єкта. Зазначимо, що допускається гібридний підхід.

7.2.7 *Керівна роль*

OSI-керування не робить істотних припущень про характеристики менеджера. У контексті відкритого розподіленого керування менеджера розглядають як об'єкт. Об'єкт у керівній ролі може делегувати відгук іншому об'єктові у керівній ролі для генерації операцій керування та (або) обробляння прийнятих сповіщень.

Об'єкт у керівній ролі позначається локальним ідентифікатором у ділянці дії його АЕ-заголовка. Ці *локальні ідентифікатори адмініструє диспетчер і використовує для адресованих сповіщень і відгуків операцій керування для правильного керівного інтерфейсу.* На рисунку 21 зображено інжиніринговий погляд, де ЕО — інжиніринговий об'єкт, ESO — інжиніринговий об'єкт підтримування.

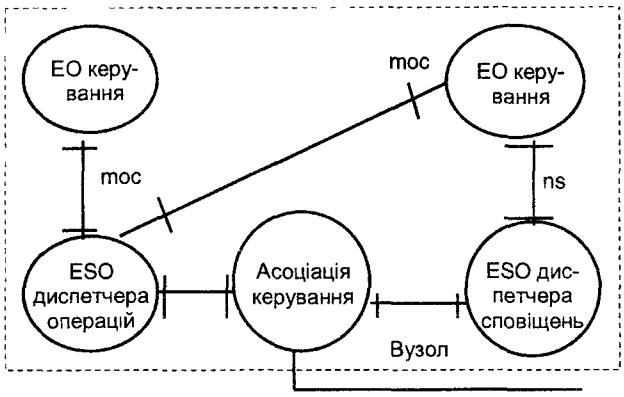


Рисунок 21 — Інжиніринговий погляд на диспетчера керівної сторони

Об'єкт підтримування диспетчера операцій адмініструє операції, які генерує один із клієнтських інтерфейсів операцій керування в його ділянці дії. Об'єкт Асоціацію Керування адресують за допомогою звичайної інформації за адресою (АЕ-заголовок, що задає адресу). Якщо відгук операції керування прийнято, то диспетчер гарантує відправлення відгуку правильному клієнтському інтерфейсові операції керування. Для ідентифікації клієнтського інтерфейсу операції керування диспетчер операцій може використовувати локальний ідентифікатор.

Диспетчера сповіщень використовують для відправляння сповіщень серверним інтерфейсам згідно з підпискою сповіщень. Для цих цілей також використовують локальні ідентифікатори серверного інтерфейсу сповіщень. Об'єкт підтримування диспетчера сповіщень контролює, який серверний інтерфейс підписався і на які сповіщення.

7.2.8 *Підтримування прозорості розміщення*

У OSI-керуванні прозорості розміщення можна досягти у разі глобального іменування, тобто якщо кожен об'єкт керування має унікальне глобальне ім'я, що не залежить від систем, які забезпечують доступ до об'єктів (тобто агентів). Для реалізації прозорості розміщення варто виконати такі дві рекомендації.

— Якщо менеджер знає ім'я об'єкта керування, треба мати змогу одержати адреси прикладних сутностей, що зробить об'єкт керування видимим. Цю вимогу може виконувати функція керування знаннями. Довідник може бути центральною ланкою зберігання інформації, що зв'язує керовані об'єкти з їхніми адресами.

— Керований об'єкт повинен мати унікальне глобальне ім'я. Принцип, за яким іменування керованих об'єктів здійснюється в контексті тільки тих систем керування, де об'єкти видимі, перешкоджає прозорості розміщення. Для керованих об'єктів, яким необхідна гнучкість розміщення на різних системах керування, варто вибрати незалежне від локалізації унікальне глобальне ім'я.

7.2.9 *Підтримування прозорості транзакцій*

Прозорості транзакцій можна досягти в OSI-керуванні у разі використовування контексту застосування керування системами разом з обробляннням транзакцій (ITU-T Rec. X.702 і ISO/IEC 11587).

Примітка 1. Це можна використовувати для зберігання відношень між всіма об'єктами крізь множину систем. ODMA-функції для керування відношеннями вивчають далі.

Примітка 2. Не всі реалізації OSI-керування підтримують обробляння транзакцій.

Один об'єкт у керівній ролі може вибрати множину керованих інтерфейсів для об'єктів у керованій ролі. Об'єкт у керівній ролі може викликати операцію над цим керованим інтерфейсом об'єкта у керованій ролі, що вимагає синхронізації відповідно до ACID-властивостей. Синхронізації можна досягти за допомогою групових викликів. У цьому випадку множинний інтерфейс автоматично доступний для одного менеджера.

7.2.10 *Підтримування прозорості сталості*

Хоча OSI-керування зосереджено переважно на інтерфейсі, прозорість сталості — необхідна і підтримується більшістю наявних реалізацій керованого об'єкта.

7.2.11 *Підтримування прозорості відгуку*

За такої прозорості керовані об'єкти видимі для одного або кількох агентів. Хоча OSI-керування переважно зосереджено на інтерфейсі, але в деяких випадках виникає необхідність у прозорості відгуку. ODMA-функції, що підтримують цю характеристику, вивчаються далі.

8 CORBA-ПІДТРИМУВАННЯ ДЛЯ ODMA

Механізм реалізації ODMA-системи не унікальний і допускає багато підходів. Один з таких підходів є керування OSI-системами, розглянуте у розділі 7. У розділі 8 розглянуто інший підхід, який використовує OMG CORBA як інфраструктуру розподіленого обробляння. Розглянемо, як CORBA може використовуватися для підтримування ODMA.

Примітка. Всесвітньо відомий консорціум OMG здійснює величезний внесок у розвиток інформаційних технологій. ODP IDL згідно з ITU-T Rec. X 920 і ISO/IEC 14750 технічно зіставний з OMG IDL згідно з CORBA.

Засновані на CORBA технології мають такі ділянки потенційного розвитку.

— Під час проектування CORBA усуває відмінності операційних систем і мов.

— Розвиток систем керування може суттєво підсилити використовування CORBA-служб і CORBA-засобів.

— Поліпшується сумісність систем керування. Як стандартний компонент проміжного забезпечення (middleware), CORBA підтримує мови програмування незалежно від визначення інтерфейсу, стандартні відображення мов і підтримку багатьох постачальників. Публікація інтерфейсу для прикладних функцій, підтримуваних системами керування, в ODP IDL повинна сприяти забезпеченню міжмережного обміну між такими системами керування.

— Заснована на CORBA інфраструктура сприяє розвиткові систем керування зі стандартними компонентами, заснованими винятково на CORBA.

Передбачено (там, де це необхідно) використовувати знання і специфікації, сформовані в межах традиційного керування OSI-системами. Крім того, використано узагальнену реалізацію, яка для більшості переваг, наданих розподіленим обчисленням об'єктів загалом і CORBA зокрема, буде розвивати і поліпшувати характеристики систем, розроблених у конкретному контексті.

Опис CORBA-підтримки ODMA з поглядом на предметну область й інформаційний погляд відповідають описові у розділі 6.

8.1 Обчислювальний погляд

Функції, введені ODMA, повинні доповнити більш загальні CORBA-служби і CORBA-засоби. Повний набір специфікацій служб складає фундамент для побудови середовища (framework) розроблення розподілених систем керування телекомунікаціями.

8.1.1 *CORBA-керовані об'єкти*

Тут термін «CORBA-керований об'єкт» позначає обчислювальний об'єкт у керованій ролі, реалізований CORBA-середовищем.

Також необхідно визначити тип погодженого базового серверного інтерфейсу операцій керування (commonly agreed base management-operation server interface type — bmos), який буде підтримувати CORBA-керовані об'єкти. Глобальна підтримка bmos типу інтерфейсу CORBA-керованими об'єктами забезпечує спільність для різних груп служб, визначених для CORBA-відкритого розподіленого керування.

Примітка. Термін «базовий інтерфейс» використовують для позначення того, що інші типи інтерфейсу можуть бути його похідним інтерфейсом, отриманим як результат успадкування або від інших форм типізації.

Кожен CORBA-керований об'єкт має хоча б один інтерфейс, визначений у ODP IDL, для підтримуваних атрибутів і операцій. У цьому контексті тип інтерфейсу bmos повинен містити всі загальні базові характеристики, асоційовані з довільним CORBA-керованим об'єктом. Він уміє включати інформацію про тип інтерфейсу, підтримуваному CORBA-керованим об'єктом, і ідентифікатор екземпляра CORBA-керованого об'єкта. Такий тип інтерфейсу bmos потім повинен адаптуватися до конкретних типів CORBA-керованих об'єктів системи.

ODP IDL пропонує засоби визначення інтерфейсу. Для клієнта і сервера використовують однакові визначення інтерфейсу.

Важливо спланувати розподілені системи керування таким чином, щоб нові типи CORBA-керованих об'єктів (з відповідними новими типами інтерфейсу) могли інсталюватися у систему під час виконання, без перекомпонування системного програмного забезпечування для роботи з новими визначеннями. Динамічний інтерфейс каркасів (The Dynamic Skeleton Interface), що забезпечує подібну гнучкість систем, можна використовувати як такий механізм, але допустимі й інші механізми.

Використовуючи інтерфейс типу bmos CORBA-керованого об'єкта система керування може викликати лише обмежене число операцій без додаткових знань про специфічні характеристики спеціального типу CORBA-керованого об'єкта.

8.1.2 *Обробляння сповіщень*

Сповіщення ODMA обробляють як операційні виклики з сутностей керування у механізм розподілу подій [який засновано на розширенні CORBA-служби подій (CORBA-служби)]. Кожне конкретне ODMA-сповіщення приводить до специфікації множини операцій у серверному інтерфейсі сповіщень. Цей серверний інтерфейс сповіщень може підтримуватися об'єктами прив'язки диспетчера сповіщень (наприклад, CORBA-канал події) чи об'єктами адресатів.

Обробляння ODMA-сповіщень у CORBA-середовищі (як визначено у специфікації CORBA-служби) можна виконувати з використанням як типізованих або нетипізованих у CORBA подій, так і моделей доставляння звітів подій з проштовхуванням або виштовхуванням.

У CORBA-службі подій за моделлю:

— з проштовхуванням (push model) відправник звіту події викликає операцію об'єкта-приймача (з контекстом звіту події, що міститься у виклику);

— за моделлю з виштовхуванням (pull model) приймач звіту події викликає операцію об'єкта-відправника (з контекстом звіту події в кінцевому сповіщенні).

Визначення ODMA-повідомлень і всі рисунки цього стандарту допускають використовування під час доставляння сповіщень моделі з проштовхуванням. Якщо використовують модель з виштовхуванням, ролі сервера і клієнта сповіщення міняються місцями; цей принцип не знайшов явного відображення у цьому стандарті.

Необхідний CORBA-механізм розподілу подій (наприклад, ITU-T Rec. X.770 I ISO/IEC 15427-1), що забезпечить доставляння сповіщень від одиничного CORBA-керованого об'єкта до множинних обчислювальних об'єктів-адресатів, що підписалися на конкретний тип сповіщень.

Визначання ODMA-сповіщень, що допускає розподіл на множинах адресатах, не передбачає іншої інформації, крім підтвердження доставляння, що повертається у відгуку до операції доставляння сповіщення, у випадку моделі з проштовхуванням (або від CORBA-керованого об'єкта до об'єкта розподілу події, або від об'єкта розподілу події до адресата).

Примітка. Модель доставляння з виштовхуванням робить підтвердження доставляння надлишковим з позицій приймача.

Необхідним є механізм розподілу подій, який можна сконфігурувати для різних рівнів якісних характеристик послуг, зокрема можливість обчислювального об'єкта щодо розподіляння подій відкладати у чергу сповіщення для наступного доставляння, коли адресат стане доступний.

8.1.3 *Обробляння зв'язних відгуків*

У CORBA зв'язкові відгуки реалізуються клієнтом вихідної операції (тобто обчислювальним об'єктом у керівній ролі), що забезпечує вхідні параметри операції з сигнатури, яка дає посилання на Irs-інтерфейс, який далі вживають для виклику зв'язного відгуку.

Примітка. Цей принцип іноді називається зворотним викликом (callback).

8.1.4 *Підтримування ділянки дії*

Множинний доступ до об'єктів через ділянку дії може оброблятися або безпосередньо CORBA-керованим об'єктом (через операції одного зі своїх серверних інтерфейсів операцій керування), або механізмом ділянки дії (наявний в іншого об'єкта, поза CORBA-керованим об'єктом), який є узагальненим інтерфейсом керування, що підтримує параметри ділянки дії.

8.1.5 *Підтримування фільтрації*

Обробляння фільтрації підтримується безпосередньо CORBA-керованим об'єктом (через параметри операцій одного зі своїх серверних інтерфейсів операції керування), або використовуючи спеціальні об'єкти фільтрів, розміщені між обчислювальним об'єктом у керівній ролі і CORBA-керованим об'єктом.

8.2 Інжиніринговий погляд

8.2.1 *Підтримування прозорості доступу*

У ODMA інтелектуальність застосувань керування розподіляється між керівною і керованою системами. Керівна система забезпечує функції керування за допомогою використання об'єктів у керівній ролі, що ініціюють операції, які зрештою впливають на об'єкти у керованій ролі і локалізовані в системах керування. Якщо керівна система спирається на відповідне використання stub-посередників і динамічний інтерфейс викликів (Dynamic Invocation Interface — DII) у разі доступу до служб керованих систем, то немає потреби в нових версіях керівної системи у випадку модифікації мережі.

Щоб уникнути модифікацій керівної системи у разі введення нового інтерфейсу до керованої системи, керівній системі слід використовувати CORBA-репозитарій інтерфейсу (CORBA Interface Repository — IR) для дослідження визначень інтерфейсу і CORBA DII для виклику нових або розширених служб.

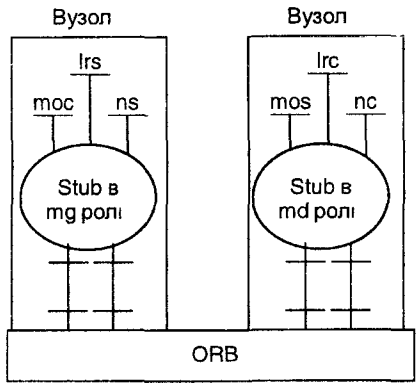


Рисунок 22 — CORBA інжиніринг, забезпечений ORB

8.2.2 *Підтримування прозорості розміщення*

На рисунку 22 показано, як CORBA підтримує прозорість розміщення, забезпечувану ORB (тут mg — керівний, md — керований):

— stub у керівній ролі для клієнтського інтерфейсу операцій керування, серверного інтерфейсу зв'язного відгуку і серверного інтерфейсу сповіщень;

— stub у керованій ролі для серверного інтерфейсу операцій керування, клієнтського інтерфейсу зв'язного відгуку і клієнтського інтерфейсу сповіщень;

— брокер об'єктних запитів (Object Request Broker — ORB), що забезпечує прозорість.

Це означає, що CORBA-керовані об'єкти можуть розміщуватися в різних адресних просторах (або лише в одному).

8.2.3 *Підтримування CORBA-реалізації*

Реалізацію керівної системи іноді можна розглядати як специфічний випадок реалізації більш загальної інформаційної системи.

Природним є припущення проектувальників об'єктно-орієнтованих систем керування про існування сучасних середовищ розробляння застосувань (мов, кодів, баз даних, GUI тощо). CORBA є інфраструктурою, яка підтримує розробляння застосувань у таких середовищах.

CORBA-системам керування можуть знадобитися шлюзи (gateway) для взаємодії із системами, що використовують відмінні парадигми комунікації. Дискусія про підходи до взаємодії шлюзів, що забезпечують міжмережний обмін між різними інфраструктурами комунікацій, подана у додатку Н.

8.2.4 *CORBA-керовані об'єкти з гетерогенними агентами*

Розглянемо, яким чином CORBA-керовані системи забезпечують різний інтерфейс комунікації до систем керування.

Стандарти керування OSI-системами задають інтерфейс між керівною і керованою системами, використовуючи в керованій системі процес Агент, що діє в інтересах керованих об'єктів. На рисунку 23 показаний CMI-доступ до CORBA-керованих об'єктів і об'єктів підтримки з використанням OSI-агента.

На рисунку 23 показана відкрита CORBA-керована система, яка підтримує множинні протоколи керування. Керовані об'єкти, визначені з використанням GDMO і доступні CMIP, можна реалізувати як CORBA-об'єкти. У цьому випадку до таких реалізацій CORBA-керованих об'єктів можливий доступ з використанням CORBA-протоколів взаємодії. Механізми оригінальної інфраструктури можуть аналогічно забезпечити множинну взаємодію з CORBA-об'єктами.

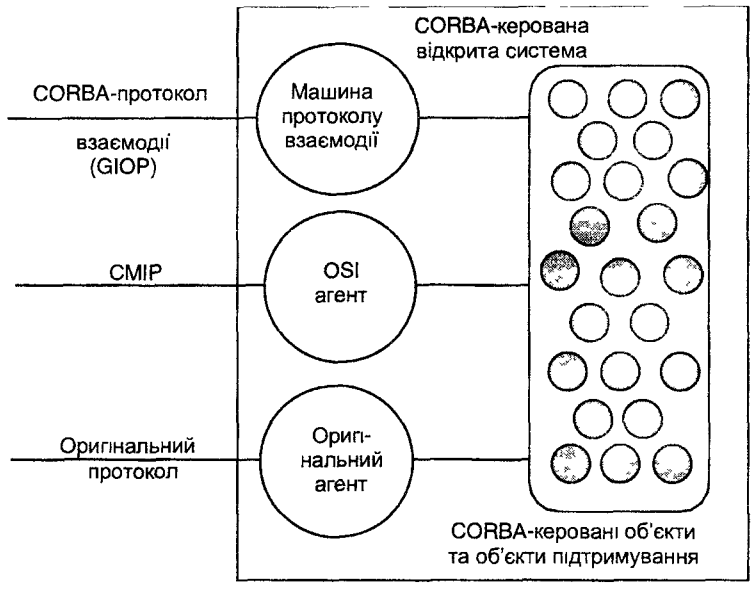


Рисунок 23 — Відкрита CORBA-керована система

ДОДАТОК А

(довідковий)

ТЕРМІНОЛОГІЯ OSI-КЕРУВАННЯ

Подана нижче таблиця використовує термінологію ODMA для опису принципів OSI-керування.

Термін OSI-керування системами	Термінологія ODMA
Керований об'єкт — Managed object	Комбінація серверного інтерфейсу операцій і клієнтського інтерфейсу сповіщень обчислювального об'єкта керування.
Клас керованого об'єкта — Managed object class	Опис серверного інтерфейсу операцій керування і клієнтського інтерфейсу сповіщень, що містить сигнатуру і поведінку.
Агент — Agent	Сукупність інжинірингових об'єктів підтримування, що забезпечують комунікацію з об'єктами в керованій ролі.
Менеджер — Manager	Сукупність інжинірингових об'єктів підтримування, що забезпечують зв'язок з об'єктами в керівній ролі.
Керована система — Managed system	Вузол, що містить інжинірингові об'єкти в керованій ролі.
Керівна система — Managing system	Вузол, що містить інжинірингові об'єкти у керівній ролі.
Сповіщення — Notification	Взаємодія, коли контракт між об'єктом-ініціатором (клієнтом) і об'єктом-приймачем (сервером) обмежується можливістю прийому сервером інформації, відправленої клієнтом.
Операція керування системами — Systems management operation	Операція, виконувана об'єктом у керівній ролі, над об'єктами у керованій ролі.
Атрибут — Attribute	Частина сигнатури інтерфейсу операцій керування (скорочена нотація для всіх операцій доступу до значень атрибутів).
Функція керування системами — Systems management function	Один чи кілька інжинірингових об'єктів підтримування і множина типів обчислювального інтерфейсу керування, які задовольняють вимоги, що логічно відносяться до користувачів

Термін OSI-керування системами	Термінологія ODMA
Сутність CMIP протоколу — CMIP protocol entity	Частина інжинірингового об'єкта протоколу.
Дерево імен — Naming tree	Дерево ідентифікаторів інтерфейсу операцій, використовуваних для цілей ділянки дії
Ділянка дії — Scoping	Ділянка дії є функцією інжинірингового об'єкта, що дає змогу за допомогою одиничного виклику об'єкта з ділянки дії ідентифікувати серверний інтерфейс операцій керування, які поширюють виклик.
Фільтрація — Filtering	Фільтрація є функцією інжинірингового об'єкта, що застосовує до серверного інтерфейсу операцій тестування на допустимість виконання виклику операції над даним інтерфейсом.

Примітка1. У правому стовпчикові термін «операція» використовується з ODP-значенням, що не слід плутати з поняттям операція керування системами.

Примітка 2. Якщо термін «об'єкт» явно не уточнений, то його застосовують як до обчислювального, так і до інжинірингового об'єктів і використовують у ODP-значенні.

Далі наведено визначення термінів, перелік яких вказано у розділі 3 цього стандарту, а самі визначення запозичені з ISO/IEC-стандартів, для яких відсутні згармонізовані ДСТУ.

абстрагування чи **абстракція** (*Abstraction*)

Процес усування несуттєвих деталей для визначання спрощеної моделі або результату цього процесу

дія (*Action*)

Яка-небудь чинність. Кожна дія, яка є цікавою для моделювання, зв'язана, принаймні, з одним об'єктом. У наборі дій, пов'язаних з об'єктом, виділяють внутрішні дії та взаємодії. Внутрішня дія завжди відбувається без участі навколишнього середовища об'єкта. Взаємодія здійснюється за участю навколишнього середовища об'єкта.

Примітка 1. Дія означає виконання дії. Залежно від контексту деталізація може вказати, що дія вже відбулася, зараз відбувається або лише може відбутися.

Примітка 2. Рівень модульності дій залежить від конкретного проекту. Дія може не бути миттєвою. Дії можуть накладатися.

Примітка 3. Взаємодію можна тлумачити у термінах причини та наслідку відношень між об'єктами, що взаємодіють.

Примітка 4. Об'єкт може взаємодіяти сам з собою. У цьому випадку розглянуто, принаймні, дві його ролі у взаємодії. У цьому контексті об'єкт можна розглядати як частину власного навколишнього середовища.

Примітка 5. Залучення навколишнього середовища сприяє спостережності. Таким чином, за взаємодіями можна спостерігати, тоді як за внутрішніми діями не можна спостерігати через інкапсуляцію об'єктів

активність (*Activity*)

Однокорінний, направлений, неперіодичний граф дій, у якому здійснення кожної дії стало можливим завдяки здійсненню всіх безпосередньо попередніх до нього дій (тобто всіма суміжними діями, найближчими до розглянутої)

архітектура (*Architecture*)

Набір правил для визначання структури системи і взаємозв'язків між її частинами

поведінка (*Behavior*) об'єкта

Сукупність дій з набором обмежень на можливість виконання. Конструкціями використовуваної мови визначають обмеження, які таким чином можна виразити. Обмеження можуть містити, наприклад, послідовність, недетермінізм чи паралелізм обмежень реального часу. Поведінка може долучати внутрішні дії. Дії, що фактично наявні, обмежені середовищем, у якому міститься об'єкт.

Примітка 1. Композицію сукупності об'єктів цілком заміняють еквівалентним об'єктом, що задає композицію. Поведінку цього об'єкта часто згадують просто як поведінку сукупності об'єктів.

Примітка 2. Дія й активність — вироджені випадки поведінки.

Примітка 3. Загалом кілька епізодів взаємодії сумісні з певною поведінкою

зв'язування (*Binding*)

Договірний контекст, що випливає з конкретної встановленої поведінки. Встановлення поведінки, договірний контекст і допустима поведінка можуть містити два і більше інтерфейсів об'єкта. Об'єкт, що ініціює встановлення поведінки, може брати або не брати участі у наступній поведінці, що допускається. Допустима поведінка (за аналогією договірний контекст) може бути однорідна (тобто кожен об'єкт, що бере участь, може робити те саме, що й інші) або неоднорідною (тобто роль деякого об'єкта, що бере участь, відрізняється від ролі іншого об'єкта, як клієнт і сервер). Немає жодної потреби у відповідності між об'єктом, який ініціює встановлення поведінки, і приватною роллю в неоднорідній допустимій поведінці (наприклад, у клієнт-серверному договірному контексті, будь-який об'єкт може обґрунтовано ініціювати встановлення поведінки)

клас <X> (*Class of* <X>*'s*)

Набір усіх <X>, що задовольняють тип. Елементи набору співвідносяться як члени класу.

Примітка 1. Клас може не мати членів.
Примітка 2. Чи згодом зміниться розмір набору, залежить від визначання типу

клієнтський об'єкт (*Client object*)

Об'єкт, що запитує виконання функції іншим об'єктом

зв'язок (*Communication*)

Передача інформації між двома чи більше об'єктами в результаті однієї або більше взаємодій, можлива, тільки з деякими проміжними об'єктами.

Примітка 1. Зв'язок можна визначати в термінах причини і наслідку взаємозв'язку між об'єктами-учасниками.
Примітка 2. Кожна взаємодія є прикладом зв'язку

погодженість (*Compliance*)

Вимога необхідної сумісності одного елемента родини ODP-стандартів з іншим (типу RM-ODP), визначена у процесі стандартизації. Підтримка таких вимог зветься погодженістю. Якщо технічні вимоги сумісні, безпосередньо чи опосередковано, з деякими іншими стандартами, то пропозиції, істинні в цих стандартах, — також є істинними в сумісній реалізації технічних вимог

композиція (*Composition*)

1. (об'єктів) — сукупність двох і більше об'єктів призводить до появи нового об'єкта на різних рівнях абстракції. Характеристики нового об'єкта визначені поєднуваними об'єктами і їхнім об'єднанням. Поведінка складеного об'єкта — відповідна композиція поведінки складових об'єктів.

2. (поведінки) — комбінація двох і більше варіантів поведінки призводить до появи нової поведінки. Характеристики підсумкової поведінки визначено поєднуваними варіантами поведінки і їхнім об'єднанням.

Примітка 1. Приклади комбінаційних методів — послідовна і паралельна композиція, черговість, вибір і приховування дії. Ці загальні визначання завжди використовують в певному сенсі, ідентифікуючи специфічні засоби комбінації.
Примітка 2. У деяких випадках композиція поведінки може призводити до появи виродженої поведінки, тобто тупиків через обмеження вихідної поведінки

конфігурація об'єктів (*Configuration of objects*)

Сукупність об'єктів, здатних взаємодіяти за допомогою інтерфейсу. Конфігурація визначає набір об'єктів, задіяних у кожній взаємодії. Специфікації конфігурації можуть бути статичні чи задаватися в термінах динамічних механізмів, що змінюють конфігурацію, тип прив'язки й відв'язки.

Примітка. Конфігурацію можна висловити в термінах паралельної композиції. Процес композиції генерує еквівалентні конфігурації об'єкта на іншому рівні абстракції

контракт (*Contract*)

Угода, керівна частина колективної поведінки набору об'єктів. Контракт визначає зобов'язання, дозвіл і заборону для долучених об'єктів. Технічні вимоги контракту можуть охоплювати:

— технічні вимоги різних ролей, що можуть приймати долучені до контракту об'єкти, і інтерфейсу, зв'язаного з цими ролями;
— якість атрибутів обслуговування;
— свідчення тривалості або періодів законності (обґрунтованість);
— свідчення поведінки, що позбавляє контракт законної сили;
— живучість і умови безпеки.

Примітка 1. Об'єкти в контракті не повинні бути ієрархічно зв'язані, але можуть бути зв'язані на основі одноранговості . Вимоги в контракті не обов'язково однаково застосовні до всіх об'єктів-учасників контракту.

Примітка 2. Контракт можна застосовувати в заданій контрольній точці системи. У цьому випадку він визначає поведінку, яка може очікуватися в контрольній точці.

Примітка 3. Шаблон об'єкта забезпечує простий приклад контракту. Цей шаблон визначає поведінку, загальну для *сукупності об'єктів. Також визначено, що середовище будь-яких таких об'єктів можна брати до уваги, розглядаючи їхню* поведінку. Зазначено, що для часткових технічних вимог шаблон об'єкта залишає невизначеним поведінку об'єкта за деяких станів навколишнього середовища (наприклад, специфічні взаємодії); контракт поширюється лише на зазначену поведінку

створення <X> (*Creation of an* <X>)

Ілюстрація прикладами <X>, що досягнуті діяльністю об'єктів у складі моделі. <X> може бути будь-чим, що може ілюструватися прикладами приватних об'єктів та інтерфейсу.

Якщо <X> — інтерфейс, то він створений для узгодження у процесі створювання певного об'єкта або як додатковий інтерфейс до створюваного об'єкта. Зрештою будь-який інтерфейс повинен бути частиною об'єкта

дані (*Data*)

Форма подання інформації, з якою мають справу системи і їхні користувачі як споживачі інформації

декомпозиція (*Decomposition*)

a) (об'єкта) — детальний опис об'єкта як композиції;

b) (поведінки) — детальний опис поведінки як композиції.

Композиція і декомпозиція — двоїсті терміни і протилежні дії технічних вимог

видалення <X> (*Deletion of an* <X>)

Дія руйнування, проілюстрована прикладами <X>. <X> може бути будь-чим, що можна проілюструвати прикладами приватних об'єктів і інтерфейсу. Якщо <X> — інтерфейс, то його може видалити тільки об'єкт, з яким <X> зв'язаний.

Примітка. Видалення дії не має чіткої мети; дія відбувається

розподілене обробляння (*Distributed processing*)

Обробляння інформації, у якому окремі компоненти можуть бути розташовані в різних місцях, а зв'язок між компонентами може зазнавати затримки і може зазнати невдачі

Прозорість розподілу (*Distribution transparency*)

Властивість приховування від приватного користувача потенційної поведінки деяких частин розподіленої системи.

Примітка. Користувачами можуть бути, наприклад, кінцеві користувачі, прикладні розробники і проектувальники

сутність *(Entity)*

Будь-яка конкретна чи абстрактна річ, яка є цікавою. Загалом слово «сутність» можна використовувати для співвіднесення з будь-чим. У контексті моделювання воно зарезервовано для співставляння з речами у сфері дослідження

контракт середовища *(Environment contract)*

Контракт між об'єктом і його середовищем, зокрема обмеження: якість обслуговування (послуги), використовування і керування. Обмеження якості обслуговування містять:

— тимчасові обмеження (наприклад, eadlines);

— обмеження обсягу (наприклад, пропускна здатність);

— обмеження залежності, які замінюють аспекти приступності (готовності), надійності, надійності в експлуатації, захисту і безпеки (наприклад, середнє напрацювання на відмову).

Обмеження використовування і керування містять:

— обмеження розташування (тобто обране розміщення в просторі та часі);

— обмеження прозорості розподіляння (тобто вибір прозорості розподіляння).

Обмеження якості обслуговування можуть мати на увазі обмеження керування і використання. Наприклад, деяке обмеження якості обслуговування (наприклад, доступність) задоволено для забезпечення однієї і більше розподіленої прозорості (наприклад, дублювання). Обмеження середовища можуть описувати:

— вимоги, що ґрунтуються на середовищі об'єкта для надання правильної поведінки об'єкту;
— обмеження на поведінку об'єкта в «правильному» середовищі

відмова *(Failure)*
Порушення контракту.

Примітка 1. Поведінка, зазначена в контракті, є за визначенням «правильною» поведінкою. Таким чином, відмова — ухиляння від узгодження з правильною поведінкою.

Примітка 2. Шляхи, якими можуть здійснюватися відмови об'єкта, називаються його моделлю відмов. Розрізняють кілька типів відмов:
— випадкові відмови (недотримування технічних вимог як причина більшості відмов);
— відмова через помилку (коли очікувані взаємодії не відбуваються);
— відмови аварійних зупинок (постійні відмови через помилки);
— тимчасові відмови (некоректність через несвоєчасну поведінку).

Примітка 3. Відмова може бути сприйнята по-різному різними об'єктами в середовищі об'єкта, що репрезентує його. Відмова може бути: стійка, якщо будь-яке сприйняття відмови залишається таке саме; нестійка, якщо об'єкти в середовищі можуть по-різному сприймати різні дані відмови

аварія чи **помилка** *(Fault)*
Ситуація, викликана помилкою, що відбулася в об'єкті.

Примітка 1. Аварія, викликана помилкою, може відбутися з часу визначання об'єкта до моменту його руйнування. Аварія на ранньому етапі (наприклад, аварія проекту) не може призводити до відмови у пізнішого періоду (наприклад, час виконання).

Примітка 2. Аварія є активною або бездіяльною. Аварія активна, коли вона робить помилки. Наявність активної аварії визначено тільки виявленням помилок.

Примітка 3. Аварії можуть бути:
— випадкові (виникають чи з'являються випадково) або навмисні (створені навмисно);
— фізичні (через деякі фізичні явища) чи людські (що відбулися через діяльність людини);
— внутрішні (частина стану об'єкта, що може викликати «АБО») чи зовнішні (що відбулися через вплив чи взаємодію із середовищем);
— постійні або тимчасові.

У разі визначання аварії, помилки і відмови йдеться про рекурсивні причинні залежності між аварією, помилкою і відмовою:
— аварія може призвести до помилки (якщо аварія активна);
— помилка може призвести до відмови системи (якщо система не може справитися з аварією);
— відмова відбувається у випадку, коли помилка впливає на коректність служби, наданої системою (чи елементом системи)

ідентифікатор *(Identifier)*
Однозначне ім'я в заданому контексті позначання

інформація (Information)
Будь-яке знáння у сфері досліджування, придатне для обміну серед користувачів, про речі, факти, принципи тощо. Інформація обов'язково має форму подання, щоб її можна було передавати. Першорядно важливим є інтерпретація цього подання (тобто зміст)

екземпляр *типу (Instance)* <X>, що задовольняє тип

взаємодія *(Interaction)*
Щось, що відбувається. Кожна дія, яка є цікавою для моделювання, зв'язана, принаймні, з одним об'єктом. У наборі дій, зв'язаних з об'єктом, виділяють внутрішні дії і взаємодії. Внутрішня дія завжди відбувається без участі навколишнього середовища об'єкта. Взаємодія — за участю навколишнього середовища об'єкта. Див. примітки до «дія»

інтерфейс *(Interface)*
Абстракція поведінки об'єкта, що складається з підмножини взаємодій цього об'єкта разом з набором обмежень на обставини, у яких вони можуть відбуватися. Кожна взаємодія об'єкта відповідає унікальному інтерфейсу. Таким чином, інтерфейс об'єкта формує розділ із взаємодій цього об'єкта.

Примітка 1. Інтерфейс є частиною об'єктної поведінки, отриманої з розгляду тільки взаємодії цього інтерфейсу і приховування всіх інших взаємодій. Приховування взаємодій від іншого інтерфейсу призводитиме загалом до недетермінізму.

Примітка 2. Фраза «інтерфейс між об'єктами» використовують для передавання зв'язування між інтерфейсом об'єктів-учасників

сигнатура інтерфейсу *(Interface signature)*
Набір шаблонів дій, зв'язаних із взаємодіями інтерфейсу. Об'єкт може мати багато видів інтерфейсу з тією самою сигнатурою

інваріант *(Invariant)*
Предикат, на вимоги якого необхідна істинність для всього терміну служби набору об'єктів

інформація керування *(Management information)*
Знання щодо об'єктів, які мають відношення до керування

ім'я *(Name)*
Термін, що відноситься до об'єкта

операція іменування *(Naming action)*
Дія, що співвідносить термін з простору імен з конкретним об'єктом. Усі операції іменування співвідносяться з контекстом іменування

об'єкт *(Object)*
Модель сутності. Об'єкт характеризується поведінкою і станом. Об'єкт відмінний від будь-якого іншого об'єкта. Об'єкт інкапсульований, тобто будь-яка зміна його стану може відбуватися тільки в результаті внутрішньої дії чи взаємодії з навколишнім середовищем.
Об'єкт взаємодіє із середовищем у точках взаємодії.
Залежно від контексту, акцент можна робити на поведінці чи на стані. Коли акцент зроблено на поведінці, об'єкт неформально виконує функції і пропонує послугу (об'єкт, що робить функцію доступною, вважається таким, що пропонує послугу). З метою моделювання ці функції і послуги визначено в термінах поведінки об'єкта та його інтерфейсу. Об'єкт може виконувати більше, ніж одну функцію. Функцію можуть виконувати кілька об'єктів у кооперації.

Примітка 1. Принципи обслуговування і функції використовують неформально для окреслювання цілей стандартизації. У родині ODP-стандартів функція і послуга виражені формально в термінах детального опису поведінки об'єктів та інтерфейсу, що вони підтримують. «Послуга» — приватна абстракція поведінки, що виражає гарантії, запропоновані постачальником послуги.
Примітка 2. Вираз «використання функції» є коротким записом взаємодії з об'єктом, що виконує цю функцію

зобов'язання *(Obligation)*
Розпорядження, яке вимагає специфічної поведінки. Зобов'язання виконують за допомогою виконання запропонованої поведінки

ODP система *(ODP system)*
Система, що відповідає вимогам ODP-стандартів

відкрите розподілене оброблляння *(Open distributed processing)*
Розподілене оброблляння, що відповідає ODP-стандартам

дозвіл *(Permission)*
Розпорядження, яке допускає специфічну поведінку

живучість *(Persistence)*
Властивість об'єкта продовжувати існувати всупереч змінам договірного контексту чи протягом деякого періоду

політика *(Policy)*
Набір правил, зв'язаних зі специфічною метою. Правило може бути виражено як зобов'язання, дозвіл чи заборона.

Примітка. Не всяка політика — обмеження. Деяка політика — дозвіл

мобільність *(Portability)*
Властивість, коли допускається адаптація до ряду конфігурацій об'єкта в контрольних точках.

Примітка. Якщо контрольна точка — програмна, результат може бути вихідним текстом чи підтримкою мобільності Якщо контрольна точка — контрольна точка множинної взаємодії, то результат — мобільність середовища

постумова *(Postcondition)*
Предикат, що вимагає істинності специфікації негайно після виконання дії

передумова *(Precondition)*
Предикат, що вимагає істинності специфікації до виконання дії

заборона *(Prohibition)*
Розпорядження про те, що специфічний режим не повинен виникнути. Заборона еквівалентна зобов'язанню для неприпустимої поведінки

якість обслуговування *(Quality of service)*
Набір вимог якості колективної поведінки одного і більше об'єктів. Якість обслуговування можна визначити в контракті чи виміряти та повідомити після події. Якість обслуговування має бути параметризована.

Примітка. Якість обслуговування зв'язана з такими характеристиками, як швидкість передавання інформації, час очікування, ймовірність переривання передачі, ймовірність системної відмови, ймовірність відмови в пам'яті тощо

роль *(Role)*
Ідентифікатор поведінки, що може виступати як параметр у шаблоні для складеного об'єкта і зв'язаний з одним зі складових об'єктів складеного об'єкта. Детальний опис шаблона як композиції ролей допускає пояснення реалізації процесу як залежності визначеного компонента підсумкового складеного об'єкта від кожної ролі. Залежність складового об'єкта від ролі може випливати після актуалізації параметра

серверний об'єкт *(Server object)*
Об'єкт, що виконує деяку функцію від імені клієнтського об'єкта. Взаємодія клієнта і сервера має різну природу (чи рівень абстракції). Такі відношення можуть існувати між об'єктними чи різними композиціями об'єктів.

Примітка. Неформально сервер забезпечує послугу, викликану клієнтом

стан об'єкта *(State of an object)*
Сукупність чинників об'єкта в конкретний момент часу, що визначає набір всіх епізодів дії, у яких об'єкт може взяти участь.
Загалом поведінка охоплює багато можливих серій дій, у яких об'єкт міг би взяти участь. Знання стану не обов'язково дозволяє пророкувати епізод дії, що фактично відбудеться. Зміни стану є результат дії; отже, стан частково визначений попередніми діями, у яких об'єкт взяв участь. Оскільки об'єкт інкапсульований, то його стан не можна змінити безпосередньо середовищем, але тільки опосередковано в результаті взаємодій, у яких об'єкт бере участь

система *(System)*
Щось, що є цікавим як ціле чи таке, що складається з частин. Тому про систему можна говорити як про об'єкт. Компонентом системи може бути самостійна система. У цьому випадку вона називається підсистемою.

Примітка. Для цілей моделювання концепцію системи розуміємо у загальному системно-теоретичному змісті. Термін «система» може відноситися до системи обробляння інформації, але його можна також застосовувати і загальніше

тип <X> *(Type of an* <X>*)*
Предикат, який характеризує сукупність < X>. Говорять, що <X> має тип чи задовольняє тип, якщо предикат містить цей <X>. Технічно вимоги визначають, який з використовуваних термінів означає тип, тобто <X>. У RM-ODP типи необхідні, принаймні, для об'єктів, інтерфейсу та дій. Поняття типу класифікує сутності категоріально, деякі з них є цікаві як специфікатори

подання *системи (Viewpoint on a system)*
Форма абстракції, що досягла у використанні обраного набору архітектурних принципів і правил структурування, щоб зосередитися на специфічних відношеннях у межах системи

оголошення *(Announcement)*
Взаємодія як звертання, ініційоване клієнтським об'єктом, що призводить до передавання інформації від цього клієнтського об'єкта до серверного об'єкта, і вимагає виконання функції серверним об'єктом

базовий об'єкт інжинірингу *(Basic engineering object)*
Інжиніринговий об'єкт, що вимагає підтримування розподіленої інфраструктури

редактор зв'язків *(Binder)*
Інжиніринговий об'єкт у каналі, що обслуговує розподілену прив'язку між взаємодійними базовими інжиніринговими об'єктами

канал *(Channel)*
Конфігурація посередників (заглушок), редакторів зв'язків, протоколів об'єктів і перехоплювачів, яка забезпечує прив'язку набору інтерфейсу до базових інжинірингових об'єктів, за допомогою взаємодій, у яких вони можуть брати участь.

Примітка. Прив'язки, що вимагають каналів, названі технічною мовою як розподілені зв'язування; прив'язки між інжиніринговими об'єктами, що не вимагають каналів (наприклад, між інжиніринговими об'єктами в тому самому кластері) окреслюють як локальні зв'язування

кластер *(Cluster)*
Конфігурація базових інжинірингових об'єктів, що формують єдиний модуль з метою дезактивації, введення контрольних точок, реактивації, евакуації і переміщення.

Примітка. Сегмент віртуальної пам'яті, що містить об'єкти, — приклад кластера

домен зв'язку *(Communication domain)*
Набір об'єктів протоколу, здатних до міжмережної взаємодії

динамічна схема *(Dynamic schema)*
Технічні вимоги допустимих змін стану одного і більше інформаційних об'єктів.

Примітка 1. Поведінка інформаційної системи може бути змодельована як перехід від однієї статичної схеми до інших, тобто перекласифікація екземплярів від одного типу до іншого.
Примітка 2. Інформаційною мовою зміну стану певного набору об'єктів можна розцінити як взаємодію між цими об'єктами. Не всі об'єкти, залучені до взаємодії, вимагають зміни стану; деякі з об'єктів можуть бути під'єднані способом «тільки для читання»

інваріантна схема *(Invariant schema)*
Набір предикатів для одного і більше інформаційних об'єктів, що завжди повинні бути істинними. Предикати утримують можливі стани і зміни стану об'єктів, до яких вони застосовуються.

Примітка. Таким чином, інваріантна схема — детальний опис типів одного і більше інформаційних об'єктів, яких будуть завжди задовольняти будь-які поведінки поданих об'єктів

вузол *(Node)*
Конфігурація інжинірингових об'єктів, що формує єдиний модуль, щоб розмістити в просторі і реалізувати функції обробляння, зберігання та передавання.

Примітка 1. Комп'ютер і програмне забезпечення (операційна система і застосування) — приклад вузла.
Примітка 2. Вузол може мати внутрішню структуру, що не стосується технічних деталей. Отже, вузол може бути, наприклад, паралельним комп'ютером під керуванням однієї операційної системи

операція *(Operation)*
Взаємодія між клієнтським і серверним об'єктом, що є запитом чи оголошенням

сигнатура інтерфейсу операції *(Operation interface signature)*
Сигнатура інтерфейсу для інтерфейсу операції. Сигнатура інтерфейсу операції містить набір оголошень і сигнатур опитування, оскільки відповідний, один для кожної операції тип інтерфейсу, разом з індикацією причини зв'язку (клієнт чи сервер, але не обоє відразу) для інтерфейсу загалом, щодо об'єкта, який ілюструється шаблоном

об'єкт протоколу *(Protocol object)*
Інжиніринговий об'єкт у каналі, який зв'язується з іншими об'єктами протоколу в тому самому каналі для досягнення взаємодії між базовими інжиніринговими об'єктами (можливо, у різних кластерах, у різних капсулах, а можливо, у різних вузлах)

реактивація *(Reactivation)*
Клонування кластера після його дезактивації

статична схема *(Static schema)*
Технічні вимоги стану одного і більше інформаційних об'єктів у певний вибраний момент часу.

Примітка. Отже, статична схема — технічні вимоги типів одного і більше інформаційних об'єктів у деякий специфічний момент часу. Ці типи — підтипи типів, зазначених в інваріантній схемі

посередник чи заглушка *(Stub)*
Інжиніринговий об'єкт у каналі, що інтерпретує взаємодії, передані каналом, і виконує будь-яке необхідне перетворення чи контроль, ґрунтуючись на цій інтерпретації.

Примітка. Наприклад, посередник може виконувати пошук параметрів у буферах зв'язків чи реєстрацію дії з метою контролювання

об'єкт *(Object)*
Подання сутності в реальному світі. Містить інформацію і пропонує послугу. Система складена з взаємодійних об'єктів. Об'єкт характеризується тим, що робить його відмінним від інших об'єктів, та інкапсуляцією, абстрагуванням і поведінкою

агент *(Agent)*
MIS-користувач, що у конкретній системі керування в результаті взаємодії прийняв роль агента

адміністратор *(Manager)*
MIS-користувач, що у конкретній системі керування в результаті взаємодії прийняв роль адміністратора

клас керованих об'єктів *(Managed object class)*
Заданий набір керованих об'єктів, які спільно використовують ті самі (задані) набори атрибутів, сповіщень, операції керування (пакети) і ті самі умови для наявності цих пакетів

MIS-користувач *(MIS-user)*
Застосування, що використовує кероване системне обслуговування

каталог *(Directory)*
Сукупність відкритих систем, що співпрацюють для безпосереднього забезпечування обслуговування.

ДОДАТОК В
(довідковий)

ФУНКЦІЇ ODMA

Ці функції є спеціальними функціями підтримування розподілу керівної діяльності або керованих ресурсів. Такі функції, як функції диспетчеризації чи операції сповіщень забезпечують прив'язку між множинним обчислювальним інтерфейсом сповіщень. Цю семантику можна уточнювати в обчислювальних об'єктах керування, що виконують узагальнені функції керування. Для диспетчеризації операцій і сповіщень, наступний опис дає один з методів розробляння таких функцій. Вони є абстракціями, які слід уточнити для конкретних парадигм (наприклад, це зроблено для диспетчера операцій і диспетчера сповіщень у розділі 7). Крім того, розглянуто лише інформаційний та обчислювальний погляд.

Примітка. Розглянуто лише схеми ODMA-функцій. Вони можуть відрізнятися від реальних ODMA-функцій.

В.1 Функція диспетчеризації операцій

Функцію диспетчеризації операцій використовують для контролю встановлювання прив'язки між серверним і клієнтським інтерфейсом операцій керування. Функція сприяє динамічним змінам групи серверного інтерфейсу операцій керування.

В.1.1 *Інформаційний погляд*

Розглянемо такі інформаційні об'єкти:

а) запит операції; складається з імені операції, набору параметрів і запитаного набору інтерфейсу адресатів;

b) ім'я операції; є іменем операції, що диспетчеризується;

c) запитуваний набір інтерфейсу адресатів; визначає, які з адресатів можуть прийняти операцію;

d) узагальнена ділянка дії; визначає список посилань керованих ресурсів, який може динамічно змінюватися;

e) максимальний набір інтерфейсу адресатів; становить підмножину узагальненої ділянки дії, що може динамічно змінюватися;

f) посилання керованого ресурсу; є потенційною метою функції диспетчеризації й асоціюється з іменованим набором імен операцій;

g) фактичний набір інтерфейсу адресатів є результатом перетину максимального набору і інтерфейсу адресатів із запитаним набором інтерфейсу адресатів і асоціюється з кожним запитом операції.

Статичну і динамічну схеми можна уточнювати для спеціальних парадигм (наприклад, OSI-SM).

B.1.2 ***Обчислювальний погляд***

В обчислювальному погляді ODMA-функція операції диспетчеризації вкладена в обчислювальний об'єкт прив'язки, який називають **диспетчером операції**. Цей об'єкт обслуговує список серверного інтерфейсу операцій керування, до яких він прив'язаний.

B.1.2.1 *Обчислювальний інтерфейс.* Обчислювальний об'єкт операції диспетчеризації як сервер повинен підтримувати наступні типи інтерфейсу (рисунок B.1):

— інтерфейс запиту операції;

— інтерфейс контролю диспетчера операцій.

Обчислювальний об'єкт Диспетчер операцій повинен обслуговувати для кожного клієнтського об'єкта, прив'язаного до нього, такий тип інтерфейсу — інтерфейс операцій керування.

Інтерфейс запиту операції дозволяє клієнтським ODMA-об'єктам «проштовхувати» операції до інших ODMA-об'єктів. Цей інтерфейс містить одну операцію, яка зветься **запитом операції**. Ця операція має, принаймні, три параметри: перші два відповідно включають ім'я операції та інформацію, що буде передана з операцією. Третій параметр — необов'язковий і містить список адресатів операції.

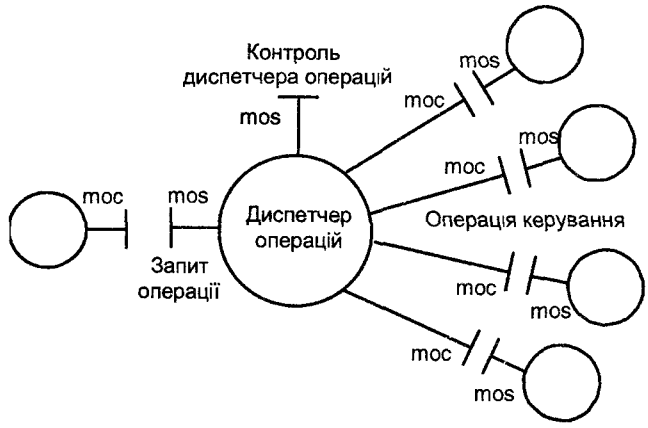


Рисунок B.1 — Обчислювальний погляд операції диспетчеризації

Інтерфейс контролю диспетчера операцій дозволяє іншим об'єктам динамічно маніпулювати набором зв'язаного інтерфейсу операцій керування, який приймають диспетчеризовані операції. Цей інтерфейс повинен також надавати критерії добору імен операцій. Такий інтерфейс містить щонайменше дві операції. Перша операція зміни набору адресатів дає змогу ODMA-об'єкту змінити список адресатів і має єдиний параметр, який подає набір інтерфейсу операцій керування або безпосередньо як список, або опосередковано через селектор. Друга операція зміни інформації про дозволені операції використовується для зміни критерію добору операцій, а новий критерій передається як параметр.

В.1.2.2 *Специфікація поведінки.* Інформацію, яка міститься в запиті ODMA-об'єкта, опосередковано через інтерфейс запиту операцій потрібно доставляти набору адресатів — зв'язаних інтерфейсів операцій керування, що задовольняють критерій диспетчера операцій, які відносяться до конкретного імені операції. Операція керування містить інформацію запиту операції.

В.1.2.3 *Контракт середовища.* Диспетчер операцій повинен підтримувати паралельний доступ.

В.2 Функція диспетчеризації сповіщень

Цю функцію використовують для контролю прив'язки між серверним і клієнтським інтерфейсом сповіщень. Вона сприяє динамічним змінам групи серверного інтерфейсу сповіщень.

Примітка. Розглянута лише спрощена схема, що не дозволяє визначати адресатів, виходячи зі значень параметрів сповіщення. Для простоти, визначання адресата базується на імені сповіщення.

В.2.1 *Інформаційний погляд*

Необхідно описати наступні інформаційні об'єкти:

a) запит сповіщення; складається з імені сповіщення і списку параметрів;

b) ім'я повідомлення, яке будуть диспетчеризувати;

c) узагальнені адресати; визначають набір посилань серверного інтерфейсу сповіщень, який може динамічно змінюватися;

d) максимальний набір інтерфейсу адресатів; становить підмножину узагальнених адресатів і може змінюватися динамічно. Кожен член підмножини асоційований з набором відібраних імен сповіщень;

e) посилання серверного інтерфейсу сповіщень; є потенційною метою функції диспетчеризації й асоціюється з одним іменем сповіщення;

f) фактичний набір інтерфейсу адресатів; асоційований з кожним запитом сповіщення і є результатом фільтрації максимального набору інтерфейсу адресатів з використанням імен сповіщень.

Статична і динамічна схеми можуть деталізуватися для спеціальних парадигм (наприклад, OSI-SM).

В.2.2 *Обчислювальний погляд*

З обчислювального погляду ODMA-функція диспетчеризації сповіщень вкладена в обчислювальний об'єкт прив'язки, що зветься **диспетчером повідомлень**. Цей об'єкт обробляє список серверного інтерфейсу сповіщень, до яких він прив'язаний (рисунок В.2).

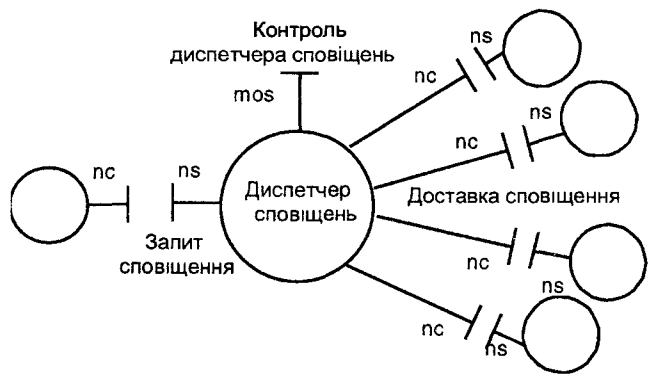


Рисунок В.2 — Обчислювальний погляд диспетчеризації сповіщень

В.2.2.1 *Обчислювальний інтерфейс.* Обчислювальний об'єкт Диспетчер сповіщень як сервер повинен підтримувати такі типи інтерфейсу:

— інтерфейс контролю диспетчера сповіщень;

— інтерфейс запиту сповіщення.

Обчислювальний об'єкт Диспетчер сповіщень як клієнт також повинен підтримувати такий тип інтерфейсу — інтерфейс доставляння сповіщення.

Примітка. Усі ODMA-об'єкти, що є потенційними приймачами такого сповіщення, як сервери повинні підтримувати конкретний інтерфейс доставляння сповіщень.

Інтерфейс доставляння сповіщень (який містить єдину операцію доставляння сповіщення) є клієнтським інтерфейсом сповіщень, який диспетчерує сповіщення. Кожен екземпляр цього інтерфейсу прив'язаний до ODMA-об'єкта, що підтримує як сервер цей інтерфейс .

Інтерфейс запиту сповіщень дозволяє клієнтським ODMA-об'єктам «проштовхувати» сповіщення до набору інших ODMA-об'єктів. Цей інтерфейс містить одну операцію, що зветься **запит сповіщення**. Ця операція має, принаймні, три параметри: перші два — це відповідно імена сповіщень та інформація, яка відправляється разом із сповіщенням. Третій параметр необов'язковий і містить список адресатів сповіщення.

Інтерфейс контролю диспетчера сповіщень дозволяє іншим ODMA-об'єктам динамічно маніпулювати багатьма зв'язаними інтерфейсами доставляння сповіщень, які можуть приймати диспетчеризовані сповіщення. Цей інтерфейс також повинен забезпечувати критерій добору імен сповіщень. Такий інтерфейс містить щонайменше дві операції. Перша операція зміни набору адресатів дозволяє ODMA-об'єктам змінювати список адресатів і має єдиний параметр, який подає набір інтерфейсу доставляння сповіщення або, безпосередньо використовуючи список, або опосередковано через селектор. Другу операцію зміни інформації про дозволене сповіщення, використовують для зміни критерію добору сповіщень, новий критерій передають як параметр.

B.2.2.2 *Специфікація поведінки*

Інформацію, що міститься в запиті ODMA-об'єкта, передану через інтерфейс запиту сповіщень, потрібно доставляти набору адресатів — зв'язаним інтерфейсам доставляння сповіщень, які задовольняють критерій диспетчера сповіщень, що відноситься до імені сповіщення, наданого ODMA-об'єктом, який запитує диспетчеризацію. Операція доставляння сповіщення містить інформацію запиту сповіщення.

B.2.2.3 *Контракт середовища*

Диспетчер сповіщень повинен підтримувати паралельний доступ. Отже, диспетчер сповіщень повинен пропонувати атомарні операції для свого серверного інтерфейсу.

B.3 Функція нав'язування політики

Функція нав'язування політики забезпечує системну політику керування і її використовують для фіксації її порушень. Політику керування розглядають в ITU-T Rec. X.701 | ISO/IEC 10040 і ITU-T Rec. X.749 | ISO/IEC 10164-19.

B.3.1 *Інформаційний погляд*

Необхідно визначити такі інформаційні об'єкти:

a) специфікацію політики; визначає застосовувану політику;

b) звіти про порушення; породжуються за спроби порушити політику.

Крім того, необхідно задати операції і сповіщення, які нав'язує політика.

B.3.2 *Обчислювальний погляд*

Нижче проілюстровано дві можливі обчислювальні реалізації:

— у першій (рисунок B.3) взаємодія між об'єктами в керівній ролі й об'єктами в керованій ролі перехоплюється єдиним обчислювальним об'єктом прив'язки Нав'язування політики;

— у другій (рисунок B.4, де ре — нав'язування політики) кожен об'єкт у керованій ролі зв'язується через окремий обчислювальний об'єкт прив'язки Нав'язування політики.

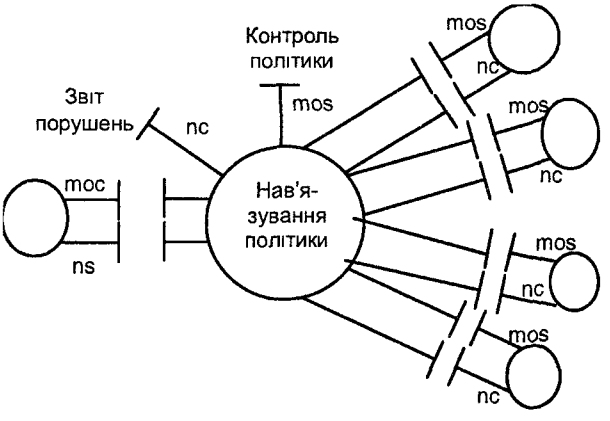


Рисунок B.3 — Обчислювальна реалізація нав'язування політики з єдиним об'єктом Нав'язування політики

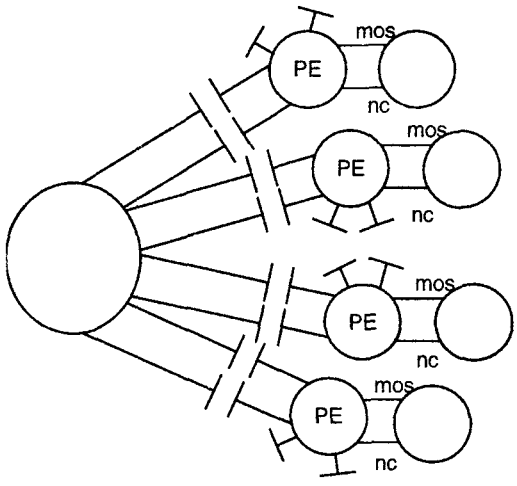


Рисунок В.4 — Обчислювальна реалізація нав'язування політики з набором об'єктів Нав'язування політики

Примітка. Інакше можливе рішення полягає у використанні сутності, незалежної від керованого об'єкта, до якого застосовують політику, що інспектує усі взаємодії з керованими об'єктами, які є суб'єктами політики, та збуджує відповідну реакцію на порушування політики.

В.3.2.1 *Обчислювальний інтерфейс.* Крім звичайного інтерфейсу прив'язки, об'єкт Нав'язування політики повинен підтримувати такий інтерфейс:

— інтерфейс контролю політики, використовуваний для контролю застосовуваної політики;

— сповіщення про порушення (клієнтський інтерфейс сповіщень) використовують для сповіщення про порушення політики.

Відмінності між двома моделями обчислювальних об'єктів керування виражають в термінах специфікації поведінки і контрактів середовища.

В.3.2.2 *Специфікація поведінки*

— Рішення 1 (рисунок В.3). Об'єкт Нав'язування політики використовує у разі перехоплювання взаємодії між об'єктами в керівній ролі і керованій ролі за допомогою єдиного обчислювального об'єкта прив'язки. Таким чином, цей об'єкт прив'язки повинен допускати встановлювання політики і забезпечувати сповіщення про спроби порушення політики. Об'єкт Нав'язування політики повинен допускати перехоплення взаємодії керування з всіма об'єктами — суб'єктами політики.

— Рішення 2 (рисунок В.4). Об'єкт Нав'язування політики використовують у разі перехоплювання взаємодій між об'єктами в керівній ролі й об'єктами в керованій ролі за допомогою набору обчислювальних об'єктів прив'язки, де кожен об'єкт Нав'язування політики прив'язують до єдиного об'єкта в керованій ролі і єдиного об'єкта в керівній ролі.

В.3.2.3 *Контракт середовища*

— Рішення 1 (рисунок В.3). Необхідно забезпечити захист доступу до об'єкта Нав'язування політики.

— Рішення 2 (рисунок В.4). Повинні існувати засоби, які забезпечують погоджене встановлювання політики на усі об'єкти, задіяні до нав'язування цієї політики. Необхідний захист доступу до об'єктів Нав'язування політики.

ДОДАТОК С
(обов'язковий)

ПРИКЛАД OSI-КЕРУВАННЯ, ЩО ВИКОРИСТОВУЄ ODP-RM

У додатку поданий ODP-погляд на опис застосування відкритого розподіленого керування. На прикладі розглядається функція керування системами для контролю журналу реєстрації (опис див. у CCITT Rec.X.735|ISO/IEC 10164-6).

Примітка. Метод, розглянутий у додатку, не є єдиним, що може використовуватися ODMA.

С.1 Погляд предметної області

Текст з погляду предметної області копіює текст з CCITT Rec. X.735|ISO/IEC 10164-6, функції контролю журналу реєстрації.

Для багатьох функцій керування необхідно зберігати інформацію про події, що відбулися, або операції, виконані різними об'єктами. У реальних відкритих системах для зберігання такої інформації виділяються різні ресурси. У OSI-керуванні такі ресурси оформлюють як журнал реєстрацій (log) та реєстраційні записи, що містяться в журналі.

Керування, необхідне для реєстрації інформації, може змінюватися з часом. Крім того, у разі читання такої інформації з журналу реєстрації, менеджер повинен мати змогу визначати, чи записи були загублені, чи змінені.

Отже, можна сформулювати такі вимоги:

a) визначання гнучкої служби контролю журналу реєстрації, що дає змогу вибирати записи, зареєстровані системою керування у приватному журналі реєстрацій;

b) можливість для зовнішніх систем змінювати критерій, використовуваний щодо реєстрації записів;

c) можливість для зовнішніх систем визначати, чи зареєстровані записи були змінені, чи характеристики запису реєстрацій були загублені;

d) специфікації механізму контролю часу реєстрації, наприклад, за допомогою припиняння і поновлювання реєстрації;

e) можливість для зовнішніх систем одержувати і видаляти записи реєстрації;

f) можливість для зовнішніх систем створювати і знищувати журнали реєстрацій.

С.2 Інформаційний погляд

Необхідна інформація про:

— зареєстровану інформацію (журнал реєстрацій і реєстраційні записи);

— керівні об'єкти, які моделюють джерело зареєстрованих подій.

На рисунку С.1 показано один з інформаційних поглядів на журнал реєстрацій.

Специфікація з використовуванням формальних нотацій також є частиною інформаційного погляду, але у прикладі не досліджується.

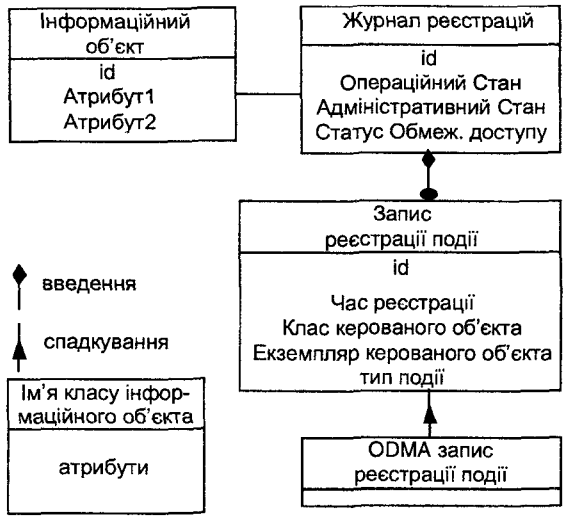


Рисунок С.1 — Приклад погляду на контроль реєстрації з використовуванням OMT-нотації

С.3 Обчислювальний погляд

З обчислювального погляду візуалізується інтерфейс операцій між обчислювальними об'єктами керування. У випадку контролю реєстрацій необхідні наступні обчислювальні об'єкти керування: журнал реєстрацій, запис реєстрації, об'єкт, що породжує сповіщення, та два об'єкти-менеджери. Доступні GDMO-специфікації (без обліку пакетів вимог) відносяться до інтерфейсу, ідентифікованого з обчислювального погляду.

Нижче наведено вихідний GDMO-шаблон класу керованого об'єкта log, взятий з CCITT Rec.X.721:

log MANAGED OBJECT CLASS
DERIVED FROM top;
CHARACTERIZED BY
— див.: CCITT Rec. X. 735 | ISO/IEC 10164-6 для опису класів керованих об'єктів.
logPackage PACKAGE
BEHAVIOUR
logBehaviour BEHAVIOUR
DEFINED AS "This managed object is used to store incoming event reports and local system notifications. Additional details are defined in CCITT Rec. X. 735 | ISO/IEC 10164-6. ";;
ATTRIBUTES
logId GET,
discriminatorConstruct GET-REPLACE,
administrativeState GET-REPLACE,
operationalState GET,
availabilityStatus PERMITTED VALUES Attribute-ASN1Module.LogAvailability
REQUIRED VALUES Attribute-ASN1Module.UnscheduledLogAvailability GET, logFullAction GET-REPLACE;
NOTIFICATIONS
objectCreation,
objectDeletion,
attributeValueChange,
stateChange,
processingErrorAlarm;;;
REGISTERED AS {smi2MObjectClass 6};

На рисунку С.2 об'єкт Менеджер1 приймає сповіщення через серверний інтерфейс сповіщень nsl. Крім того, створюється об'єкт Запис реєстрації, який може прочитати об'єкт Менеджер1 з використанням клієнтського інтерфейсу керування операцій ocl.

На рисунку С.3 інший об'єкт Менеджер2 може читати об'єкт Журнал реєстрації, використовуючи інтерфейс oc2, і об'єкт Запис реєстрації за допомогою oc3.

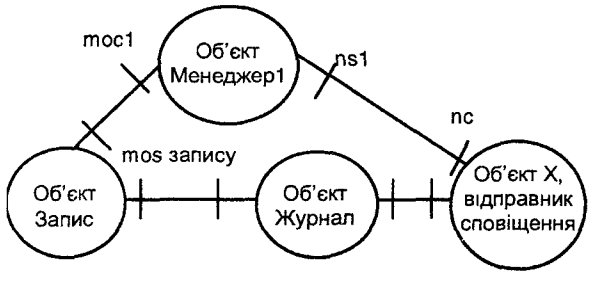


Рисунок С.2 — Приклад обчислювального погляду на контроль реєстрації (звичайні операції)

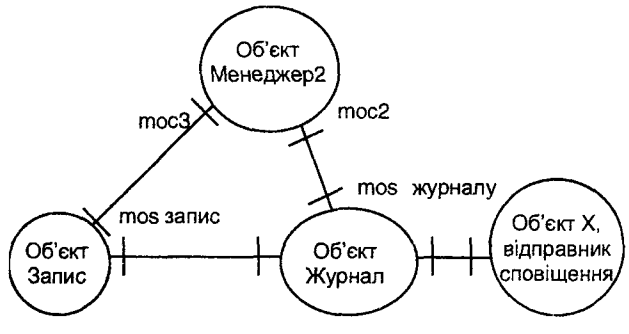


Рисунок С.3 — Приклад обчислювального погляду на контроль реєстрації (керування журналом реєстрації)

С.4 Інжиніринговий погляд

Для керування OSI-системами інжиніринговий погляд описує функціональність, необхідну для забезпечення комунікації між об'єктами в керівній ролі та об'єктами в ролі ресурсів з використанням CMISE і, наприклад, обробки транзакцій. На рисунку С.4 наведено приклад інжинірингового контролю реєстрацій.

У прикладі журнал реєстрацій і об'єкт, який генерує події, існують на одному комунікаційному вузлі. Розташування сервера сповіщень не зрозуміле з рисунка, він може існувати як на тому самому вузлі, так і на іншому. В останньому випадку йому потрібні власні комунікаційні канали.

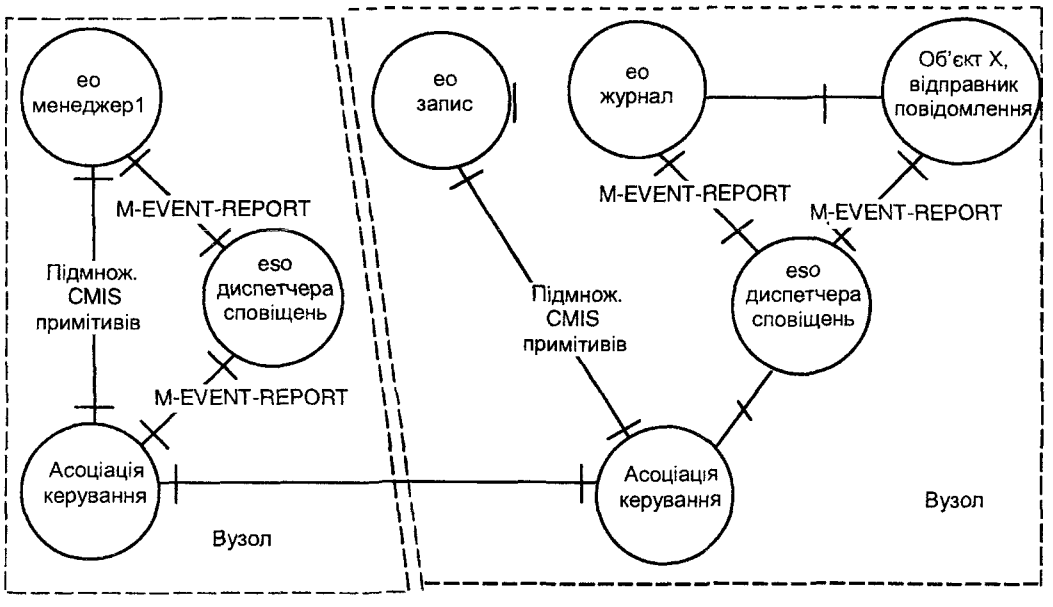


Рисунок С.4 — Приклад інжинірингового погляду керування журналом реєстрацій

С.5 Залежності між поглядами

Примітка. У випадках узагальнення немає потреби однозначного відображення об'єктів з різних поглядів.

На рисунку С.5 використано позначки: eo — інжиніринговий об'єкт, eso — інжиніринговий об'єкт підтримування.

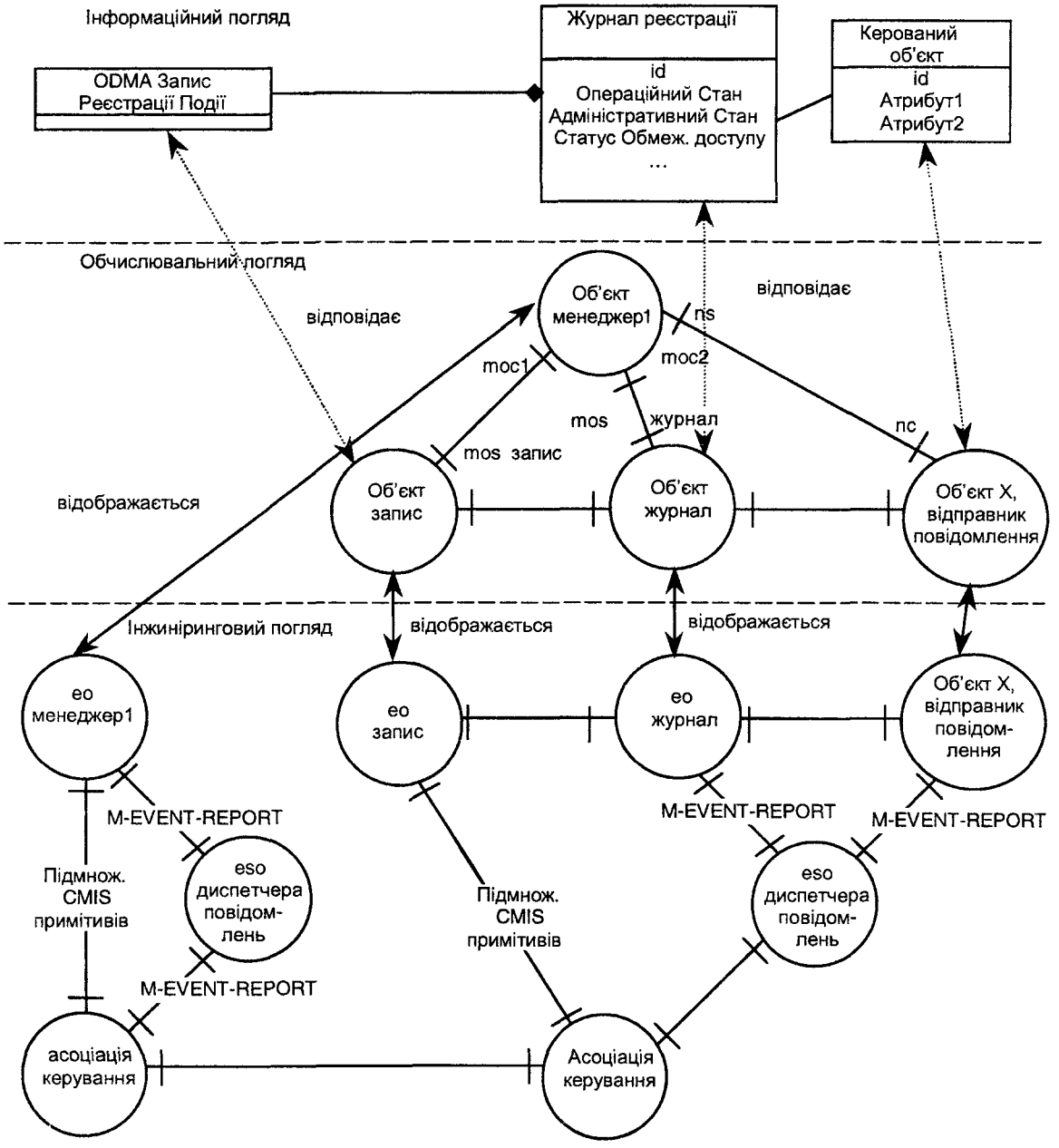


Рисунок С.5 — Залежності між поглядами

ДОДАТОК D

(довідковий)

ПРИКЛАД МОНІТОРИНГУ МЕТРИКИ

Мета цього прикладу — проілюструвати, спосіб подання клієнтського інтерфейсу операції керування, та фактично повністю специфікувати його, використовуючи GDMO і GRM.

Примітка 1. Метод, наведений у цьому прикладі, не єдиний, який можна використовувати в ODMA.
Примітка 2. Серверний інтерфейс сповіщень можна розібрати аналогічно, однак він не долучений до прикладу.

Приклад базується на структурі простого керованого об'єкта monitorMetric, який визначено в ITU-T Rec. X.739 | ISO/IEC 10164-11. Формальну GRM-нотацію (ITU-T Rec. X.725 | ISO/IEC 10165-7) використовують лише для специфікації відношень між серверним інтерфейсом операцій керування одного обчислювального об'єкта керування та серверним інтерфейсом операцій керування іншого обчислювального об'єкта керування (рисунок D.1).

Доволі просто подати клієнтський інтерфейс операцій керування як дзеркальне відображення серверного інтерфейсу цих операцій, який визначений з використанням GDMO і відповідає обмеженням, що накладаються специфікованим відношенням (рисунок D.2).

Досліджуваний об'єкт використовують винятково з метою ідентифікації клієнтського інтерфейсу операцій керування об'єкта моніторингу метрики.

Рисунок D.1 — Специфікація відношень

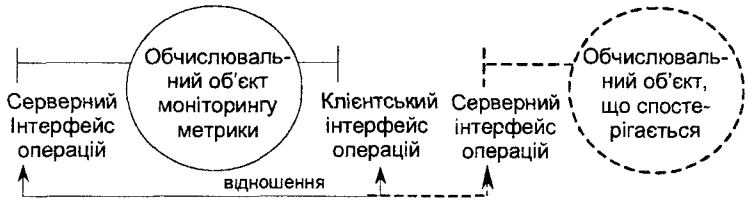


Рисунок D.2 — Специфікація інтерфейсу керування

D.1 Визначення об'єктів метрики

Необхідні такі визначення:

— Клас відношень meanMonitorMetric подає обчислювальний об'єкт meanMonitorMetric, обчислювальний інтерфейс цього об'єкта інтерпретується як специфічні ролі, описані класом керованого об'єкта.

— Клас керованого об'єкта meanMonitorControl описує характеристики екземплярів керованих об'єктів, які можна використовувати для контролю meanMonitorMetric.

— Клас керованого об'єкта scanObservedObjectValue описує характеристики екземплярів керованих об'єктів, за якими можна спостерігати, використовуючи meanMonitorMetric.

— Клас керованого об'єкта qualityOfServiceAlarm описує характеристики екземплярів керованих об'єктів, які можуть генерувати сповіщення meanMonitorMetric.

На рисунку D.3 це зображено графічно. Тут клієнт служби сповіщень — quality of service alarm client; сервер засобів контролю моніторингу — mean monitor control server, клієнт екземпляра об'єкта, який спостерігається — observed object instance client.

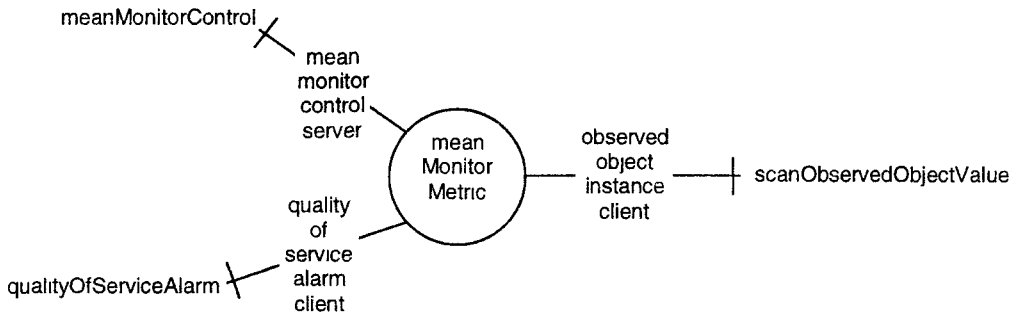


Рисунок D.3 — Обчислювальний об'єкт засобу моніторингу метрики

D.2 Визначення класу відношень
meanMonitorMetric RELATIONSHIP CLASS
BEHAVIOUR meanMonitorMetricBhv BEHAVIOUR DEFINED AS
«Обчислювальний об'єкт meanMonitorMetric ініціює операцію get екземпляра обчислювального об'єкта в ролі observedObjectInstanceClient, для одержування значення активного атрибута, який спостерігають; якщо операція сканування не приводить до одержування значення об'єкта, який спостерігається, до початку наступного періоду деталізації, то таке сканування недійсне і його не використовують для метричного алгоритму процесу відновлювання»;;
ROLE meanMonitorControlServer COMPATIBLE WITH meanMonitorControl;
ROLE observedObjectInstanceClient COMPATIBLE WITH scanObservedObjectValue;
ROLE qualityOfServiceAlarmClient COMPATIBLE WITH qualityOfServiceAlarm;
REGISTERED AS {};

D.3 Визначення класів керованих об'єктів
meanMonitorControl MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721 | ISO/IEC 10165-2":top;
CHARACTERIZED BY meanMonitorControlPkg PACKAGE
BEHAVIOUR meanMonitorControlBhv BEHAVIOUR DEFINED AS
«операції контролю повинні впливати на параметризовану поведінку метричного алгоритму сканування, виконуваного за установкою атрибутів»;
ATTRIBUTES
observedObjectInstance GET-REPLACE,
observedAttributeId GET-REPLACE,
granularityPeriod GET-REPLACE.
movingTimePeriod GET-REPLACE,
derivedGauge GET-REPLACE
notificationTriggerThreshold GET-REPLACE,
re-armThreshold GET-REPLACE,
operationalState GET-REPLACE;;
REGISTERED AS {};

scanObservedObjectValue MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721 | ISO/IEC 10165-2":top;
CHARACTERIZED BY scanObservedObjectValuePkg PACKAGE
BEHAVIOUR scanObservedObjectValueBhv BEHAVIOUR DEFINED AS
«Сумісні класи конкретного класу керованого об'єкта повинні мати як мінімум один атрибут із синтаксисом величини типу integer чи real.»;;
REGISTERED AS {};

Примітка. Необхідно використовувати механізм емуляції, оскільки об'єкт метрики працює на довільних класах керованих об'єктів.

qualityOfServiceAlarm MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721 | ISO/IEC 10165-2":top;
CHARACTERIZED BY qualityOfServiceAlarmPkg PACKAGE
BEHAVIOUR qualityOfServiceAlarmBhv BEHAVIOUR DEFINED AS
«Сумісні класи конкретного класу керованих об'єктів матимуть потребу в qualityOfService AlarmNotification у випадку позитивного кросингу події для тригера сповіщень»;
NOTIFICATION
"ISO/IEC 10164-4":qualityOfServiceAlarmNotification;;;
REGISTERED AS {};

D.4 Приклад обчислювальних об'єктів метрики

На рисунку D.4 наведено приклад використання об'єкта метрики. Визначені:
— обчислювальний об'єкт observedObjectExample, поданий як клас відношень;
— відображення відношень metricObjectExample;
— клас керованого об'єкта fullMeanMonitorControl;
— клас керованого об'єкта observedObjectInterfaceExample.
На рисунку D.4 використані позначення:
— засіб повного контролю моніторингу — full Mean Monitor Control;
— клієнт служби сповіщень — quality of Service Alarm Client;
— сервер засобів контролю моніторингу — mean Monitor Control Server;
— клієнт екземпляра спостеріганого об'єкта — observed Object Instance Client.

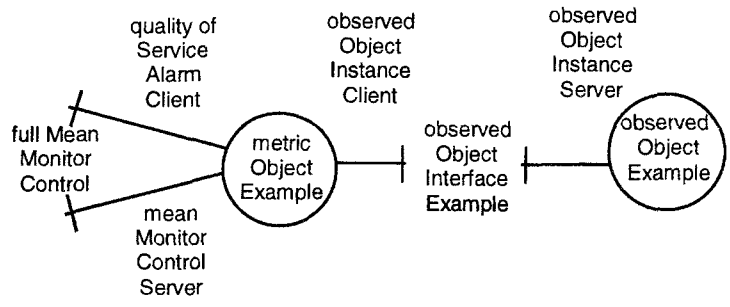


Рисунок D.4 — Приклад обчислювального об'єкта метрики

D.5 Приклад класу відношень

observedObjectExample RELATIONSHIP CLASS
BEHAVIOUR observedObjectExampleBhv;
ROLE observedObjectInstanceServer COMPATIBLE WITH observedObjectInterfaceExample;
REGISTERED AS {};

D.6 Приклад відображення відношень

metricObjectExample RELATIONSHIP MAPPING
RELATIONSHIP CLASS meanMonitorMetric;
RELATIONSHIP OBJECT fullMeanMonitorControl;
ROLE meanMonitoi-ControlServer RELATED CLASSES fullMeanMonitorControl;
ROLE observedObjectInstanceClient RELATED CLASSES observedObjectInterfaceExample
REPRESENTED BY RELATIONSHIP-OBJECT-USING-POINTER observedObjectInstance;
ROLE qualityOfServiceAlarmClient RELATED CLASSES fullMeanMonitorControl;
REGISTERED AS {} ;

D.7 Приклад класів керованих об'єктів

fullMeanMonitorControl MANAGED OBJECT CLASS
DERIVED FROM meanMonitorControl, qualityOfServiceAlarm;
REGISTERED AS {};

observedObjectInterfaceExample MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721 | ISO/IEC 10165-2":top;
CHARACTERIZED BY observedObjectInterfaceExamplePkg PACKAGE
BEHAVIOUR observedObjectInterfaceExampleBhv;
ATTRIBUTES
observedValue GET-REPLACE;;
REGISTERED AS {};

ДОДАТОК Е

(обов'язковий)

ПРИКЛАДИ ОБЧИСЛЮВАЛЬНИХ ШАБЛОНІВ

У додатку наведено приклади різних типів нотацій, які можна використовувати як обчислювальні шаблони.

Примітка. Метод, наведений у додатку, не є єдиним методом, який можна використовувати в ODMA.

Е.1 Обчислювальний шаблон ITU-T Rec. G.851.1

У описі операції з обчислювального погляду посилаються на схему інформаційного погляду, щоб визначити передумови, які система[3)] повинна перевіряти для можливості виконання конкретної операції, та постумови, які система повинна перевіряти після виконання операції. Клас відношень GRM специфікує ряд систем із забезпеченням:

— об'єктів, включених в усі екземпляри класу системи (через використання директиви <role>),

— допустимих обмежень їхніх станів (через використання директиви <invariant>),

— відношень, включених до класу системи (через використання директиви <invariant>),

— допустимих обмежень на комбіновані стани, описані у відношеннях (через використання директиви <invariant>).

За параметром операції буде вибрано екземпляр системи, який буде адресовано операцією в передумові і який буде подано як результат операції в постумові. Отже, поведінка повинна охоплювати:

— директиву передумови, яка посилається на клас системи специфікації інформаційного погляду.

— директиву постумови, яка посилається на клас системи специфікації інформаційного погляду.

— правила узгоджування параметрів для надання інформації про вибір.

Е.2 Приклади використовування обчислювального шаблону

У цьому прикладі наведено обчислювальний інтерфейс та одну з його операцій, яка забезпечує перехід від ssccNotConnected до ssccConnected серед простих підмережних з'єднань (тобто встановлює підмережне з'єднання між заданими a-кінцевою і z-кінцевою точками).

Посилання на мітки. У цьому додатку показано посилання на інформаційну статичну схему і ASN.1 процедури:

Повністю уточнене посилання на мітки	Використовуване локальне посилання
<"Rec. G.853.2", STATICSCHEMA:ssccNotConnected >	<ssccNotConnected>
<"Rec. G.853.2", STATICSCHEMA:ssccConnected >	<ssccConnected>

Повністю уточнене посилання на ASN.1-продукції	Використовуване локальне посилання
"M.3100:1995 : ASNIDefmedTypesModule"::Directionality	Directionality
"M.3100:1995 : ASNIDefmedTypesModule"::UserLabel	UserLabel

[3)] Термін «система» означає інформаційну систему.

Інтерфейс простого SNC-виконавця необхідний для задоволення вимог предметної області, встановлених у:

<"Rec. G.852.1", COMMUNITY:sscc, ACTION:scccl > ,
<"Rec. G.852.1", COMMUNITY:sscc. ACTION:sccc2 > .

COMPUTATIONAL_INTERFACE simpleSncPerformerIfce {
OPERATIONS <setupSubnetworkConnection>;
<releaseSubnetworkConnection>;

BEHAVIOUR «Обчислювальний об'єкт, що підтримує цей інтерфейс, як сервер, повинен задовольняти вимоги, встановлені у директиві BEHAVIOUR шаблонів OPERATION для кожної операції інтерфейсу»;

}

setupSubnetworkConnection operation
OPERATION setupSubnetworkConnection {
INPUT_PARAMETERS
subnetwork : Subnetworkid ::= REF(snQueryIfce);
snpa : SnTPId ::= REF(snTPQueryIfce);
snpz : SnTPId ::= REF(snTPQueryIfce);
dir : Directionality;
suppliedUserLabel: UserLabel;
-- рядки нульової довжини не підтримують
serviceCharacteristics: CharacteristicsId ::= REF(serviceCharacteristicsQueryIfce);
-- посилання можна використовувати для визначання Qo або характеристик процедур);
OUTPUT_PARAMETERS
newSNC : SNCId ::= REF(sncQueryIfce) ;
agreedUserLabel: UserLabel;

RAISED EXCEPTIONS
incorrectSubnetworkTerminationPoints : SnTPId;
subnetworkTerminationPointDisabled : SnTPId;
subnetworkDisabled : NULL;
subnetworkTerminationPointConnected : SnTPId;
operation Fails : NULL;
wrongDirectionality : Directionality;
userLabelInUse : UserLabel;

BEHAVIOUR
PARAMETER_MATCHING
subnetwork: < ssccNotConnected , ROLE:involvedSubnetwork > AND < ssccConnected , ROLE:involvedSubnetwork > ;
snpa : < ssccNotConnected , ROLE:potentialAEnd > AND < scmConnected , ROLE:connectedAEnd > ;
snpz : < ssccNotConnected , ROLE: potentialZEnd > AND < ssccConnected , ROLE:connectedZEnd > ;
dir : < ssccConnected, ROLE: involvedSubnetwork , ATTRIBUTE: directionality > ;
newSNC : <ssccConnected, ROLE: involvedSubnetwork> ;
suppliedUserLabel: <ssccConnected, ROLE:involvedSubnetwork, ATTRIBUTE: userLabel >
OR <>; *-Користувач не повинен підтримувати користувацьке значення мітки*
agreedUserLabel: <ssccConnected, ROLE:involvedSubnetwork, ATTRIBUTE: userLabel > ;
serviceCharacteristics:<ssccConnected,ROLE: involvedServiceCharacteristics > ;
PRE_CONDITIONS < ssccNotConnected> ;

-- і scmNotConnecled схема визначає тип схеми з двома незв'язними інформаційними
-- networkTP об'єктами — кандидатами на з'єднання типу «точка-точка»
-- служби керування.

POST CONDITIONS < ssccConnected> ;

-- і Схема scmConnected визначає тип схеми з двома зв'язковими networkTP
-- інформаційними об'єктами — кандидатами на з'єднання типу «точка-точка»
-- служби керування.

EXCEPTIONS
IF PRECONDITION <inv 1> NOT_VERIFIED RAISE_EXCEPTION
incorrectSubnetworkTerinination Points ;
IF PRECONDITION <inv_2> NOT_VERIFIED RAISE_EXCEPTION
subnetworkTerminationPointConnected ;
IF PRE CONDITION <inv_3> NOT_VERIFIED RAISE_EXCEPTION
subnetworkTerminationPointConnected ;
IF POST_CONDITION <inv_I> NOT_VERIFIED RAISE EXCEPTION
operationFails ;
IF POST_CONDIT10N <inv 2>NOT_VERIFIED RAISE EXCEPTION
operationFails ;
IF POST_CONDITION <inv_3> NOT_VERIFIED RAISE EXCEPTION
operationFails ;
IF POST CONDITION <inv_4> NOT_VERIFIED RAISE_EXCEPTION
userLabelInUse ;
INFORMAL
«Ця операція встановлює підмережне з'єднання між заданими a-кінцевою точкою snip і z-кінцевою точкою sntp»;

ДОДАТОК F

(довідковий)

ПРИКЛАД СПЕЦИФІКАЦІЇ СПІЛЬНОСТІ ПРЕДМЕТНИХ ОБЛАСТЕЙ

Примітка. Метод, поданий у додатку, не єдиний, який можна використовувати в ODMA.

F.1 ODP-принципи погляду предметної області

цей додаток повторює ODP-RM визначення для полегшення розуміння шаблонів у F.2.

контракт

Погоджено регульована складова колективної поведінки набору об'єктів. Контракт накладає зобов'язання на включені в нього об'єкти. Специфікація контракту може містити:

— специфікацію різних ролей об'єктів, включених у контракт, і інтерфейсу, асоційованого з такими ролями;

— атрибути якісних характеристик служб;

— ознаки чи тривалості періодів ймовірності;

— ознаки поведінки, що робить контракт недійсним;

— умови життя і безпеки.

роль предметної області

Підмножини, які виділені з поведінки об'єкта і відповідають специфічній функціональності, називають ролями предметної області. У розгляді об'єкта в термінах ролі предметної області цікавою є лише зазначена підмножина його дій, інші дії відкидаються, можливо, для цілей інших ролей предметної області. Кожен об'єкт може мати кілька ролей предметної області в різний час взаємодії та приймати різні набори ролей предметної області в різний час.

Ролі спільності предметних областей забезпечують приватні функції цілей сукупності.

Ролі предметної області специфічних ODMA-інтересів є різними варіантами керованої і керівної ролей.

спільність

Загальні вимоги формалізують за допомогою первинної ідентифікації спільностей ODP-систем, де спільність — це група ролей, спільно виконуваних з загальною метою. Мета спільності повинна бути чітко сформульована. Зазвичай метою спільності є забезпечення приватної служби.

Спільність — композиція об'єктів, сформована для досягнення визначеної мети. Мету задають як контракт, який специфікує, як саме досягти цієї мети.

Спільність визначають через її цілі (тобто загальні цілі ролей, що утворюють спільність), визначання кожної ролі у спільності і політиці, застосовуваної до спільності, як до цілого.

політика

Політику спільності визначають як набір дозволів, зобов'язань і заборон, застосовуваних або до клієнта або до провайдера в інтересах спільності.

— Політика: набір правил, який відноситься до приватної мети. Правило може виражатися як дозволи, зобов'язання або заборони.

— Дозвіл: розпорядження про дозвіл на особливу поведінку.

— Зобов'язання: розпорядження про вимогу особливої поведінки.

— Заборона: розпорядження про заборону на особливу поведінку.

дія

Щось, що відбувається. Кожну дію визначають за допомогою імені дії і специфікації політики дії, яку визначають як набір дозволів, зобов'язань і заборон, застосовуваних до ролей в інтересах спільності.

діяльність

ODP-RM дає таке визначення діяльності:

Діяльність — однокореневий ациклічний граф (single-headed acyclic graph) дій, де входження кожної дії в граф означає входження всіх безпосередніх попередніх дій.

F.2 Приклад специфікації спільності предметної області

F.2.1 *Спільність керування простим підмережним з'єднанням*

цілі спільності

Метою спільності є керування підмережними з'єднаннями типу «точка-точка» між заданим набором кінцевих точок, раніше встановлених на межі підмережі. Це відбувається за допомогою найменування двох дій у запиті підмережного з'єднання: установлення і звільнення.

ролі спільності

Роль *ініціатор* відображає клієнта служби керування простим підмережним з'єднанням.

Роль *провайдер* відображає сервер служби керування простим підмережним з'єднанням.

Роль *порт* відображає два G.805 порти, включених у спільність керування простим підмережним з'єднанням.

Роль *sn* відображає G.805 підмережу, для якої визначена служба керування простим підмережним з'єднанням.

Роль *snc* відображає G.805 підмережне з'єднання, що входить у спільність керування простим підмережним з'єднанням.

політика спільності

Не визначена.

F.2.2 *Опис дій спільності*

Установка SNC типу «точка-точка»

Ця дія встановлює підмережне з'єднання типу «точка-точка» між двома портами однієї підмережі. Нижче специфікована політика дії.

Зобов'язання 1

Ініціатор повинен ідентифікувати два порти, що є частиною підмережного домена.

Зобов'язання 2

Якщо запитувана послуга є односпрямованою, то ініціатор повинен ідентифікувати один порт джерела й один порт приймача.

Зобов'язання 3

У випадку відхилу запиту на послугу провайдер повинен дозволити ініціаторові довідатися, яка політика була порушена.

Зобов'язання 4

У випадку установлювання послуги провайдер повинен надати ініціаторові таку інформацію з'єднання:

— унікальний ідентифікатор підмережного з'єднання,

— про одно- чи двоспрямованість з'єднання,

— про порти, що виступають як кінцеві точки підмережного з'єднання.

Дозвіл 1

Ініціатор може специфікувати, яке саме підмережне з'єднання запитується: дво— чи односпрямоване.

Дозвіл 2

Ініціатор може специфікувати обмеження маршрутизації, засноване на ідентифікаторі каналу (Link ID) чи ідентифікаторі підмережі (Subnetwork(Matrix) ID).

Дозвіл 3

Ініціатор може задати ступінь готовності послуги.

Дозвіл 4

Ініціатор може задати ідентифікатор якісних характеристик послуги (Quality of Service — QOS).

Дозвіл 5

Ініціатор може задати смугу частот.

Заборона 1

Провайдер не може задовольнити запит, якщо використовують один з портів.

зміна SNC типу «точка-точка»

Цю дію використовують для зміни простого підмережного з'єднання. Нижче специфікована політика дій.

Зобов'язання 1

З'єднання повинно бути частиною підмережного домену.

Зобов'язання 2

Ініціатор повинен ідентифікувати з'єднання як частину запиту.

Допущення 1

Ініціатор може задати нову смугу частот.

звільнення з'єднання

Використовують для звільнення простого підмережного з'єднання. Нижче специфікована політика дій.

Зобов'язання 1

З'єднання повинне бути частиною підмережного домену.

Зобов'язання 2

Ініціатор повинен ідентифікувати з'єднання як частину запиту.

Зобов'язання 3

Провайдер повинен відновити доступність усіх ресурсів після очищування з'єднання.

Зобов'язання 4

Провайдер повинен вказати причину на випадок відмови дії.

Зобов'язання 5

Провайдер повинен забезпечити з'єднання, що звільняється, унікальним ідентифікатором з'єднання.

ДОДАТОК G

(довідковий)

ТЕРМІНОЛОГІЯ CORBA

Далі наведено ODMA-термінологію, використовувану для опису принципів CORBA. У правому стовпчику подані українська й англійська назви термінів CORBA, а в лівому — тлумачення термінів ODMA (ODP).

Динамічний інтерфейс виклику (Dynamic Invocation Interface)	— механізм виклику на інжиніринговому рівні операції об'єкта з боку клієнта
Динамічний інтерфейс каркаса (Dynamic Skeleton Interface)	— механізм прийому на інжиніринговому рівні операції об'єкта з боку сервера
Подія (Event)	— сигнал
Канал події — Event channel	— спеціальний тип каналу
Виключення (Exception)	— припинення взаємодії, що виражає помилку виконування операції
Протокол взаємодії (Interoperability protocol) (наприклад, GIOP)	— проектний протокол
Об'єкт (Object)	— інтерфейс обчислювального об'єкта
ORB	— набір інженірингових об'єктів
Каркас (Skeleton)	— заглушка з боку сервера
Посередник, заглушка (Stub)	— заглушка з боку клієнта
Репозитарій інтерфейсу (Interface Repository)	— та частина ODP-репозитарію типів, яка зберігає визначення інтерфейсу

ДОДАТОК H

(довідковий)

ЗАБЕЗПЕЧЕННЯ МІЖМЕРЕЖНОГО ОБМІНУ CORBA З ІНШИМИ СТАНДАРТНИМИ ПРОТОКОЛАМИ КЕРУВАННЯ

Існує два основних стандарти інтерфейсу мережного керування: інформаційні CMIS/CMIP та GDMO/ASN.1 моделі й керування Інтернет [4)]. Важливо забезпечити цілісний погляд на цей інтерфейс у CORBA-середовищі.

На рисунку H.1 показано, як CORBA-керована система може забезпечувати міжмережний обмін з різними протоколами керування (наприклад, CMIP чи SNMP) через шлюзи чи за допомогою CORBA протоколів взаємодії (наприклад, GIOP). Трансляцію між взаємодіями (передбачається, що трансляції існують між GDMO/ASN.1 та ODP IDL або між SNMP MIB-стислим визначенням та ODP IDL) виконують в шлюзах, а не безпосередньо у системі керування.

ITU-T Рекомендації / міжнародні стандарти таких трансляцій підпорядковують міжгалузевим ODMA-функціям.

[4)] SNMPv2 Working Group, CASE (J.), MCCLOGHRIE (K.), ROSE (M.) and WALDBUSSER (S.): Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2) (Структура керівної інформації для версії 2 протоколу керування простою мережею (SNMPv2)) RFC 1902, January 1996.

SNMPv2 Working Group, CASE (J.), MCCLOGHRIE (K.), ROSE (M.) and WALDBUSSER (S): Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2) (Текстові узгодження для версії 2, протоколу керування простою мережею (SNMPv2)) RFC 1903, January 1996.

SNMPv2 Working Group, CASE (J.), MCCLOGHRIE (K.), ROSE (M.) and WALDBUSSER (S.): Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2) (Протокольні операції для версії 2 протоколу керування простою мережею (SNMPv2)) , RFC 1905, January 1996.

CASE (J.), FEDOR (M.), SCHOFFSTALL (M.) and DAVIN (J.): Simple Network Management Protocol (Протокол керування простою мережею), STD 15, RFC 1157, May 1990.

Примітка. У JIDM-документі трансляції специфікацій[5] визначено алгоритми трансляції специфікацій між:

— GDMO/ASN. 1 і ODP IDL,
— ODP IDL і GDMO/ASN. 1,
— SNMP MIB-стислим визначенням і ODP IDL.

Н.1 Міжмережний обмін між GDMO/ASN.1 і ODP IDL

Використовуючи ODP IDL, необхідно так описати GDMO керовані об'єкти, щоб CORBA системи керування могли здійснити доступ до GDMO керованих об'єктів. Для досягнення цього необхідна трансляція специфікацій, визначених у GDMO/ASN.1, у специфікації, визначені в ODP IDL. Така трансляція називається трансляцією специфікацій.

Базові принципи об'єктів, спадковування, атрибутів, операцій, станів, поведінки, ікапсульовані в GDMO/ASN.1 та ODP IDL, є подібними. Отже, досяжна двостороння трансляція специфікацій GDMO у ODP IDL та двостороння динамічна трансляція взаємодій (тобто шлюз) CMIP у CORBA протоколи взаємодії.

Н.2 Міжмережний обмін між SNMP і ODP IDL

Внаслідок спрощеної природи SNMP немає придатних засобів подання звичайного ODP IDL інтерфейсу, що використовує SNMP MIB визначення. Проте доволі легко відобразити SNMP MIB визначення в ODP IDL визначенні інтерфейсу.

Оскільки SNMP MIB визначення відображаються у відповідному наборі ODP IDL за допомогою механізму трансляції специфікацій, то досяжна двостороння динамічна трансляція взаємодій між SNMP та CORBA протоколами взаємодії.

---

[5] Preliminary Specification - Inter-domain Management: Specification Translation, The Open Group (Попередня специфікація - Внутрішньо-доменне керування: Трактування специфікації, Відкрита група), ISBN: 1-85912-150-0, Document Number: P509.

35.080

Ключові слова: відкрита система, розподілене керування, брокер об'єктних запитів, об'єкт, інтерфейс, контракт, клієнт, сервер, упорядкування/розупорядкування.

Редактор **С. Ковалець**
Технічний редактор **О. Касіч**
Коректор **О. Тарасун**
Комп'ютерна верстка **С. Павленко**

Підписано до друку 11.05.2004. Формат 60 × 84 1/8.
Ум. друк. арк. 6,97. Зам. 1267 Ціна договірна.

Редакційно-видавничий відділ ДП «УкрНДНЦ»
03115, Київ, вул. Святошинська, 2