МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

РОЛЬЩИКОВ В. Б.

ТЕХНОЛОГІЇ РОЗПОДІЛЕНИХ СИСТЕМ ТА ПАРАЛЕЛЬНИХ ОБЧИСЛЕНЬ. ГРІД-ТЕХНОЛОГІЇ. ЗМІСТОВНИЙ МОДУЛЬ №3

Конспект лекцій

Одеса
Одеський державний екологічний університет
2019

УДК 681.518
Р68

Рекомендовано методичною радою Одеського державного екологічного університету Міністерства освіти і науки України як конспект лекцій (протокол №1 від 30.08. 2019 р.)

Рольщиков В. Б.
Технології розподілених систем та паралельних обчислень. Грід-технології. Змістовний модуль №3: конспект лекцій. Одеса, Одеський державний екологічний університет, 2019. 186 с.

В конспекті лекцій з курсу «Технології розподілених систем та паралельних обчислень» розглянуті основні питання архітектури грід-систем. Паралельні обчислення є перспективною (і дуже привабливою) областю застосування обчислювальної техніки і являють собою складну науково-технічну область діяльності. Тим самим, знання сучасних тенденцій розвитку комп'ютерів і апаратних засобів для досягнення паралелізму, уміння розробляти моделі, методи й програми паралельного рішення завдань обробки даних варто віднести до числа важливих кваліфікаційних характеристик сучасного фахівця із прикладної математики, інформатиці й обчислювальній техніці.

Конспект лекцій призначений для студентів четвертого курсу Одеського державного екологічного університету, які навчаються за кваліфікаційним рівнем «бакалавр» та спеціальністю 122- «Комп'ютерні науки».

ISBN 978-966-186-086-4

ЗМІСТ

ТЕРМІНИ І СКОРОЧЕННЯ, ЗАСТОСОВАНІ У КОНСПЕКТІ

Бозон – Частинка з цілим значенням спіну.

ВО – Віртуальна організація.

Геном – Сукупність всієї спадкової генетичної інформації організму.

Геноміка – Розділ генетики, предметом дослідження якого є організація та функціонування геномів (див.) живих організмів.

Глюон – Електрично нейтральна елементарна частинка, яка відіграє таку ж роль у сильній ядерній взаємодії, як фотон в електромагнітній.

Глюонне поле – 4-векторне поле в теоретичній фізиці елементарних частинок, яке описує еволюцію глюонів (див.) та задає сильну взаємодію між кварками (див.).

Докінг молекулярний (молекулярне стикування) – Метод молекулярного моделювання, який дозволяє передбачити найбільш вигідну для утворення стійкого комплексу орієнтацію і конформацію однієї молекули (ліганда) у центрі зв’язування іншої (рецептора).

Експресія генів – Процес, при якому спадкова інформація генів використовується для синтезу білка.

Ензими (ферменти) – Органічні каталізатори білкової природи, які утворюються в живих організмах.

ЕЦП – Електронний цифровий підпис.

ІБ – Інформаційна безпека.

Кварки – Елементарні частинки і фундаментальні складові матерії.

Конфайнмент (утримання) – Явище квантової хромодинаміки (див.), яке робить неможливим існування у вільному стані кварків (див.) із кольоровим зарядом.

МР	– Магнітний резонанс.
Мюон	– Нестійка елементарна частинка з негативним електричним зарядом і спіном 1/2.
Орбіталь атомна	– У квантовій механіці й хімії базисна хвильова функція (див.) електрона в атомі.
Орбіталь молекулярна	– Наближена хвильова функція (див.) електронів молекули, утворена суперпозицією атомних орбіталей (див.) різних атомів.
ПЗ	– Програмне забезпечення або Програмний засіб.
ППЗ	– проміжне програмне забезпечення Грід.
Протеом	– Сукупність всіх білків клітини, тканини, організму або популяції, які експресуються (див.) за певними, чітко визначеними умовами.
Протеоміка	– Наука про протеом (див.), яка вивчає протеїни в цілому та їх структуру і функції зокрема.
Стеларатор	– Замкнута магнітна пастка для утримання високотемпературної плазми.
СУПО	– Система управління пакетною обробкою завдань.
СУРБД	– Система управління реляційною базою даних.
Теватрон (Tevatron)	– Кільцевий прискорювач заряджених частинок, другий у світі за величиною прискорювач після LHC (див.).
ФВЕ	– Фізика високих енергій.
Філогенія або філогенез	– Історичний розвиток, як окремих видів і систематичних груп організмів, так і органічного світу в цілому.
ФКП	– Функція контрастного перенесення.
Хвильова функція	– Комплекснозначна функція, яка використовується в квантовій механіці для опису стану квантовомеханічної системи.
Хіггса бозон (див.)	– Єдина відома скалярна елементарна частинка.

Хромодинаміка квантова – Розділ теоретичної фізики, який описує сильну взаємодію між кварками через глюонні поля (див.).

ACL – Access Control List; Список контролю доступу до ресурсів Грід.

AES – Advanced Encryption Standard; Симетричний алгоритм блочного шифрування.

AFS – Andrew File System; Розподілена мережева файлова система на основі набору захищених серверів, надає всім клієнтським робочим станціям однорідний територіально-незалежний простір імен файлів.

ALICE – A Large Ion Collider Experiment; Експеримент на великому іонному колайдері з вивчення фізики сильних взаємодій і утворення кварк-глюонної (див.) плазми.

AliEn – ALICE Environment; ALICE (див.) оточення.

API – Application Programming Interface; Прикладний програмний інтерфейс.

ATLAS – A Toroidal LHC ApparatuS; Тороидальная установка на LHC (див.) для вивчення глибинних основ будови речовини і фундаментальних сил, які сформували Всесвіт.

BaBar – експеримент в галузі фізики елементарних частинок за участю понад 600 фізиків з 75 інститутів 10 країн світу.

CA – Certificate Authority; Центр сертифікації, спеціальна організація, яка має повноваження видавати, підписувати цифрові сертифікати.

CC – Combined Catalog; Об’єднаний каталог.

CC або Common Criteria – Common Criteria for Information Technology Security Evaluation; Стандарт загальних критеріїв безпеки інформаційних технологій.

CDF – Collider Detector at Fermilab; Колайдерний детектор Фермілаб для детектування зіткнень протонів-антипрото-

нів на прискорювачі частинок Теватрон (див.).

CDSS – Clinical decision support system; Система підтримки прийняття лікарських рішень, система штучного інтелекту в медицині для допомоги лікарям.

CE – Computing Element; Вузол обчислювального кластера.

CERN – l'Organisation européenne pour la recherche nucléaire; Європейська організація з ядерних досліджень

ClassAd – Classified Advertisements; Класифікація «оголошень» (класів), універсальна мова SDK (див.) HTCondor (див.).

CLI – Command Line Interface; Інтерфейс командного рядка.

CMS – Compact Muon Solenoid; Компактний мюонний (див.) соленоїд, один з детекторів Великого адронного колайдера для вирішення широкого спектра задач фізики елементарних частинок.

CMS – Cryptographic Message Syntax; Стандарт синтаксису криптографічно-захищених повідомлень для цифрового підпису або шифрування будь-якої форми цифрових даних.

CRL – Certificate Revocation List; Список відкликаних сертифікатів.

CS – Catalog Services; сервіс каталогів в Грід-системах.

CVS – Concurrent Versions System; Система паралельних версій, механізм управління версіями вихідного і об’єктного коду.

DGAS – DataGrid Accounting System; Сервіс обліку споживання ресурсів Грід.

DiRAC – Distributed Research using Advanced Computing; Розподілені дослідження з використанням передових обчислень.

DMS – Data Management Subsystem; Підсистема управління даними.

DMTF – Distributed Management Task Force; Цільова група з розподіленої розробки, підтримки і поширення стандартів управління системами в промислових IT середовищах.

DN – Distinguished Name; Унікальне ім’я власника сертифікату.

DS – Data Scheduler; Планувальник передачі даних.

EAL – Evaluation Assurance Levels; гарантовані параметри відповідності, які пропонуються CC (див.).

e-Commerce (E-Commerce, eCommerce) – Застосування технологій Грід-обчислень і інтенсивних обчислень у високорозвинених мережних середовищах у комерції.

EDG – European DataGrid; один з європейських сегментів глобального Грід EGEE (див.).

EFDA – European Fusion Development Agreement; Європейська угода з розробок в галузі термоядерного синтезу.

EGA – Enterprise Grid Alliance; Альянс Ініціатив Грід-розробників.

EGEE – Enabling Grids for E-sciencE; Грід придатний для e-science (див.)

EGEODE – Expanding GEOsciences on Demand; Розширення кола інтересів наук про Землю на вимогу.

e-Health (E-Health, eHealth) – Застосування технологій Грід-обчислень і інтенсивних обчислень у високорозвинених мережних середовищах в медицині.

e-Science (E-Science, eScience) – Наукові розробки, які потребують інтенсивних обчислень у високорозвинених мережних середовищах, або використовують величезні набори даних, які вимагають Грід-обчислень.

ESR – Earth Science Research; Дослідження в галузі наук про Землю.

FC – File Catalog; Файловий каталог.

FTP – File Transfer Protocol; Протокол передачі файлів з гарантією передачі, або повернення повідомлення про помилку.

FTS – File Transfer service; Служба передачі файлів.

GASS – Global Access to Secondary Storage; Глобальний доступ до додаткових сховищ, компонент GTK (див.) для здійснення глобального доступу до вторинної системи зберігання у розподіленому середовищі.

GEMS – Grid Enabled Molecular Simulator; Молекулярний симулятор на основі Грід-технологій.

GENIUS – Grid Enabled web eNvironment for site Independent User job Submission; Веб-середовище з підтримкою Грід для сайта виконання завдань незалежних користувачів (повнофункціональний Грід-портал для демонстрації можливостей технології Грід).

GILDA – Grid INFN Laboratory for Dissemination Activities; лабораторія INFN (див.) з питань розповсюдження Грід.

gLite – ППЗ (див.) для комп'ютерних обчислень, яке використовується в експериментах CERN LHC (див.) та інших наукових галузях.

GMA – Grid Monitoring Architecture; Архітектура системи виявлення несправностей Грід.

GME – Grid Management Entity; одиниця управління Грід.

GRACE – GRid enabled seArch and Categorization Engine; Грід-система пошуку й категоризації.

GRAM – Globus Resource Allocation Manager; Менеджер розподілу ресурсів.

GRIP – Grid Resource Information Protocol; Інформаційний протокол Грід-ресурсів, протокол ресурсного рівня.

GRMA – Globus Resource Management Architecture; Спільна архітектура компонентів і засобів GTK (див.) для управлін-

ня ресурсами.

GSI – Globus (або Grid) Security Infrastructure; Інфраструктура безпеки Грід, компонент GTK (див.) для забезпечення захисту системи і даних з підтримкою їхнього шифрування.

GSS-API – Generic Security Service API (див.); Стандартний прикладний програмний інтерфейс Загальної служби безпеки

GTK – Globus Toolkit; Набір програмних засобів Грід систем.

GUID – Grid Unique Identifier; Глобальний унікальний ідентифікатор.

HC – Host Certificate; Сертифікат обчислювального вузла.

HLR – Home Location Register; Служба в децентралізованій інфраструктурі серверів для збирання і збереження записів використання ресурсів.

HTC – High Throughput Computing; Обчислення з високою пропускною здатністю.

HTCondor – Високопродуктивна платформа програмування з відкритим вихідним кодом для розподіленого розпаралелювання інтенсивних обчислювальних завдань.

ICTP – The Abdus Salam International Centre for Theoretical Physics; Міжнародний центр теоретичної фізики ім. Абдуса Салама.

IDUP-GSS-API – Independent Data Unit Protection Generic Security Service API; API (див.) Загальної служби захисту для незалежного захисту блоку даних.

IETF – Internet Engineering Task Force; Інженерна рада Інтернету.

INFN – Istituto Nazionale di Fisica Nucleare; Національний інститут ядерної фізики Італії

Interlogger – Демон служби подій, який пересилає події одному з

серверів зберігання.

IPC – Inter-Process Communication або Interprocess Communication; Механізми, які надає операційна система, щоб дозволити процесам управляти загальними даними.

ITER – the International Thermonuclear Experimental Reactor; Міжнародний експериментальний термоядерний реактор.

ITF – Industry Task Force; промислова робоча група.

ITU – International Telecommunication Union; Міжнародний консультаційний комітет з телефонії і телеграфії.

JDL – Job Description Language; Високорівнева мова опису задач, заснована на ClassAd (див.).

LB – Logging and Bookkeeping; Підсистема протоколювання.

LCG – LHC Computing Grid; Грід, спроектований в CERN (див.) і призначений для обробки великих обсягів даних, які надходять з LHC (див.).

LDAP – Lightweight Directory Access Protocol; Полегшений протокол доступу до каталогів у веб-системах.

LFN – Logical File Name; Логічне ім’я файла.

LHC – Large Hadron Collider; Великий адронний колайдер, найбільший у світі прискорювач елементарних частинок.

LHCb – The Large Hadron Collider Beauty Experiment; Експеримент з вивчення порушення симетрії заряду і парності.

Locallogger – Локальна служба подій CE (див.).

Logical Symlinks – Символічні посилання.

LSF – Load Sharing Facility; Комерційна система аналогічна PBS (див.) для динамічного балансування навантаження в умовах багатокористувальницького гетерогенного середовища.

MC – Metadata Catalog; Каталог метаданих.

MDS – Monitoring and Discovery Service; Служба моніторингу і пошуку інформації про Грід-систему.

MSS – Mass Storage System; Системи масового зберігання.

NDGF – Nordic Data Grid Facility; Північна система передачі даних, загальна інфраструктура e-Science (див.), яка надається скандинавськими країнами для наукових обчислень і зберігання даних.

NFS – Network File System; Мережева файлова система.

NFS – Network File System; Протокол мережевого доступу до файлових систем, дозволяє підключати (монтувати) віддалені файлові системи через мережу.

NorduGrid – Колаборація яка координує розробку та забезпечує підтримку вільно поширюваного ППЗ (див.) з відкритим кодом для Грід-інфраструктур, відомого під назвою Advanced Resource Connector (ARC).

NQE – Network Queuing Environment; Менеджер ресурсів на суперкомп'ютерах, кластерах і системах Cray.

OASIS – Organization for the Advancement of Structured Information Standards; Організація з вдосконалювання стандартів структурованої інформації, для управління розробкою, конвергенцією і прийняттям промислових стандартів, які відносяться до веб-служб.

OGSA – Open Grid Services Architecture; Відкрита Архітектура Грід-служб, в якій основний об'єкт – це сервіс (служба) і Грід-служба визначається як спеціальний вид веб-служби.

OGSI – Open Grid Services Infrastructure; Відкрита Інфраструктура Грід-служб, це стандарт створення, іменування, керуванням часом життя, моніторингу і передачі інформації між Грід-службами.

OSG – Open Science Grid; Всесвітня мережа технологічних ресурсів для розподілених обчислень і наукових досліджень, керується ВО (див.) Open Science Grid Consortium (OSGC).

PA – Price Authority; Служба розцінок послуг Грід.

PBS – Portable Batch System; Система управління ресурсами й завантаженням кластерів.

Pharmacokinetics – Експеримент в EGEE (див.) з вивчення впливу лікарського засобу на організм.

PKI – Public Key Infrastructure; Технологія відкритих криптографічних ключів

PSE – Problem Solving Environment; Сервіси управління робочим завантаженням і організацією спільних робіт в Грід.

RC – Replica Catalog; Каталог реплік.

RDIG – Russian Data Intensive Grid; Грід для великих обсягів даних.

Resource HLR – Служба HLR (див.) для провайдерів.

RFT – Remote File Transfer; Сервіси запуску FTP (див.) для обміну файлами по мережі.

R-GMA – Relational Grid Monitoring Architecture; Реляційна реалізація GMA (див.).

RSC – Remote System Call; Виклик віддалених системних функцій.

RSL – Resource Specification Language; Мова специфікації ресурсів Грід.

SCAI – Fraunhofer Institute for Algorithms and Scientific Computing; Інститут алгоритмів і наукових обчислень ім. Фраунгофера.

SDK – Software Development Kit; Набори інструментальних засобів для розробки програмного забезпечення.

SE – Storage Element; Ресурс зберігання даних.

SOAP – Simple Object Access Protocol; Протокол взаємодії веб-сервісів заснований на XML

SRM – Storage Resource Manager; Диспетчер ресурсів зберігання.

SURL – Site Universal Resource Locator, Site URL; URL, який обумовлює фізичне місце розташування файла або його репліки.

SWIG – Simplified Wrapper and Interface Generator; Вільний інструмент для зв'язування програм і бібліотек.

TURL – Transport URL; URL, який використовується для фактичного пересилання файла за допомогою будь-якого стандартного транспортного протоколу.

UC – User Certificate; Сертифікат користувача Грід.

UDDI – Universal Description Discovery and Integration; Універсальний метод пошуку й інтеграції веб-сервісів.

UR – Usage Records; Запис використання ресурсів Грід.

User HLR – Служба HLR (див.) для користувачів.

User Tag – Тег користувача, спеціальний фрагмент користувальницького коду в завданні.

VOMS – Virtual Organization Membership Service; Сервіс управління членством у віртуальних організаціях.

W3C – World Wide Web Consortium; Консорціум Всесвітнього павутиння, головна міжнародна організація, з розробки і впровадження технологічних стандартів для Всесвітнього павутиння.

WM – Workload Manager; Менеджер завантаження обчислювального вузла.

WMS – Workload Management System; Система управління завантаженням обчислювального вузла.

WSDL – Web Services Description Language; Мова опису веб-

сервісів.

WSDM – Web Services Distributed Management; Розподілене управління веб-службами, стандарт веб-служби для управління та моніторингу стану інших служб.

WS-I – Web Services Interoperability Organization; Організація з розвитку можливостей взаємодії веб-сервісів, розроблювач рекомендацій щодо використання стандартів веб-сервісів.

WSRF – Web Services Resource Framework; Структура Ресурсів Веб-служб, сімейство специфікацій, опублікованих OASIS (див.) для веб-служб.

ВСТУП

В процесі створення й застосування суперкомп'ютерних технологій, з якими ми познайомилися в попередньому модулі курсу, легко виявити такі властиві їм проблеми:

- суперкомп'ютерні системи доволі коштовні, але вони вкрай швидко «морально старіють». Наприклад, суперкомп'ютер, який сьогодні потрапив навіть в першу сотню списку «Top 500» і не на останні місця, через 2–3 роки взагалі буде витиснутий зі списку. Звичайно, це не означає, що через 2–3 роки суперкомп'ютер викидають на звалище, але, скоріше за все, для тієї організації (проблематики), для якої створювався, він уже стане непридатним. У цей момент найбільш правильним рішенням було б передати таку техніку в іншу організацію (наприклад, із НДІ передати у ВНЗ). Однак розбирання, транспортування, складання й налагодження такого сорту пристроїв – настільки заморочлива справа, що непогано було б її уникнути;
- дорога техніка не повинна простоювати. Але якщо не потурбуватися об'єднанням всіх суперкомп'ютерів країни в єдину систему, то цілком можливі такі ситуації, коли буде спостерігатись перевантаження (величезні черги на проведення обчислень) суперкомп'ютерів в одній частині країни й простій, недовантаження обчислювальних систем – в іншій;
- який би потужний одиночний суперкомп'ютер не був створений, завжди знайдеться завдання, яке треба вирішити саме сьогодні, і для якого потрібна більша потужність, чим продуктивність будь-якого одиночного суперкомп'ютера. При цьому сумарна продуктивність всіх існуючих сьогодні супереом цілком є достатньою для вирішення такого завдання.

Всі ці проблеми досить очевидні. А пошук їхнього розв'язання однозначно приводить до ідеї інтеграції розрізнених комп'ютерних ресурсів у єдину територіально-розподілену систему. Це і є основна ідея Грід-систем і

Грід-технологій.

1 БАЗОВІ СКЛАДОВІ GRID І РЕСУРСИ

Наразі в Україні, як і в усьому світі все більшого розповсюдження набувають розподілені інформаційні системи, які надають доступ до великих сховищ інформації й інших ресурсів (даних фізичних експериментів, моделей, курсів дистанційного навчання і т.ін.). Інформаційні технології і новітні досягнення в галузі високопродуктивних обчислень сьогодні доступні всім зацікавленим особам практично у всіх сферах людської діяльності.

Технологією, що забезпечує доступ до таких розподілених інформаційних систем, є Grid-технологія.

1.1 Основні поняття і визначення

Англійський термін – Grid (у вітчизняній літературі як термін, використовується українська калька слова – Грід), який застосовується до новітніх технологій, що бурхливо розвиваються, як не дивно не є абревіатурою, це звичайне слово англійської мови і у перекладі дослівно означає «ґрати», але в цьому конкретному випадку його трактують і застосовують у дещо іншому сенсі – «обчислювальна мережа».

Ідеї Грід-технологій запозичені із практики роботи національних електричних мереж (power Grid), які об’єднують у собі розподілених по величезних територіях споживачів електроенергії, лінії передачі потужностей і різних (різнорідних за власним устроєм) електростанцій, які генерують ці потужності (рис. 1). Споживач електроенергії обслуговується цією гігантською системою, і йому цілком байдуже, якого типу електростанція в цей час генерує потік електронів, який живить його обладнання (теплова, атомна, гідростанція або яка-небудь інша), байдуже і які лінії передачі електроенергії задіяні для цього. Електрична мережа повністю підтримує різні аспекти такого обслуговування: ефективне використання в національному масштабі наявних

потужностей, які генеруються електростанціями, перекидання надлишків потужностей з одного регіону до іншого, використання резервних ліній для нейтралізації наслідків аварій на лініях передачі електроенергії або на електростанціях і т.ін.

Рисунок 1 – Схема одержання електроенергії споживачем

За повною аналогією з енергомережею, користувач Грід так само може одержати практично будь-які обчислювальні ресурси, зовсім не турбуючись про те, яким чином і звідки він їх отримав.

Термін «GRID» в обіг був уведений Яном Фостером на початку 1998 року публікацією книги «Грід. Нова інфраструктура обчислень»[1)]. Дещо пізніше це визначення було розвинене й уточнене в його іншій книзі [2)]:

Грід – це система, яка координує розподілені ресурси за допомогою стандартних, відкритих, універсальних протоколів і інтерфейсів для забезпечення нетривіальної якості обслуговування (QoS – Quality of Service).

[1)] Foster I., Kesselman C. The Grid. Blueprint for a new computing infrastructure. San Francisco: Morgan Kaufman, 1999. 677 p.

[2)] Foster I., Kesselman C. The Grid 2: Blueprint for a New Computing Infrastructure. San Francisco: Morgan Kaufmann, 2004. 712 p.

Хоча в останні десятиліття базова ідея Грід насправді не зазнала істотних змін, але за деякими даними[1)] якихось 10–12 років потому всеосяжного визначення Грід не існувало, та й можна казати, що не існує і дотепер. Кожен з авторів надає власне визначення Грід, виходячи з того, які з властивостей системи він вважає найголовнішими.

Так наприклад, іноді зустрічається й таке, не зовсім повне й, швидше за все, не зовсім правильне визначення:

Під англійським терміном GRID (ґрати) розуміється сукупність просторово розподілених обчислювальних вузлів, зв'язаних деякою мережею для обміну даними.

У цьому визначенні не дуже правильним є використання словосполучення «деяка мережа». Насправді мається на увазі наявність глобальної мережі Інтернет між обчислювальними вузлами. Та й просторовий розподіл найчастіше передбачає географічний розподіл. І, нарешті, вузли мережі теж не завжди є обчислювальними.

Дещо правильніше сутність Грід відображає наступне визначення:

GRID – це технологія створення ефективних територіально розподілених гетерогенних мереж, які об'єднують комп'ютери із різноманітними апаратними й програмними платформами. Основне завдання GRID – реалізація гнучкого, захищеного, скоординованого обчислювального простору для спільного використання ресурсів між динамічно мінливими співтовариствами користувачів.

Ще одне визначення твердить:

Grid-система – це програмно-апаратне середовище, побудоване на основі обчислювальних пристроїв, які належать різним адміністративним доменам комунікаційної мережі і в якій дозволяється дистанційно використовувати будь-яку кількість ресурсів цих пристроїв (процесорних, оперативної і постійної пам'яті, програм і даних).

Як видно з наведених визначень вони ґрунтуються на тих або інших

[1)] Stockinger H. Defining the Grid: A Snapshot on the Current View. The Journal of Super-computing. 2007. № 42(1). P. 3–17.

властивостях характерних для Грід-технологій і іноді навіть суперечать одне одному.

Насправді не все так погано, як це може здатись на перший погляд. Не дивлячись на таке розмаїття визначень Грід-систем, достатньо лише розуміти, що зовні, як вже відмічалось, Грід-системи дуже схожі на енергетичні системи. Так само, як і енергосистеми, Грід має велику кількість ресурсів, просторово розподілених мабуть по всій земній кулі (див. рис. 2). До складу цих ресурсів в загальному випадку входять великі надпотужні суперкомп’ютери, обчислювальні кластери, кластери робочих станцій і, навіть, окремі персональні комп’ютери. До ресурсів Грід відносяться також великі сховища даних, одержаних від комплексів вимірювальних приладів, встановлених на експериментальних установках або розрахованих при проведенні складних комп’ютерних обчислювальних експериментів

Рисунок 2 − Умовне представлення структури Грід-системи

Користувачі Грід-систем, які як і споживачі електроенергії, також просторово розподілені майже по всьому світу і мають можливість використовувати ресурси систем, підключаючись до них через глобальну мережу Інтер-

нет. Підключення здійснюється не прямо, а за допомогою додаткового проміжного програмного забезпечення (ППЗ), яке призначене для управління функціонуванням Грід-систем.

Таким чином можна казати, що на базовому рівні визначаються служби, які забезпечують безпосередній доступ до ресурсів, використання яких розподілене за допомогою протоколів Грід.

Поєднуючи всі вище наведені визначення Грід зі схематичним розумінням Грід-структури можна назвати такі основні властивості Грід-систем.

По-перше – обчислювальні ресурси надають користувачу (точніше кажучи, завданню користувача) процесорні потужності Грід-системи. Обчислювальними ресурсами можуть бути як кластери, так і окремі робочі станції. При всій розмаїтості архітектур будь-яка обчислювальна система може розглядатися як потенційний обчислювальний ресурс Грід-системи.

По-друге – ресурси пам’яті Грід-системи надають загальний простір для зберігання даних. А доступ до ресурсів пам’яті можливий лише за умов використання програмного забезпечення проміжного рівня, яке реалізує уніфікований інтерфейс управління й передачі даних.

По-третє – інформаційні ресурси й каталоги, як окремі складові Грід-систем, є особливим видом ресурсів пам’яті. Вони призначені для зберігання й надання метаданих і інформації про інші ресурси Грід-системи.

І останнє – глобальний мережний ресурс є сполучною ланкою між розподіленими ресурсами Грід-системи. Основною обов’язковою характеристикою мережного ресурсу є можливість швидкісної передачі даних.

1.2 Відмінності і спільні риси Грід, розподілених систем і кластерів

Слід відзначити, що іноді помилково ототожнюють такі поняття як розподілені системи і Грід.

Насправді є два базових критерії, застосовуючи які, завжди можна виділити Грід-системи серед інших систем, які також забезпечують розподілений доступ до ресурсів:

– Грід-система координує розрізнені ресурси, тобто ресурси не мають загального центра управління, а Грід-система займається координацією їхнього використання, наприклад, балансуванням завантаження.
– Грід-система будується на базі стандартних і відкритих протоколів, сервісів і інтерфейсів, таким чином, без наявності стандартних протоколів не можна легко й швидко підключати нові ресурси до Грід-системи, розробляти нові види сервісів і т.ін.

Іноді не тільки прості користувачі, а й деякі фахівці, не враховуючи цих двох критеріїв, помилково ототожнюють такі поняття як розподілений кластер і Грід. Хоча на перший погляд це дуже схожі обчислювальні системи, насправді вони істотно відрізняються за принципами своєї побудови і функціонування.

Насамперед кластери за своєю сутністю не є насправді просторово розподіленими обчислювальними системами, хоча і зустрічаються деякі кластери корпоративних установ, складові яких географічно розташовані доволі віддалено одна від одної. Але це не головне.

Основне, що відрізняє Грід від кластера – це те, що Грід складається з різноманітних гетерогенних ресурсів, які працюють на повністю несумісних апаратних платформах від різних виробників під управлінням операційних систем, які істотно відрізняються між собою. В цьому разі програмне забезпечення таких фірм виробників ПО як IBM, Platform Computing, DataSynapse і United Devices дозволяє розподіляти робоче навантаження на різних типах комп’ютерів з різними конфігураціями. Грід комп’ютинг інтегрує у собі зовнішню пам’ять, мережні засоби, комп’ютери, програмні ресурси і дані.

Навпаки, кластерний комп’ютинг здебільшого концентрується на обчислювальних ресурсах. У кластері найчастіше містяться однотипні комп’ютери, під управлінням однієї операційної системи, виняток іноді робиться для великих кластерів робочих станцій.

За своїм походженням і ідеологією Грід є динамічна система з постійним підключенням і відключенням ресурсів. Грід-системі властива просторова розподіленість через засоби локальної, регіональної або глобальної мере-

жі. Складові Грід можуть розташовуватись і дійсно розташовуються де завгодно. Сама концепція Грід передбачає можливість динамічного розширення і масштабування. Так, наприклад, IBM, United Devices і інші партнери в 2003 році провели експеримент за допомогою Грід[1)], присвячений підтримці процесу ідентифікації перспективних лікарських сполук для лікування віспи. В експерименті були задіяними понад два мільйони персональних комп'ютерів. В разі застосування традиційних методів комп'ютерного експерименту для виконання досліджень знадобилось би декілька років. Із застосуванням Грід-технологій на все про все знадобилось «лише» шість місяців. Можна припустити, що при залученні до проекту не 2, а, наприклад, 20 мільйонів комп'ютерів експеримент зайняв би вже не місяці, а тижні і ймовірно дні і, навіть, години.

В свою чергу відомо, що кластери в більшості містять фіксовану кількість процесорів і супутніх ресурсів. Кластери, частіш за все (за винятком кластерів робочих станцій), реалізуються у вигляді пулу фізично зв'язаних комп'ютерів, які розташовується в одному приміщенні. При цьому кластерна технологія зв'язку передбачає вкрай жорсткі вимоги до латентності системи міжз'єднання, що, в свою чергу, накладає сильні обмеження на взаємне розташування кластерів. Тим самим, фізична близькість компонентів кластера й високі вимоги до латентності засобів комунікації вузлів кластера обмежують його масштабованість.

Зі сказаного можна зробити однозначний висновок: проста система управління ресурсами кластера не є системою Грід, оскільки здійснює централізоване управління всіма вузлами даного кластера, маючи повний доступ до них. Грід-системи мають лише обмежений доступ до ресурсів, який повністю залежить від політики того адміністративного домену (організації-власника), в якому цей ресурс знаходиться.

З іншого боку кластери й Грід-вирішення на практиці цілком взаємодо-

---

[1)] IBM, United Devices and Accelrys Aid U.S. Department of Defense in Search for Smallpox Cure. Smallpox Research Grid Project to Link More Than Two Million Computers. https://www-03.ibm.com/press/us/en/pressrelease/335.wss (дата звернення 15.04.2019)

повнюють і в повній мірі відповідають одне одному. Це можливе тому, що переважна більшість існуючих у світі Грід-реалізації серед своїх ресурсів обов'язково містять й кластери. Дійсно, користувач Грід може навіть й не знати, що його завдання фактично виконується на якомусь віддаленому кластері. І хоча між Грідами й кластерами існують вказані вище розбіжності, вони практично взаємозалежні між собою. Таке твердження витікає з того факту, що розв'язання великої кількості завдань з певними властивостями вимагає використання сильно зв'язаних між собою процесорів. І тому для кластерів завжди буде місце в Грідах.

Проте, різке зростання ефективності мережних технологій, призводить до того, що проблеми, які раніше були характерні лише для кластерів і розглядалися тільки у контексті їхньої побудови, сьогодні вже повинні знаходити власне вирішення і в Грід-комп'ютингу. Розуміння цього винятково важливо для встановлення балансу між природною масштабованістю Грід й можливостями високошвидкісних міжпроцесорних з'єднань, які властиві кластерам.

Таким чином можна казати і слід чітко розуміти, що всі системи, в яких з використанням стандартних, відкритих і універсальних протоколів і інтерфейсів здійснюється координація ресурсів, в звичайному стані не керованих централізовано, і робиться це для надання нетривіальної якості обслуговування, відносяться до Грід і насправді є Грід-системами.

В свою чергу кластери, мережні сховища даних, наукові прилади та й сама мережа зі всім її обладнанням є дуже важливими компонентами Грід, але самі по собі не є ним.

До сказаного вище можна додати ще декілька важливих особливостей, які зазвичай властиві Грід-системам:

- гнучкість, тобто потенціальна можливість забезпечення розподіленого доступу до будь-яких видів ресурсів (не тільки обчислювальних);
- вже згадувану вище масштабованість – працездатність Грід-системи при значному збільшенні або зменшенні її складу;
- гнучка й потужна підсистема безпеки – стійкість до атак зловмисни-

ків, забезпечення повної конфіденційності;

– можливість контролю над ресурсами, які підлягають розподіленню, що забезпечується застосуванням локальних і глобальних політик і квот;
– гарантії якості обслуговування;
– можливість одночасної, скоординованої роботи з декількома ресурсами.

1.3 Основні ресурси Грід

Хоча Грід-технології, як такі, насправді й не прив'язані до певних ресурсів, але їх можна поділити на декілька груп, з якими існуючі реалізації Грід-систем найчастіше забезпечують користувачам можливість роботи.

По-перше – це обчислювальні ресурси, до яких, як вже відомо, відносяться суперкомп'ютери, різноманітні кластери і в багатьох випадках окремі робочі станції на основі звичайних персональних комп'ютерів. Необхідною умовою для коректної роботи ресурсів цієї групи є наявність спеціального програмного забезпечення, яке називають програмним забезпеченням проміжного рівня (middleware). Проміжне ПО реалізує стандартний зовнішній інтерфейс з ресурсом і дозволяє зробити ресурс доступним для Грід-системи. Основною характеристикою обчислювального ресурсу є його продуктивність.

По-друге – це ресурси збереження даних (ресурси пам'яті), до їхнього складу входять диски і дискові масиви, накопичувачі на магнітних стрічках, системи масового збереження даних. Ресурси пам'яті надають користувачам простір для збереження даних. Доступ до ресурсів пам'яті також можливий лише завдяки програмному забезпеченню проміжного рівня. В цьому разі проміжне ПО реалізує уніфікований інтерфейс управління даними і їхню передачу. Так само, як і в випадку з обчислювальними ресурсами, фізична архітектура ресурсу пам'яті не має ніякого значення для функціонування Грід-системи, будь то жорсткий диск на робочій станції або система масового збе-

реження даних на сотні терабайтів. Основною характеристикою ресурсу пам’яті є його обсяг.

Окремим особливим різновидом ресурсів пам’яті є інформаційні ресурси і каталоги. Інформаційні ресурси дозволяють у структурованому вигляді зберігати великі обсяги інформації про стан Грід-системи на кожному конкретному відрізку часу і ефективно виконувати завдання пошуку необхідних ресурсів і даних.

До наступної групи ресурсів відносяться мережні ресурси. Це різноманітні канали зв’язку, починаючи зі звичайних локальних обчислювальних мереж і закінчуючи супутниковими каналами глобальної мережі, з відповідним супутнім до них обладнанням. Мережний ресурс є ланкою, яка сполучає розподілені ресурси Грід-системи проміж собою. Швидкість пересилання даних визначає основну характеристику мережного ресурсу. Географічно розподілені системи на базі існуючих мережних технологій можуть з’єднувати, сполучати тисячі ресурсів абсолютно різного типу, незалежно від їхнього розташування у географічному просторі.

Ще одну велику групу ресурсів Грід складає програмне забезпечення – це спеціалізоване, іноді унікальне ПО, яке призначене для виконання яких-небудь спеціальних проблемно-орієнтованих або звичайних, але дуже складних обчислень.

Таким чином, доходимо висновку, що інфраструктура Грід з одного боку основана на наданні ресурсів в загальне користування, а з іншого боку на використанні відкрито доступних ресурсів. Цілком зрозуміло, що в цьому випадку хтось і якимось чином повинен надавати всі ці ресурси. І знову ж таки не важко здогадатись, що таку велику купу абсолютно різноманітних ресурсів не може надати жодна навіть найбагатша організація чи установа: навчальна, дослідна або промислова, приватна або державна.

1.4 Поняття віртуальної організації

Такий стан справ неминуче наштовхує на думку про об’єднання ресур-

сів окремих власників у єдине ціле. Й насправді, в цім плані ключовим поняттям інфраструктури Грід є поняття так званої віртуальної організації (ВО).

Той самий Ян Фостер ще в одній із засадничих робіт[1)], відмічаючи, що реальною і конкретною проблемою, яка підкреслює особливу значущість Грід-концепції, є узгоджене розподілення ресурсів і вирішення задач в динамічних, багатопрофільних віртуальних організаціях, надає таке визначення ВО:

Об'єднання окремих спеціалістів і/або інститутів, які дотримуються певних правил розділення ресурсів, утворює те, що ми називаємо віртуальною організацією (virtual organization) – ВО.

Причому під розділенням ресурсів перш за все мається на увазі не обмін файлами, а безпосередньо прямий доступ до комп'ютерів, програмного забезпечення, даних і інших ресурсів так, як це вимагається низкою виникаючих в промисловості, науці й техніці стратегій спільного вирішення задач і посередництва в наданні ресурсів. Такий розподіл обов'язково повинен легко контролюватись не тільки провайдерами ресурсів, а й їхніми споживачами. Обидві сторони учасників Грід повинні ясно і чітко обумовлювати і знати, що розділяється, кому дозволений розподіл і умови, на яких виконується цей розподіл.

Іншими словами – інфраструктура Грід базується на віртуальній організації, в яку кооперуються як споживачі, так і власники ресурсів. В існуючих Грід-системах віртуальна організація являє з себе об'єднання спеціалістів в якій-небудь прикладній галузі, які об'єднаються для досягнення загальної мети.

Будь-яка ВО має в своєму розпорядженні певну кількість ресурсів, які надаються у загальне користування зареєстрованим в ній учасникам. Причому деякі ресурси можуть одночасно належати одразу декільком ВО. Кожна з ВО самостійно встановлює правила роботи для власних учасників, виходячи

[1)] Foster I., Kesselman C., Tuecke S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of High Performance Computing Applications, 15 (3). 2001. P. 200–222.

з підтримання балансу між потребами користувачів і наявністю відповідного обсягу ресурсів. Тому користувач обов’язково повинен обґрунтувати своє бажання працювати у середовищі Грід-системи і одержати на це згоду управляючих органів ВО.

Відношення між різними ВО і фізичними організаціями, які є їхніми учасниками можна проілюструвати таким рисунком (рис. 3).

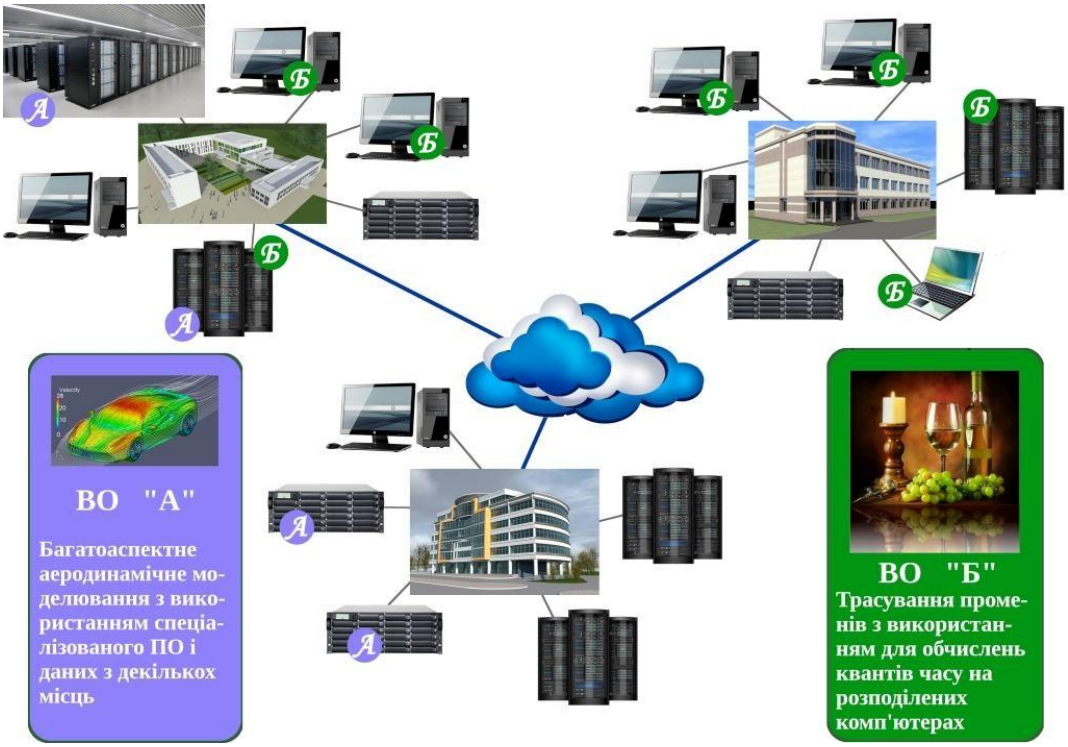

Рисунок 3 − Приклад можливого використання ресурсів Грід різними віртуальними організаціями

Зовсім не обов’язково, щоб кожна окрема реальна організація була учасником лише однієї ВО. Кожна реальна організація може брати участь одразу в декількох ВО, надаючи деяку кількість або всі власні ресурси для розділення їх у колективне застосування у Грід. На наведеному рисунку показані три реально існуючі наукові, або якогось іншого напрямку дії, організації і дві ВО, до яких вони об’єднались. ВО під літерою «А», об’єднує учасників консорціуму розробників автомобілів. А ВО під літерою «Б» зв’язує колег, які займаються обчисленням трасування променів світла при розробці, напри-

клад, сцен в створюваних комп'ютерних фільмах. Причому учасники цієї ВО згодні розділяти вільні кванти комп'ютерного часу ресурсів, переданих для користування в Грід. Реальна організація, яка показана у верхній частині рисунка ліворуч, бере участь в ВО під літерою «А», організація праворуч бере участь в ВО під літерою «Б», ну й нарешті, остання реальна організація є членом в «А» так само, як і в «Б», тобто одразу в двох ВО. Політики, які резюмуються в «квотах» і регулюють доступ до ресурсів залучених до загального процесу, відрізняються відповідно до реальних організацій учасників, ресурсів виділених ними у Грід і ВО їх об'єднуючих.

На рис. 3 видно, що учасники ВО «А» можуть на власному суперкомп'ютері запустити і виконувати деяку складну обчислювальну програму, а іншу програму виконувати на власному кластері, одночасно надаючи на ньому кванти часу для виконання завдання учасників ВО «Б». Необхідні вихідні дані для виконання обох програм і результати їхньої роботи учасники ВО «А» можуть одержувати і зберігати у сховищах даних, які належать третій фізичній організації, розташованій на значному географічному віддаленні від замовника.

В свою чергу, учасники ВО «Б», якщо їх бюджет на віддалених ресурсах не вичерпаний, можуть використовувати ресурси незайнятих комп'ютерів, які у даний час простоюють, і, навіть, поділяти вільні кванти часу на віддаленому кластері спільно з учасниками ВО «А».

Відносини розділення можуть з часом динамічно змінюватись у залежності від умов застосування виділених ресурсів, характеру дозволеного доступу і складу учасників, яким дозволений доступ. Зрозуміло, що ці відношення зовсім не обов'язково пов'язані з необхідністю явного поіменного зазначення учасників, але вони повною мірою можуть неявно визначатись через політики управління доступом до ресурсів. Можна казати, що фактична організація може забезпечити доступ будь-кому, хто зможе доказати, що він – «замовник» або, наприклад, «студент».

Такий стан справ щодо динамічного характеру відносин розділення, означає, що у загальному випадку конче необхідні механізми, здатні виявля-

ти і характеризувати сутність зв'язків, які існують в даний конкретний відрізок часу. Наприклад, новий учасник, який вступає до ВО «Б», повинен мати змогу визначитись, до яких ресурсів він може отримати доступ, визначити «якість» цих ресурсів і мати обізнаність з політиками управління доступом.

Відношення розділення в Грід-системах не лише представляються широко розповсюдженою, як в локальних, так і в глобальній мережах, схемою відносин «клієнт – сервер». Тут відношення доволі часто відповідають достатньо розвинутій одноранговій структурі типу «точка – точка», в якій провайдери можуть виявитися також й користувачами, а відношення розділення ресурсів можуть створюватись й існувати між будь-якою підмножиною учасників. Відношення розділення в віртуальних організаціях можуть об'єднуватись задля погодженого використання на багатьох ресурсах, власником кожного з яких виступає та або інша реальна організація. Наприклад, обчислення, виконувані ВО «Б», які з початку були розділені на якійсь конкретний обмежений пул обчислювальних ресурсів, пізніш можуть отримати доступ до даних, збережених десь на віддаленому сховищі даних або запустити розв'язання іншої підзадачі на якій-небудь множині комп'ютерів, яка на даний момент часу виявилась вільною. В таких випадках на перший план висувається здатність системи контрольованими засобами делегувати нові повноваження завданням, які їх зажадали. Так само це стосується й механізмів узгодження операцій (так званого, спільного планування – cosheduling) на всій множині ресурсів.

Спосіб використання і мета застосування того ж самого ресурсу в повній мірі залежать від обмежень, які політиками використання накладаються на розділення ресурсів. Наприклад, одна з можливих систем відносин передбачає використання комп'ютера лише для прогону якогось наперед визначеного блока програмного забезпечення. В той самий час інша система відносин вже передбачає, що часові кванти комп'ютера можуть розділятися для будь-якого обчислення. Умови, які накладаються на розділення або використання того чи іншого ресурсу, в значній мірі залежать не тільки від політик віртуальних організацій з використання ресурсів, але і від багатьох зовнішніх

факторів. Так власне, якість обслуговування – одна з основних характеристик Грід, складовими частинами якої є показники продуктивності, імовірнісні характеристики й обмеження на використання ресурсів тощо, через відсутність апріорних знань про можливе використання якогось ресурсу не завжди повністю задовольняє користувачів.

Всі описані характеристики і вимоги (та й багато інших) визначають те, що за формулюванням Фостера й називають віртуальною організацією. Концепція ВО від першого їх появлення виявилась і по сьогодні остається фундаментальною практично для всіх сучасних технологій обчислення і обробки даних. ВО дозволяють групам організацій і/або окремим користувачам, які суттєво відрізняються між собою, контрольовано поділяти ресурси, таким чином, щоб вони могли співробітничати задля досягнення деякої загальної мети.

1.4.1 Основні вимоги до віртуальних організацій

Безмежне розмаїття задач, які підлягають розв’язанню за допомогою Грід-технологій, спричинило до того, що створювані ВО надзвичайно різноманітні за метою, яка ставиться перед ними, за масштабом, розміром, тривалістю існування, структурою, за суспільним і соціологічним статусом. Проте, ретельне вивчення визначальних технологічних вимог, які висуваються до ВО, надає можливість виявити, сформулювати і провести класифікування широкого кола загальних для всіх ВО інтересів і потреб. Так, аналізуючи сказане вище, можна дійти таких висновків щодо вимог, які необхідно забезпечити для розділення ресурсів.

ВО в першу чергу повинні забезпечити:

– надзвичайно гнучкі відношення в широкому діапазоні можливих мережних вирішень: від стандартної структури «клієнт – сервер» до не менш стандартної схеми «однорангова мережа» («точка – точка»);
– складний і високорівневий контроль за використанням учасниками ВО поділюваних ресурсів, включаючи засоби дрібноструктурного

контролю доступу індивідуальних користувачів, делегування і застосування локальних і глобальних політик;

– можливість розділення різноманітних, докорінно відмінних за своєю сутністю ресурсів: від програм, файлів і даних до комп'ютерів, датчиків і мереж;
– різноманітні за критеріями продуктивності й вартості користувальницькі режими (від однокористувальницького до багатокористувальницького), вирішення проблем, які пов'язані із забезпеченням необхідної якості обслуговування користувачів і планування виконуваних завдань, а також спільного завантаження ресурсів й обліку їхнього використання.

На завершення є сенс, з метою чіткого окреслення, звернути увагу на різницю, яка існує між поняттями Грід-технології і Грід-системи, точніше реалізацією останніх.

Грід-технологія складається і базується лише на найбільш загальних й універсальних структурах і аспектах, які є абсолютно однаковими для будь-якої системи. Це архітектура системи, протоколи взаємодії між її складовими, інтерфейси, за допомогою яких здійснюються зв'язки між елементами системи і з зовнішнім світом, і сервіси, які надаються користувачу системою. В разі виникнення потреби, можна, застосовуючи Грід-технологію і наповнюючи її новим змістом в кожному окремому випадку, створити реалізацію тієї чи іншої Грід-системи. Кожна з таких Грід-систем буде призначена для вирішення деякого спеціалізованого класу прикладних задач. Наприклад, деяку окрему Грід-систему можна розглядати як, так званий, метакомп'ютер з великою кількістю обчислювальних вузлів. Завдяки цьому, багато хто з фахівців доволі часто ототожнює Грід-технологію лише з технологією паралельних обчислень. Звісно, що на такому метакомп'ютері за допомогою стандартних технологій паралельного обчислення таких, як PVM або MPI, можна реалізувати, і на практиці часто-густо реалізують, подібні досить складні модельні обчислення, які потребують багато процесорного часу. Однак, ні в якому разі не слід плутати Грід-технологію з технологією паралельних обчислень

тому, що друга є тільки однією зі складових першої, найважливішою задачею якої є координація використання ресурсів Грід.

1.5 Контрольні запитання і завдання для самостійної роботи

1. Що означає термін «Grid» («Грід»), хто першим і коли ввів його в обіг? Надайте авторське визначення терміна.
2. Наведіть низку визначень терміна «Грід», поясніть чим вони відрізняються між собою, чим схожі, як доповнюють і уточнюють одне одного.
3. Перелічте ресурси, які є складовими Грід. Як вони зв’язані проміж собою і де вони знаходяться? Навіщо потрібна така складова Грід, як проміжне програмне забезпечення?
4. Назвіть і охарактеризуйте основні властивості Грід-систем.
5. Як пов’язані між собою Грід-системи і кластери, які в них спільні риси і чим вони вирізняються? Назвіть базові критерії, за якими можна розпізнати Грід-системи серед інших систем, які також забезпечують розподілений доступ до ресурсів.
6. Сформулюйте і поясніть всі важливі особливості, які властиві Грід-системам.
7. Які групи ресурсів найчастіше забезпечують користувачам можливість роботи з існуючими реалізаціями Грід-систем?
8. Що таке віртуальна організація, яке визначення ВО надав Ян Фостер? Надайте докладне тлумачення цього визначення.
9. Охарактеризуйте відношення, які виникають між різними ВО і фізичними організаціями.
10. Від чого залежать спосіб використання і мета застосування того чи іншого ресурсу ВО?
11. Які вимоги накладаються на ВО з точки зору необхідності забезпечення розділення їхніх ресурсів?

2 ЗВ'ЯЗОК ГРІД І ВЕБ-ТЕХНОЛОГІЙ. ПРОГРАМНЕ ГРІД-ЗАБЕЗПЕЧЕННЯ

Хоча, як видно з рис. 2, Грід-системи існують і функціонують в середовищі глобальної мережі Інтернет, але так само як не можна ототожнювати і плутати Грід і кластери, так ні в якому разі не можна вважати, що Грід – це просто складова частина Інтернету, або уявляти собі, що Грід-технології – це хоча й цілком нова, але все-таки просто гілка веб-технологій.

2.1 Інтернет і веб-сервіси як складові Грід

Як це не дивно на перший погляд, насправді все навпаки: Інтернет є лише одним із загальних ресурсів Грід, а веб-технології – підпорядкована, хоча й десь у модернізованому вигляді, складова частина Грід-технологій.

Доказом того, що Грід не є Інтернетом, може бути той факт, що Інтернет створювався як глобальна система мереж, яка об'єднувала й до сьогодні об'єднує безліч хостів, і яка дозволяє їм взаємодіяти один з одним за допомогою відповідних протоколів. І взагалі, місія Інтернет полягає в глобалізації обміну інформацією. Завдяки мережі Інтернет її користувачі можуть за допомогою своїх персональних комп'ютерів обмінюватися наявною в них інформацією, а також одержувати доступ до готової інформації, яка розміщена в мережі. Обробка й зберігання необхідної користувачу інформації відбувається на його персональному комп'ютері.

З іншого боку Грід – це спосіб спільного скоординованого використання розподілених ресурсів. Ідеологія мереж Грід будується на іншому підґрунті: користувач обробляє інформацію за допомогою розподілених потужностей Грід-мережі і зберігає її на розподілених серверах. Персональний комп'ютер використовується тільки як пристрій уведення й виведення даних для підключення до мережі Грід. Користувач не знає (і, навіть, не замислюється), який саме сервер обробляє його інформацію і на якому саме сервері вона зберігається. Таким чином, мережі Грід надають своїм користувачам

послуги з віддаленої обробки і зберігання даних.

В свою чергу Грід – не є й Web тому, що всесвітня павутина WWW стандартизувала пошук і доставку документів. За великим рахунком, WWW лише надає користувачеві глобальний доступ до текстової, графічної і інших типів інформації. У мережі Internet можливо й віддалене зберігання даних: наприклад, безкоштовні поштові сервери (www.gmail.com тощо) надають можливість зберігання на їхньому дисковому просторі переписки користувача, на сайті www.youtube.com пропонується зберігання відеокліпів. Відмінність мереж Грід від цих видів послуг полягає в тім, що сайти Інтернет використовують опосередковану модель зберігання даних: спочатку користувач повинен зареєструватися на відповідному сайті й одержати доступ до певного обсягу дискового простору, а надалі йому доведеться щораз, коли він побажає розмістити на цьому просторі нову інформацію і/або видалити стару, спочатку заходити на цей сайт, уводити свої пароль і ім’я користувача, і тільки після цього він зможе приступити до управління даними, причому на серверах можливе тільки зберігання інформації – її обробка (якщо в ній виникає необхідність) виробляється самим користувачем за допомогою свого комп’ютерного ресурсу (як правило – персонального комп’ютера).

Грід же забезпечує глобальний практично не обмежений доступ до обчислювальних ресурсів і до ресурсів зберігання будь-яких даних. Мережа Грід, так би мовити, емулює персональний комп’ютер: підключившись до мережі, користувач відразу одержує доступ до власної інформації і може негайно почати працювати з нею, використовуючи тільки обчислювальні ресурси Грід.

Інакше кажучи, Грід став наступним кроком, наступним етапом в низці революційних перетворень – стандартизації і глобалізації використання всіх видів комп’ютерних ресурсів. Розподіл ресурсів, у якому зацікавлені розробники, вони же користувачі Грід, це не просто обмін файлами, а прямій доступ до комп’ютерів (зовсім необов’язково серверів), до програмного забезпечення, до даних і до інших ресурсів, які потрібні для спільного вирішення завдань і стратегій управління ресурсами, що виникають у промисловості, нау-

ці й техніці.

Підсумовуючи сказане, можна зробити висновок про те, що не зважаючи на значні принципові відмінності між веб-сервісами, які на власному устаткуванні і програмному забезпеченні підтримують виконання функцій, замовлених користувачами і Грідом, який надає замовнику обчислювальні потужності для вирішення його задач, самі по собі механізми веб-сервісів є базисом, основою Грід.

2.2 Проміжне програмне забезпечення і його функції

Але основа – це лише й є основа. Для повноцінної підтримки всієї архітектури Грід потрібно мати деяку більш розвинуту технологію, яка безпосередньо виконує координацію всіх доступних ресурсів. Роль такої технології виконує так зване проміжне програмне забезпечення (ППЗ) Грід (за англійською термінологією – middleware). Проміжним програмним забезпеченням в повному обсязі підтримуються:

– архітектура Грід;
– інструменти розробки програмного забезпечення (SDK);
– інтерфейси прикладного програмування (API).

На рівні архітектури Грід, ППЗ призначено для виконання цілої низки функцій, до яких входить: управління обчислювальними завданнями і даними у вигляді окремих файлів, структурування інформації в базах даних і управління ними, загальна підтримка функціонування Грід-системи.

Як вже указувалось, вся структура Грід характеризується великим ступенем розподіленості, як програмного забезпечення, так і робочих місць користувачів, а також і використовуваних комп’ютерних ресурсів. Таке становище приводить до того, що виконання операцій в Грід-системі ініціюється шляхом формування запиту до якої-небудь служби і, як правило, здійснюється послідовністю взаємодіючих служб, кожна з яких реалізує відповідну, характерну лише для неї, частину запитання.

2.3 Архітектура Грід з точки зору програмного забезпечення

При описі різних рівнів архітектури Грід часто використовується модель «пісочного годинника» (рис. 4). Горловина цього умовного годинника відповідає невеликій кількості базових абстракцій і протоколів (таких як TCP і HTTP в Інтернеті), які необхідні для здійснення двосторонніх зв'язків між високорівневими і низькорівневими службами Грід-систем.

Рисунок 4 − Модель Грід, як «пісковий годинник»

Такий підхід практично повністю відповідає базовій еталонній моделі взаємодії відкритих систем, тобто мережній моделі стека мережних протоколів OSI/ISO (ДСТУ ISO/IEC 7498:2004). Як відомо, модель OSI є еталонною моделлю, яку повинні підтримувати всі розробники мережних програм і виробники мережного обладнання з метою забезпечення сумісності своїх продуктів.

Виходячи з цього, на найнижчому рівні архітектура Грід базується на стеку протоколів, подібних до стека моделі OSI. Рис. 5 як раз ілюструє відповідність між цими стеками обміну інформацією. На кожному із представлених рівнів існують свої сервіси, які взаємодіють за допомогою певних протоколів.

Рисунок 5 − Приблизна відповідність рівнів стека протоколів Грід (ліворуч) стеку протоколів Інтернет моделі OSI (праворуч)

У такій моделі кожен рівень призначається для вирішення вузького кола завдань і використовується для надання послуг більш високим рівням. Верхні рівні моделі максимально наближені до користувача і працюють з найбільш абстрактними об’єктами, В свою чергу нижні рівні сильно залежать від фізичної реалізації Грід-ресурсів.

З рисунка видно, що з семи рівнів моделі OSI в архітектурі Грід задіяні лише чотири. Але з іншого боку видно, що прикладний рівень моделі OSI повністю відповідає прикладному і колективному рівням протоколів Грід, а частково ще й ресурсному. Так само транспортний рівень частково відповідає відразу двом рівням – ресурсному і єднальному.

Таким чином, стек Грід-протоколів складається з п’яти рівнів кожен, з яких виконує власну частку роботи з обслуговування функціонування Грід-системи в цілому.

2.3.1 Апаратний рівень стеку Грід-протоколів

Так апаратний рівень (Fabric Layer) відповідає за управління локальними ресурсами і складається з протоколів, за допомогою яких відповідні слу-

жби безпосередньо працюють з ресурсами. Апаратний рівень забезпечує спільний доступ розподілених ресурсів і надає ті ресурси, доступ до яких забезпечується через протоколи Грід і потрібен протоколам більш високого рівня.

Раніш вже говорилось, що спектр можливих ресурсів Грід дуже широкий – це можуть бути комп'ютери, пристрої масового зберігання даних, каталоги, мережні ресурси і т.ін.

Для доступу до обчислювальних ресурсів потрібні деякі механізми для запуску програми й моніторингу її виконання, механізми для визначення апаратних і програмних характеристик, а також для визначення поточного стану (наприклад, робочого навантаження).

Доступ до ресурсів зберігання здійснюється, як за допомогою тих самих механізмів для запуску програми й моніторингу її виконання, так і цілою низкою інших механізмів. До цих механізмів відносяться такі, як: механізми для відправлення/одержання файлів, у тому числі й високошвидкісні багатопоточні способи роботи з файлами; механізми для читання/запису підмножин файлів; обов'язкова наявність механізмів для здійснення віддаленої вибірки даних і механізмів менеджменту, які дозволяють контролювати ресурси, призначені для пересилання даних. В цю групу також входять механізми, які дозволяють дізнатися про апаратні і програмні характеристики й поточне навантаження і на сам кінець механізми поліпшеного резервування даних.

Доступ до мережних ресурсів потребує механізмів менеджменту й контролю над ресурсами, призначеними для мережного трафіка (пріоритети, резервування каналів) і функції визначення характеристик і завантаження мережі.

Для доступу до ресурсів репозиторія кодів програм потрібен механізм управління версіями вихідного і об'єктного коду (наприклад, система Concurrent Versions System (CVS)).

Робота з ресурсом каталогів потребує механізму забезпечення зберігання даних, обробки запитів і підтримання актуальності даних шляхом їхнього оновлення.

При цьому ресурс може бути логічною сутністю (наприклад, розподі-

леною файловою системою) або фізичною (наприклад, кластером комп'ютерів). Реалізація такого ресурсу може містити внутрішні протоколи (наприклад, такі, як Network File System (NFS) або протокол управління кластером), однак, слід відзначити, що подібні протоколи не входять як складові в стек Грід. Компоненти апаратного рівня реалізують локальні операції, які є специфічними для кожного конкретного ресурсу (логічного або фізичного). Цей рівень за власними функціями практично аналогічний канальному рівню моделі OSI і, по суті, є набором інтерфейсів для управління локальними ресурсами.

2.3.2 Єднальний рівень стеку Грід-протоколів

Призначенням єднального рівня (Connectivity Layer) є забезпечення комунікацій і безпеки. Рівень визначає базові комунікаційні й ідентифікаційні протоколи, необхідні для проведення специфічних для Грід операцій (транзакцій) і складається з протоколів, які забезпечують обмін даними між компонентами базового рівня і протоколами автентифікації.

Комунікаційні протоколи дозволяють здійснювати обмін даними між ресурсами апаратного рівня. Комунікаційні протоколи єднального рівня повинні забезпечувати надійний транспорт і маршрутизацію повідомлень, а також привласнення імен об'єктам мережі. Ідентифікаційні протоколи автентифікації цього рівня, ґрунтуючись на комунікаційних сервісах, надають захищені криптографічні механізми для ідентифікації і верифікації автентичності як користувачів, так і ресурсів.

Незважаючи на існуючі альтернативи, зараз в Грід-системах протоколи єднального рівня передбачають використання тільки стеку протоколів TCP/IP, зокрема: на мережному рівні – IP і ICMP, транспортному рівні – TCP, UDP, на прикладному рівні – HTTP, FTP, DNS, RSVP. З огляду на бурхливий розвиток мережних технологій, в наступному єднальний рівень, можливо, буде залежати і від інших протоколів.

Для забезпечення надійного транспорту повідомлень в Грід-системі по-

винні використовуватися механізми, які передбачають гнучкий підхід до безпеки комунікацій (можливість контролю над рівнем захисту, обмеження делегування прав, підтримка надійних транспортних протоколів). У цей час ці механізми ґрунтуються як на існуючих стандартах безпеки, раніш розроблених для Інтернет (SSL, TLS), так і на нових розробках. Розв'язання проблеми безпеки на єднальному рівні мусить базуватися на існуючих стандартах, що забезпечує одноманітність засобів безпеки у різних Грід-системах.

Як відомо, ідентифікація користувача може здійснюватись різними способами. Зокрема, може використовуватись, так зване, однократне реєстрування. При використанні цього методу користувачі повинні реєструватися в обчислювальному середовищі тільки один раз. Тим самим, після успішного завершення операції, користувачі одержують відповідний доступ до ресурсів апаратного рівня.

Можливе й застосування іншого способу ідентифікації – делегування. В цьому випадку користувач авторизується на деякій множині ресурсів і програма користувача повинна мати доступ до них. Бажано, щоб така програма була здатна передати підмножину своїх прав іншій програмі.

Природно, що кожен провайдер ресурсів може мати ряд власних локальних рішень з підтримки і забезпечення безпеки свого ресурсу, який він передає у спільне користування. В цьому разі система безпеки обчислювального середовища спонукається і повинна вміти взаємодіяти із цим класом локальних рішень.

На найвищому рівні інфраструктура підтримки безпеки ресурсів містить централізовану видачу сертифікатів, управління сертифікатами і ключами тощо.

2.3.3 Ресурсний рівень стеку Грід-протоколів

Наступний – ресурсний рівень (Resource Layer) призначений для спільного використання ресурсів. Рівень є ядром багаторівневої системи, протоколи цього ядра взаємодіють з ресурсами Грід, використовуючи уніфікований

інтерфейс і ні якім чином не розрізняють архітектурні особливості конкретного ресурсу.

Протоколи ресурсного і єднального рівнів є базовими, вони, так би мовити, утворюють «горлечко» у моделі «пісочного годинника» (рис. 4) і повинні обмежуватися невеликою, але добре продуманою безліччю. Ці протоколи проектуються таким чином, щоб дозволити спільне використання різнорідних ресурсів, але при цьому вони не повинні занадто обмежувати припустимі типи протоколів вищого рівня і їхню продуктивність.

Ресурсний рівень обумовлює протоколи для:

– здійснення безпечного обміну інформацією;
– ініціалізації, моніторингу й виконання спільних операцій на індивідуальних ресурсах;
– створення облікових записів користувачів;
– здійснення обліку утилізованого часу для кожного з користувачів.

На рівні за допомогою власних і комунікаційних, і автентифікаційних протоколів, які утворюють нижчерозташований єднальний рівень, виконується узгодження методів безпеки, ініціалізація і моніторинг ресурсів і управління ними. Для доступу до локальних ресурсів і подальшого управління, ресурсний рівень викликає відповідні функції апаратного рівня. Слід відзначити, що протоколи ресурсного рівня призначені винятково для роботи з локальними ресурсами, вони не враховують глобальний стан системі. Цим повинен займатись колективній рівень, який розташовується вище.

За своїм призначенням протоколи ресурсного рівня групуються у два класи:

– інформаційні протоколи, призначенням яких є отримання інформації про структуру і стан ресурсу, його конфігурації, про поточне навантаження і про політику (тобто, умови) надання ресурсів (наприклад, вартість їхнього використання);
– протоколи управління, які забезпечують узгодженість доступу до поділюваного ресурсу і визначають необхідні операції, які ресурс повинен виконати (скажімо, ініціалізацію процесу або надання доступу

до даних), зокрема, вони забезпечують виконання функцій резервування й контролю відповідності ресурсів вимогам, які формулюються користувачем при запитуванні необхідних йому ресурсів.

2.3.4 Колективний рівень стеку Грід-протоколів

Колективний рівень (Collective Layer), на відміну від ресурсного рівня, який призначений для роботи з окремо взятими ресурсами, містить більш глобальні протоколи й сервіси, які не пов’язані з яким-небудь конкретним ресурсом і забезпечують колективну взаємодію ресурсів. Рівень відповідає за глобальну інтеграцію різних наборів ресурсів. У колективному рівні розрізняють протоколи двох типів: загальні й специфічні (призначені для застосувань). До загальних протоколів, у першу чергу, відносяться протоколи виявлення й виділення ресурсів, системи моніторингу й авторизації віртуальних організацій. Специфічні протоколи спеціально створюються для різних застосувань Грід, (наприклад, протокол архівації розподілених даних або протоколи управління завданнями збереження стану системи тощо).

Компоненти колективного рівня пропонують величезну розмаїтість методів спільного використання ресурсів. Прикладами сервісів рівня можуть слугувати нижче наведені функції й сервіси, які реалізовані в протоколах цього рівня:

– сервіси каталогів дозволяють віртуальним організаціям учасникам Грід виявляти наявні вільні ресурси і їхні властивості, виконувати запити за іменами і за такими атрибутами ресурсів, як тип і навантаження, цей сервіс використовує протокол Grid Resource Information Protocol (GRIP) ресурсного рівня.
– сервіси обробки заявок користувачів, спільного виділення, планування й розподілу ресурсів, що дозволяє учасникам Грід одержати доступ до необхідних ресурсів і забезпечує виділення одного або більшої кількості ресурсів, необхідних для досягнення визначеної мети, а також планування виконуваних на ресурсах завдань, прикладa-

ми реалізації таких сервісів слугують Application-Level Scheduling (AppLeS), Condor-G, Nimrod/G і DRM брокер;

– сервіси моніторингу й діагностики відслідковують аварії, атаки й перенавантаження ресурсів;
– сервіси дублювання (реплікації) даних дозволяють управляти зберіганням інформації в Грід, координують використання ресурсів пам'яті в межах віртуальних організацій, забезпечуючи підвищення швидкості доступу до даних шляхом оптимізації розташування й кількості копій даних відповідно до обраних метрик (критеріїв), таких як час відповіді, надійність, вартість тощо;
– сервіси управління робочим завантаженням і організацією спільних робіт – Problem Solving Environment (PSE) застосовуються для опису й управління багатокроковими, асинхронними, багатокомпонентними потоками завдань;
– служби авторизації співтовариств визначають правила використання ресурсів для користувачів Грід, сприяють поліпшенню правил доступу до поділюваних ресурсів, а також визначають можливості використання ресурсів співтовариства. Подібні служби дозволяють формувати політики доступу на основі інформації про ресурси, застосовувати протоколи управління ресурсами і протоколи безпеки єднального і ресурсного рівня;
– служби обліку і оплати забезпечують збір інформації про використання ресурсів для контролю звертань користувачів;
– сервіси координації і співробітництва підтримують і забезпечують синхронний й асинхронний скоординований обмін інформацією в потенційно великому співтоваристві користувачів (прикладами таких сервісів є CAVERNsoft і Access Grid);
– сервіси пошуку програмного забезпечення, які дозволяють розшукувати й вибрати необхідне для розв'язання проблеми програмне забезпечення (прикладами слугують NetSovle і Nimf).

Системи Грід-програмування на цьому рівні дозволяють використову-

вати вже відомі програмні моделі для Грід-оточення. Вони використовують різні Грід-сервіси, такі як використання ресурсів, призначення ресурсів і інші. Прикладами можуть слугувати Грід-реализації інтерфейсу передачі повідомлень Message Passing Interface (MPI).

2.3.5 Прикладний рівень стеку Грід-протоколів

Запуск застосувань в Грід-середовищі виконується на прикладному рівні (Application Layer). На рівні описуються користувальницькі застосування, які працюють в середовищі віртуальної організації; застосування функціонують, використовуючи протоколи, визначені на нижчерозташованих рівнях.

Цей найвищій рівень Грід-архітектури містить користувальницькі застосування, які виконуються в середовищі об’єднаних ресурсів. В процесі виконання застосування використовують протоколи нижчерозташованих рівнів, які забезпечують доступ до необхідних служб, а також прикладні програмні інтерфейси – Application Programming Interface (API), які відповідають даним протоколам.

Для полегшення роботи з прикладними програмними інтерфейсами користувачам надаються ще й набори інструментальних засобів для розробки програмного забезпечення – Software Development Kit (SDK). Ці набори інструментальних засобів високого рівня можуть забезпечувати функціональність з одночасним використанням декількох протоколів, а також комбінувати операції протоколів з додатковими викликами прикладних програмних інтерфейсів нижнього рівня.

У кінцевому рахунку можна стверджувати, що взаємодія прикладної програми із сервісами різних рівнів здійснюється через функції інтерфейсу прикладних програм цих сервісів. На рис. 6 умовно наведена схема взаємодії застосувань користувачів з рівнями архітектури Грід.

Застосування можуть викликатись за допомогою доволі складних оболонок і бібліотек. Ці оболонки і самостійно можуть визначати протоколи, сервіси і прикладні програмні інтерфейси, однак, подібні надбудови не від-

носяться до фундаментальних протоколів і сервісів, які визначають архітектуру Грід-систем.

Рисунок 6 − Взаємодія прикладної програми з різними рівнями Грід-системи

2.4 Опис Відкритої архітектури Грід-служб і Globus Toolkit

На сьогодні в світі накопичено великий досвід зі створення програмного середовища – ППЗ для реалізації розподіленої Грід-інфраструктури. Роботи в цьому напрямку почались ще у далекому 1999 році зі створення організації під назвою Глобальний форум з Грід-мереж – Global Grid Forum (GGF). Ця організація оформилася як основний розробник стандартів програмного забезпечення для Грід. В 2002 році GGF спільно з корпорацією IBM запропонувала стандарт програмного забезпечення для мереж Грід під назвою Відкрита архітектура Грід-служб – Open Grid Services Architecture (OGSA). OGSA додержується об'єктно-орієнтованої моделі і в якості основного об'єкта розглядає службу (сервіс), при цьому Грід-служба визначається як спеціальний вид веб-служби.

В той самий час, у 2002 році, GGF презентувала також пакет ППЗ

Globus Toolkit (GTK) версії 3.0, який став найпопулярнішім на той час програмним забезпеченням для Грід-систем, і довгий час вважався де-факто стандартом для Грід. На момент написання конспекту на сайті globus.org (організація дочірня GGF) пропонується вже версія 6.0 цього програмного продукту.

У подальшому більш докладно будуть розглянуті питання організації і управління розподіленням ресурсів на базі стандарту OGSA і інших систем управління Грід-ресурсами, а зараз є сенс зупинитись на ознайомленні зі складовими пакету Globus Toolkit і з взаємодією їх між собою.

2.4.1 Globus Toolkit як інструмент управління Грід-системами

Даний інструментарій управління Грід-системами реалізує механізми сервісів, які охоплюють питання захисту ресурсів системи, знаходження інформації, управління даними і ресурсами, питання комунікації, виявлення помилок тощо.

Для контролю за виконанням розподіленої послідовності обробки запиту користувача, в ППЗ передбачаються спеціальні служби, які відповідають за такі функції:

– відстеження переходів між шагами обробки запиту;

– збирання діагностики, яка видається на кожному шагу;

– реєстрацію кількості використаних ресурсів.

Дуже приблизно й схематично робота ППЗ і взаємодія його складових наведена на рис. 7.

Найважливішу роль в стабільній безпомилковій роботі системи, в ефективному розподілі ресурсів Грід, в їхній координації видіграє планувальник, так званий, брокер ресурсів. Постійно одержуючи інформацію про стан Грід-системи, планувальник для кожного конкретного завдання визначає найбільш придатні ресурси й резервує їх для використання лише цим завданням. Безумовно, під час виконання завдання може звернутись до планувальника і запросити у нього додаткові ресурси, з іншого боку завдання може вивільни-

ти надлишкові вже не потрібні йому ресурси. Брокер ресурсів відповідним чином реагує на ці ситуації. По завершенні завдання всі відведені для нього обчислювальні ресурси брокером звільняються, а ресурси пам’яті планувальник може використати для зберігання результатів роботи завдання.

Рисунок 7 − Схематичне представлення взаємодії елементів ППЗ

Важливою властивістю систем Грід є те, що користувачеві не потрібно знати про фізичне розташування ресурсів, відведених його завданню. Вся робота з управління, перерозподілу й оптимізації використання ресурсів покладається на планувальника і виконується ним абсолютно прозоро для користувача. Тобто для користувача цілком створюється ілюзія роботи в єдиному інформаційному просторі, який містить величезні обчислювальні потужності й обсяги пам’яті.

За своєю структурою Грід є найбільш складним інформаційним середовищем, яке коли-небудь створювала людина. Для системи такої складності дуже важливим є вирішення проблеми забезпечення надійного функціонування й відновлення роботи при збоях. Зрозуміло, що людина не здатна встежити за станом тисяч різних ресурсів, які входять у Грід-систему. Але такий

контроль виконувати необхідно, тому задача контролю над помилками покладається на систему моніторингу, яка стежить за станом окремих ресурсів. Дані про стан ресурсів системи постійно оновлюються і заносяться в інформаційні ресурси, звідки вони читаються планувальником і іншими сервісами, що надає цілком достовірну інформацію про стан Грід-системи і дозволяє відповідним чином реагувати на виникаючі збої в роботі.

Складність самої Грід-системи, наявність в ній великої кількості різноманітних ресурсів очікувано призводить до необхідності використовувати й складну систему виявлення і класифікації помилок. Всю безліч абсолютно різних за походженням помилок, виникаючих в системі, можна поділити на дві великі групи. Перша з них – це помилки, які відбуваються з вини завдання і друга – помилки, пов’язані зі збоями у будь-якому ресурсі. При виникненні помилки з першої групи планувальник зупиняє те завдання, з вини якого виникла помилка, а повідомлення про помилку з відповідною діагностикою спрямовує його власникові (користувачеві). В іншому разі, коли причиною збою послужив ресурс, планувальник поводиться інакше. Завдання лише призупиняється на деякий незначний час, робиться перерозподіл ресурсів для даного завдання і воно знову запускається на виконання. Така поведінка брокера можлива тому, що зазвичай Грід-системи володіють великою надмірністю, а брокер має повну інформацію про стан ресурсів системи на кожному відрізку часу.

Збої ресурсів є не єдиною причиною відмов у Грід-системах. Через величезну кількість завдань і постійно мінливу складну конфігурацію системи важливо вчасно визначати переобтяжені й вільні ресурси, роблячи перерозподіл навантаження між ними. Такий моніторинг обов’язковий тому, що переобтяжений мережний ресурс може стати причиною одночасної відмови значної кількості інших ресурсів. Планувальник, використовуючи систему моніторингу, постійно стежить за станом ресурсів і автоматично вживає необхідних заходів для запобігань з одного боку перевантаженню мережі, а з другого – марному простою ресурсів.

У розподіленому середовищі, яким є Грід-система, життєво важливою

властивістю є відсутність так званої єдиної точки збою. Це означає, що відмова будь-якого ресурсу не повинна призводити до збою в роботі всієї системи. Саме тому планувальник, система моніторингу й інші сервіси Грід-системи розподілені й продубльовані. Незважаючи на всю складність, архітектура Грід розроблялася з метою забезпечити максимальну якість сервісу для користувачів.

Щодо інструментальних засобів, які розроблені в рамках проекту Глобус (Globus Project), то вони утворюють набір програмних засобів Globus Toolkit (GTK) і дозволяють побудувати повноцінну Грід-систему. Засоби Globus Toolkit – це сукупність програмних компонентів, які й реалізують необхідні частини архітектури.

2.4.2 Складові програмного забезпечення Globus Toolkit

До складу Globus Toolkit входять п’ять основних компонентів.

Першим з таких компонентів є Globus Resource Allocation Manager (GRAM), або менеджер розподілу ресурсів Globus. Компонент призначений для створення/видалення процесів. Він встановлюється на кожному обчислювальному вузлі Грід-системи, роль такого вузла може виконувати як проста робоча станція (навіть персональний комп’ютер), так і потужний обчислювальний кластер або суперкомп’ютер. Запити від користувальницьких застосувань до GRAM повинні формулюватись спеціальною мовою Resource Specification Language (RSL) – мова специфікації ресурсів.

Другим важливим компонентом є Monitoring and Discovery Service (MDS) – служба моніторингу і пошуку. Цей сервіс забезпечує способи одержання інформації про Грід-систему. До складу інформації входять найрізноманітніші дані, які містять, наприклад, відомості про конфігурацію або стан всієї системи. З іншого боку це може бути інформація про її окремі ресурси, такі як тип ресурсу, доступний дисковий простір, кількість процесорів, обсяг пам’яті, продуктивність та багато чого ще. Вся зібрана інформація для спрощення доступу до неї організується у вигляді дерева. Така логічна організація

одержаної інформації надає змогу звертатись до неї за допомогою стандартного протоколу Lightweight Directory Access Protocol (LDAP) – полегшеного протоколу доступу до каталогів. Як відомо, цей відносно простий протокол використовує TCP/IP і дозволяє проводити операції автентифікації, пошуку та порівняння складових інформації, а також дозволяє виконувати операції додавання, зміни або видалення записів.

Ще один компонент Globus Toolkit – Globus Security Infrastructure (GSI) – інфраструктура безпеки Globus. Призначенням цієї сервісної служби є забезпечення захисту системи і даних з підтримкою їхнього шифрування. Крім того система також виконує автентифікацію, при якій однозначно встановлюється, що користувач або ресурс, який проходить автентифікацію дійсно є тим, за кого він себе видає. Додатково сервіс виконує й авторизацію, тобто процедуру перевірки, при якій встановлюється, що автентифікований користувач або ресурс дійсно має заявлені ним права доступу. Автентифікація і авторизація виконується з використанням стандартного механізму цифрових сертифікатів X.509.

Компонент Global Access to Secondary Storage (GASS) призначений для здійснення глобального доступу до вторинної системи зберігання і надає можливість зберігання великих масивів даних у розподіленому середовищі і здійснення доступу до цих даних. Важливою функцією зазначеного сервісу є визначення різноманітних стратегій розміщення даних і їхніх реплікацій.

Спеціальні бібліотеки globus_io і Nexus, які також входять до складу Globus Toolkit, призначені для спільного використання як прикладними програмами користувачів, так і компонентами самого пакету для мережної взаємодії вузлів у гетерогенному середовищі.

Говорячи про компоненти пакету Globus Toolkit, слід особливо відзначити, що пакет, як такий, не містить власного брокера ресурсів. Завдання його реалізації, з найбільш ранніх версій і практично до останнього часу, покладається на розробників, які створюють конкретну Грід-систему на його основі.

Всі компоненти Globus Toolkit об’єднуються спільною архітектурою

засобів управління ресурсами – Globus Resource Management Architecture (GRMA), яка має багаторівневу структуру і представлена на рис. 8.

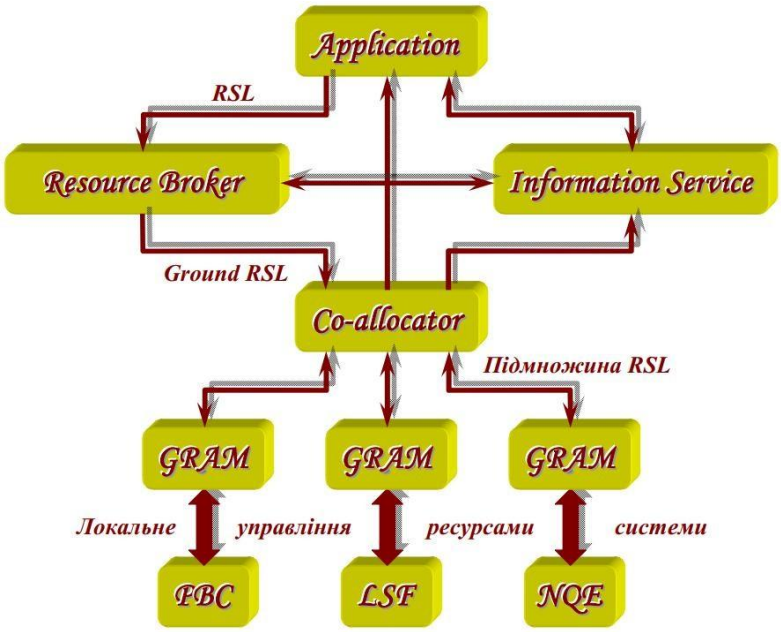

Рисунок 8 − Архітектура засобів управління ресурсами

Запити користувальницьких застосувань на виділення ресурсів, сформульовані мовою RSL, передаються брокеру ресурсів, який відповідає за високорівневу координацію щодо використання ресурсів (балансування навантаження) у тому чи іншому домені. Одержавши від користувальницького застосування запит, брокер ресурсів на його основі і, користуючись політиками відповідального адміністративного домену з прав доступу, обмеження з використання ресурсів тощо, приймає рішення щодо того, на яких обчислювальних вузлах буде виконуватися завдання, який відсоток обчислювальної потужності вузла воно може використати і т.ін.

Перш ніж направити завдання на обчислювальний вузол брокер ресурсів повинен визначити, які вузли доступні на момент одержання запиту, навантаження вузлів, їхню продуктивність і інші параметри, які були зазначені в RSL-запиті користувача. Потім зі всіх можливих варіантів використання ресурсів Грід вибрати найбільш оптимальний. В залежності від наявних ре-

сурсів, таким варіантом може виявитися потужний обчислювальний кластер з декількох вузлів або, навпаки, користувачу може на деякий час дістатись лише один обчислювальний вузол. На наступному кроці брокер ресурсів генерує новий RSL-запит на підмножині мови Ground RSL, знову сформований запит передається високорівневому менеджеру ресурсів (на рис. 8 це co-allocator). Цей запит вже містить більш конкретні дані про ресурси, які виділені для виконання завдання. До цих даних відносяться імена конкретних обчислювальних вузлів, необхідна кількість пам'яті для збереження даних тощо. Взагалі високорівневй менеджер ресурсів повинний виконувати, як мінімум, такий набір основних функції:

– колективне виділення ресурсів;
– додавання/видалення ресурсів до раніш виділеного;
– одержання інформації про стан завдань;
– передача початкових параметрів завданням.

Виконання цих функцій високорівневий менеджер ресурсів досягає шляхом декомпозиції запитів GroundRSL на безліч простіших RSL-запитів, які потім надсилаються до підсистем GRAM. Якщо GRAM не виявили помилок і не повернули менеджеру повідомлень про них, завдання користувача запускається на виконання. У противному випадку, якщо хоча б один з GRAM виявляє й повертає помилку, завдання або повністю знімається з виконання, або виконується ще одна спроба запуску завдання користувача повторно.

Як вже згадувалося, менеджер GRAM – один з основних компонентів Globus Toolkit і сам по собі є досить низькорівневим інтерфейсом між високорівневим менеджером ресурсів і локальною системою управління ресурсами вузла. Компонент призначається для надання верхнім рівням універсального API для управління всіма ресурсами вузла Грід.

Більш докладно структура цього компонента Globus Toolkit буде розглянута дещо пізніше після розгляду питань, пов'язаних з поняттям Грід-сервісів, а зараз лише має сенс розглянути його взаємодію з локальними системами управління ресурсами.

2.4.3 Взаємодія Globus Resource Allocation Manager з локальними системами управління ресурсами

З метою надання універсального API для управління всіма ресурсами вузла Грід, GRAM постійно взаємодіє з локальними засобами управління ресурсами вузла. Як вже згадувалося, вузлом може бути, наприклад, робоча станція або обчислювальний кластер. Не викликає сумніву та обставина, що такі абсолютно різні, повністю гетерогенні ресурсні вузли, кожен окремо, мають і повністю не схожі системи управління власними ресурсами. Але менеджер GRAM повинен вміти зв'язуватись з кожним з них, передавати запити й отримувати відповідні повідомлення від ресурсів. Тому GRAM через власний інтерфейс може взаємодіяти з великою кількістю локальних систем управління ресурсами.

Як приклад таких систем на рис. 8 представлена Portable Batch System (PBS) – система управління ресурсами й завантаженням кластерів. Система мультиплатформна і може працювати на великій кількості ресурсів під керуванням різних операційних систем відкритого типу, таких як: Linux, FreeBSD, NetBSD, Digital Unix, Tru64, HP-UX, AIX, IRIX, Solaris тощо. На сьогодні (на час написання конспекту) генеральній розробник системи – компанія Altair Engineering реалізує пакет Professional (PBS Pro). Система реалізується як комерційний продукт, але компанія пропонує і вільно розповсюджувану дещо обмежену реалізацію з відкритим кодом. Існує й багато конкурентних продуктів від різних розробників, серед яких можна окремо відзначити розповсюджувану під вільною ліцензією OpenPBS Software License реалізацію PBS під назвою Torque. Система розробляється і підтримується співтовариством Adaptive Computing Enterprises, Inc на базі проекту OpenPBS і є відгалуженням від нього. За деякими джерелами програмний продукт має більш широкі можливості, навіть у порівнянні з PBS Pro.

На тому ж самому рис. 8 надана ще одна система управління ресурсами – Load Sharing Facility (LSF). Ця комерційна система практично аналогічна PBS і розробляється компанією Platform Computing. Диспетчер ресурсів LSF

реалізує динамічне балансування навантаження в умовах багатокористувальницького гетерогенного середовища й вимагає, щоб обчислювальний вузол Грід, на якому він функціонує, був під його повним контролем. Балансування навантаження обчислювальних вузлів Грід-системи здійснюється диспетчером LSF на основі заздалегідь сформованої ним підсумкової черги завдань. Чергове завдання користувача з підсумкової черги LSF запускається на виконання, як тільки звільняються необхідні завданню ресурси. Крім можливості включення завдання в одну з наперед визначених черг, диспетчер LSF надає користувачу можливість переміщати власне завдання з однієї черги в іншу, припиняти й видаляти завдання із черги тощо. Програмне забезпечення LSF так само, як і PBS, здатне працювати на багатьох відкритих ОС, але може працювати під керуванням ОС Windows. За даними[1)], LSF є найбільш популярним диспетчером управління ресурсами.

Наступним можливим представником диспетчерів управління ресурсами, також представленим на рис. 8, є Network Queuing Environment (NQE). Цей продукт винятково розробляється і є власністю компанії Cray Research, тому найчастіше він використовується як менеджер ресурсів на суперкомп’ютерах, кластерах і системах Cray, хоча може працювати й на інших апаратних платформах під керуванням ОС відкритого типу.

На рис. 8 не позначений такий диспетчер управління ресурсами, як IBM LoadLeveler. Продукт компанії IBM здебільшого використовується на кластерах, як масивно-паралельних процесорів (MPP), так і кластерів робочих станцій (COW) на апаратній платформі IBM, і є системою керування завданнями, яка, за допомогою узгодження вимог обробки завдань із доступними ресурсами, дозволяє користувачам запустити на виконання більшу кількість завдань за менший час. Це система керування робочим навантаженням, яка дозволяє користувачам призначати, планувати й виконувати завдання в пулі ресурсів. Вона забезпечує динамічне планування й узгодження ро-

[1)] Карпенко А.П., Чернецов С.А. Балансировка загрузки распределенной гетерогенной вычислительной системы средствами GRID при распараллеливании одного класса задач. Наука и образование. № 11, 2008. ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". URL: http://engineering-science.ru/doc/111074.html (дата звернення 07.05.2019).

бочого навантаження для оптимального використання ресурсів кластера, надає єдиний центр управління завданнями й робочим навантаженням. LoadLeveler виконується у вигляді набору фонових процесів (демонів) на кожній клієнтській машині. Група клієнтських машин, яка зв'язується із центральною машиною управління, називається LoadLeveler-кластером. У цілому такий кластер забезпечує можливість створення, підтвердження, виконання й управління послідовними й паралельними пакетними завданнями.

Крім зазначених вище менеджерів управління завданнями, доволі великою популярністю користується вільно доступний менеджер ресурсів Condor (про нього мова буде йти далі). Менеджер розроблений в основному студентами різних університетів Європи й США. За своїми характеристиками він практично є аналогічним до перерахованих вище. Менеджер має ту перевагу, що здатний працювати не лише на різних відкритих Unix-сумісних платформах, але і на сімействі ОС Windows NT.

Додатково до менеджера ресурсів LoadLeveler компанія IBM спільно з високопродуктивним обчислювальним міжгалузевим дослідницьким центром Cornell Theory Center при Корнельському університеті, основним напрямком технічних досліджень і розробок якого є великомасштабні кластерні обчислення на платформі NT, розробила ще й систему Easy-LL. Програмне забезпечення цієї системи призначено для управління великим 256-процесорним кластером IBM – AC3 Velocity. Безпосередньо кластер складається з 64 серверів Dell PowerEdge, кожен з яких має чотири процесори Intel Pentium III Xeon 500 МГц і працює під управлінням Microsoft Windows NT. Цей спільний продукт по суті є об'єднанням власного LoadLeveler і продукту EASY національного дослідницького центра міністерства енергетики США – Argonne National Lab.

На завершення опису доволі складних локальних систем управління ресурсами, з якими може взаємодіяти GRAM, необхідно відзначити, що він здатний самостійно звертатись до системного виклику fork – найпростішому, визначеному стандартом POSIX, засобу запуску процесів в Unix-сумісних ОС.

2.4.4 Характеристика інформаційної системи Globus Toolkit

Інформаційна система Globus Toolkit, до якої для одержання всієї необхідної інформації про стан Грід-системи можуть звертатись, як всі перераховані компоненти системи, так і користувальницькі застосування, будується на базі інформаційного сервісу (Information Service). В Globus Toolkit роль такого інформаційного сервісу грає Система Виявлення Несправностей – Malfunction Detection System (MDS). Цей компонент відповідає за збір і надання конфігураційної інформації, інформації про стан Грід-системи і її підсистем, а також забезпечує універсальний інтерфейс одержання необхідної інформації. MDS має децентралізовану, легко масштабовану структуру й працює як зі статичними, так і з динамічно мінливими даними, необхідними користувальницьким застосуванням і різним сервісам Грід-системи. Ієрархічна структура MDS представлена на рис. 9.

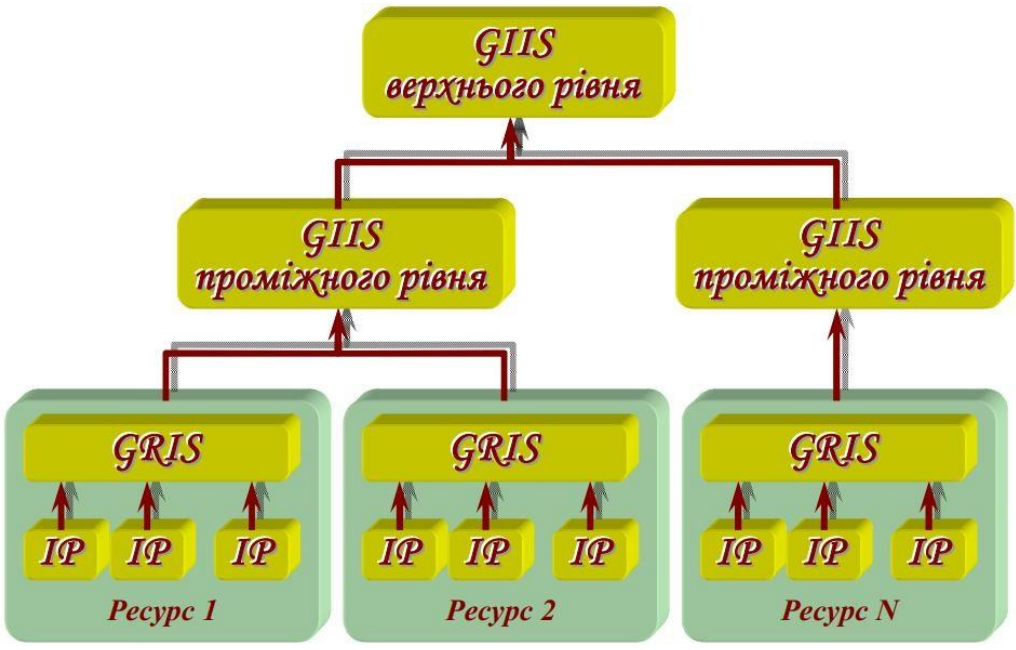

Рисунок 9 − Ієрархічна структура MDS

Три основних компоненти входять до складу MDS.

Першим з них є Information Provider (IP). Він слугує основним джере-

лом інформації про той чи інший конкретний ресурс або, навіть, про деяку окрему частину ресурсу. Велика кількість IP складають перший – найнижчий шар MDS.

Наступний шар, також безпосередньо пов'язаний з ресурсами системи, утворюють Grid Resource Information Services (GRIS). Ці сервіси опитують вся зв'язані з ними індивідуальні IP, отримують від них первинну інформацію про стан ресурсу або його частин. Виконують обробку і узагальнення отриманої інформації з метою надання інформації про вузол Грід-системи в цілому. Вузлом системи може бути як обчислювальний вузол, так і який-небудь інший ресурс.

Зібрана, оброблена і узагальнена інформація від GRIS надходить на наступний шар MDS, який має два рівні і обслуговується Grid Index Information Services (GIIS). GIIS проміжного рівня збирають і узагальнюють інформацію від різних GRIS і передають її GIIS верхнього рівня, який вже містить всю можливу інформацію про стан даної системи Грід. Для зменшення часу реакції на запит і зниження мережного трафіка GIIS кешують дані.

Грід-системи здебільшого функціонують в незахищених мережах загального доступу (Інтернет). Таке становище зумовлює необхідність серйозно підходити до вирішення проблеми безпеки даних і взагалі ресурсів Грід. Безпечна робота систем на основі Globus Toolkit забезпечується відповідною інфраструктурою безпеки – Grid Security Infrastructure (GSI). Цією підсистемою надається низка сервісів, таких як автентифікація, конфіденційність передачі інформації і єдиний вхід у Грід-систему. Під поняттям «єдиний вхід» мається на увазі, що користувачу потрібно лише один раз пройти процедуру автентифікації, а далі система сама подбає про те, щоб автентифікувати його на всіх ресурсах, якими він збирається скористатися. GSI ґрунтується на надійній і широко використовуваній інфраструктурі криптографії Public Key Infrastructure (PKI) з відкритим ключем.

2.4.5 Коротко про цифрові сертифікати X.509

Як ідентифікатори користувачів і ресурсів в GSI використовуються цифрові сертифікати X.509. У роботі із сертифікатами X.509 і в процедурі видачі/одержання сертифікатів задіяні три сторони.

По-перше – це «Центр Сертифікації» Certificate Authority (CA) – спеціальна організація, яка має повноваження видавати (підписувати) цифрові сертифікати. Зазвичай різні CA незалежні між собою. Відносини між CA і його клієнтами регулюються спеціальним документом.

Другою стороною процедури є «Передплатник» – це людина або ресурс, який користується сертифікаційними послугами CA. CA включає в сертифікат дані, які надаються передплатником (ім’я, організація тощо) і затверджує його власним цифровим підписом.

Насамкінець третьою стороною виступає «Користувач» – це людина або знов-таки ресурс, який покладається на інформацію із сертифіката при одержанні його від передплатника. Користувачі можуть приймати або відхиляти сертифікати за підписом якої-небудь CA.

Взагалі в системі Globus Toolkit, в залежності від того на якого користувача оформлювався сертифікат, використовуються два типи сертифікатів X.509:

– сертифікат користувача (User Certificate) – цей сертифікат повинен мати кожен користувач-людина, яка працює з Грід-системою. Сертифікат користувача містить інформацію про ім’я користувача, організацію, до якої він належить, і про центр сертифікації, який видав даний сертифікат.
– сертифікат вузла (Host Certificate) – цей сертифікат повинен мати кожен вузол (ресурс) Грід-системи. Сертифікат вузла аналогічний сертифікату користувача, але в ньому замість імені користувача вказується доменне ім’я конкретного обчислювального вузла.

На завершення слід сказати, що сьогодні описаний вище інструментарій застосовується в багатьох проектах по всьому світу. Але він далеко не

єдиний програмний продукт, призначений для реалізації Грід-систем. Крім Globus Toolkit з метою управління Грід-системами розроблені і існують ще багато інших програмних середовищ ППЗ, наприклад таких, як Legion, Condor, Unicore, ARC, gLite тощо.

2.5 Контрольні запитання і завдання для самостійної роботи

1. Поясніть, чому Грід не є частиною Інтернету, а навпаки, між ними існує протилежний зв’язок.
2. Докладно розкажіть про складову Грід-систем, яка виконує роль розвинутої технології для безпосередньої координації всіх доступних ресурсів?
3. Опишіть архітектуру Грід-систем з точки зору програмного забезпечення – модель «піскового годинника», розкажіть яким стандартам відповідає ця модель.
4. Для чого призначений, за що відповідає і якими ресурсами управляє апаратний рівень (Fabric Layer) стеку Грід-протоколів? Які механізми потрібні для доступу до обчислювальних ресурсів для запуску програм й моніторингу їхнього виконання?
5. Які механізми потрібні для роботи з ресурсами каталогів, яка сутність цих ресурсів? Чи можлива коректна одночасна робота з принципово різними ресурсами у Грід?
6. Для чого призначені комунікаційні протоколи єднального рівня, як вони реалізуються, на яких стандартах безпеки ґрунтуються?
7. Якими способами здійснюється ідентифікація користувачем? Докладно розкажіть про існуючі способи.
8. Охарактеризуйте ресурсний рівень стеку Грід-протоколів, його призначення, існуючі протоколи тощо.
9. Чим відрізняється колективний рівень стеку Грід-протоколів від ресурсного, які компоненти колективного рівня пропонують методи спільного використання ресурсів?

10. Які спеціальні служби ППЗ передбачаються для контролю за виконанням розподіленої послідовності обробки запиту користувача і за які функції вони відповідають?
11. Охарактеризуйте роль брокера ресурсів в стабільній безпомилковій роботі системи, в ефективному розподілі ресурсів Грід, в їхній координації.
12. Назвіть причини і розкажіть про помилки, які виникають в Грід-системі і як на ці помилки реагує брокер ресурсів.
13. Що таке єдина точка збою і чому її відсутність є життєво важливою властивістю для розподіленого середовища, яким є Грід-система.
14. Перерахуйте всі п’ять основних компонентів, які входять до складу Globus Toolkit і надайте їм розгорнуту характеристику (склад, призначення тощо).
15. Докладно розкажіть про спільну архітектуру засобів управління ресурсами, якою об’єднуються всі компоненти Globus Toolkit.
16. Як і з якими ресурсами взаємодіє менеджер розподілу ресурсів Globus? Назвіть їх і надайте відповідні характеристики.
17. Розкажіть про систему виявлення несправностей Globus Toolkit, за що відповідає цей компонент. Докладно опишіть три основних компоненти, які входять до його складу.
18. Для чого використовуються цифрові сертифікати X.509? Охарактеризуйте кожну з трьох сторін, які задіяні в процедурі видачі/одержання сертифікатів.

3 ОРГАНІЗАЦІЯ І УПРАВЛІННЯ РОЗПОДІЛЕННЯМ РЕСУРСІВ (WSRF, GRAM, CONDOR).

На початковому етапі свого розвитку додатки взаємодіяли за допомогою приватних, спеціально розроблених (пропрієтарних) протоколів, а системні адміністратори використовували оригінальні методи, щоб ними управляти. За роки становлення і розвитку технології розподілених обчислень були

вироблені, з різним ступенем успіху, численні стандарти, щоб зменшити витрати на розгортання й обслуговування таких систем. Сьогодні найкращими підходами до побудови розподілених систем вважаються сервісно-орієнтована архітектура – Service Oriented Architecture (SOA), технологія веб-сервісів і Грід-стандарти, з яких у першу чергу слід зазначити Відкриту архітектуру Грід-сервісів – Open Grid Services Architecture (OGSA).

3.1 Опис сервісно-орієнтованої архітектури

Як відомо, архітектура – це формальний опис системи, який визначає її цілі, функції, ззовні видимі властивості і інтерфейси. У це поняття також включається опис внутрішніх компонентів системи і їхніх відносин, поряд із принципами, які управляють її дизайном, функціонуванням і можливою наступною еволюцією.

За визначенням, яке надано організацією Organization for the Advancement of Structured Information Standards (OASIS), SOA – це парадигма способу організації й використання розподілених можливостей, які можуть належати різним власникам. В основі сервіс-орієнтованої архітектури лежить сутність «дії» (на противагу сутності «об’єкта» в об’єктно-орієнтованій архітектурі). Типовими складовими сервіс-орієнтованої архітектури є:

- сервісні компоненти (сервіси);
- контракти сервісів (інтерфейси);
- з’єднувачі сервісів (транспорт);
- механізми виявлення сервісів (регістри).

Таким чином, SOA є основою для побудови надійних розподілених систем, які як послуги надають функціональні можливості, з додатковим акцентом на слабкі зв’язки між взаємодіючими сервісами. Цей стиль припускає реалізацію компонентів системи як модульних сервісів, які можуть бути знайдені й використані клієнтами.

У цьому контексті під сервісом (службою) розуміється програмний компонент, до якого можна віддалено звернутися за допомогою мережі і який

надає деякі функціональні можливості запитуючій стороні.

У загальному випадку сервіси корисні самі по собі, але їхнє об'єднання дозволяє надати єдиний вже доволі високорівневий сервіс, крім того це дозволяє багаторазово використовувати вже існуючі функціональні можливості. Іноді сервіс є повністю незалежним, іншим часом може спостерігатися його залежність від наявності інших сервісів або яких-небудь інших ресурсів, наприклад, баз даних. Сервіси надають клієнтам інформацію про свої можливості, інтерфейси, політику і протоколи зв'язку, які ними підтримуються.

Але конкретні деталі реалізації сервісу клієнтам не потрібні й ніколи не надаються. Рис. 10 ілюструє простий цикл взаємодії сервісів, так званий «трикутник SOA». В цьому трикутнику шуканий сервіс заздалегідь сповіщає сервіс реєстрації про своє існування й властивості. Способи взаємодії клієнта із сервісом реєстрації зазвичай відомі заздалегідь.

Рисунок 10 − Взаємодія сервісів в SOA-середовищі

Тут клієнт (людина або інший сервіс) робить запит на сервіс реєстрації, щоб знайти потрібний йому сервіс. Реєстраційний сервіс повертає список придатних сервісів, клієнт вибирає один з них і, використовуючи будь-який взаємно розпізнаваний протокол, вже звертається із запитом до нього. Обраний сервіс у відповідь повертає результат запитаної операції або повідомлення про помилку. Це найпростіший випадок взаємодії клієнта із сервісом – синхронний, двонаправлений спосіб обміну повідомленнями. У реальності процес буває значно складнішим – взаємодія можлива тільки однобічна, від-

повідь може вертатися не від запитаного сервісу, а від деякого іншого, якому запит був переданий для завершення обробки. Крім того сервіс може обслуговувати тільки зареєстрованих користувачів, пропонувати різні рівні сервіси різним користувачам, вимагати оплати за використання тощо.

3.2 Поняття слабкого зв'язку сервісів

SOA передбачає наявність слабкого зв'язку сервісів. Це означає, що взаємодіючі програмні компоненти мають лише мінімальне знання один про одного і знаходять необхідну для взаємодії інформацію безпосередньо перед самим процесом зв'язку. Заснована на такому принципі архітектура володіє цілою низкою переваг:

- гнучкістю – сервіс розташовується на будь-якому сервері, і може бути переміщений, але клієнти можуть знаходити й використовувати його поки на сервіс є посилання в службі реєстрації;
- масштабованістю – функціональні можливості сервісу можуть розширюватися або звужуватися шляхом динамічного опису сервісу, що приводить до зміни запитів;
- є можливість модифікації реалізації – при збереженні оригінальних інтерфейсів, реалізація сервісу може оновлюватись без збоїв в обслуговуванні клієнтів;
- стійкістю до відмов – при виникненні проблеми в роботі сервера, самого сервісу або сегмента його розташування у мережі тощо, клієнти, звернувшись до служби реєстрації, завжди знайдуть інший сервіс, який надає потрібні послуги.

Особливе значення ці переваги отримують у такому динамічно розподіленому середовищі, як Грід.

Тобто переваги слабкого зв'язку сервісів полягають у тім, що клієнт може використовувати будь-який сервіс, який здатен виконати його запит. Але при обслуговуванні сервісом складних запитів, які вимагають декількох кроків для їхнього виконання, на перший план висувається таке поняття, як

«стан сервісу». І дуже важливим стає існування таких сервісів, які називаються «сервіси без станів».

3.3 Сервіси без станів і їхнє функціонування

Функціонування сервісу, як сервісу без станів (stateless), передбачає, що при багатокроковій обробці запитів наприкінці кожного проміжного кроку сервіс повертає клієнтові достатню інформацію про свій поточний стан. Це дає можливість клієнтові на наступному кроці, попередньо передавши інформацію про стан, звернутися до будь-якого іншого сервісу з відповідними властивостями, який і продовжить обслуговування незалежно від того, чи сам він обробляв запит на попередньому кроці, чи це робив інший сервіс. Очевидно, що в цьому випадку обраний сервіс повинен мати можливість приймати й обробляти інформацію про стан, яка поставляється клієнтом і бути здатним до обробки будь-якої частини або всього запиту в цілому.

Рис. 11 показує клієнта, який робить складний запит, а для його обробки потрібно три кроки і кілька сервісів.

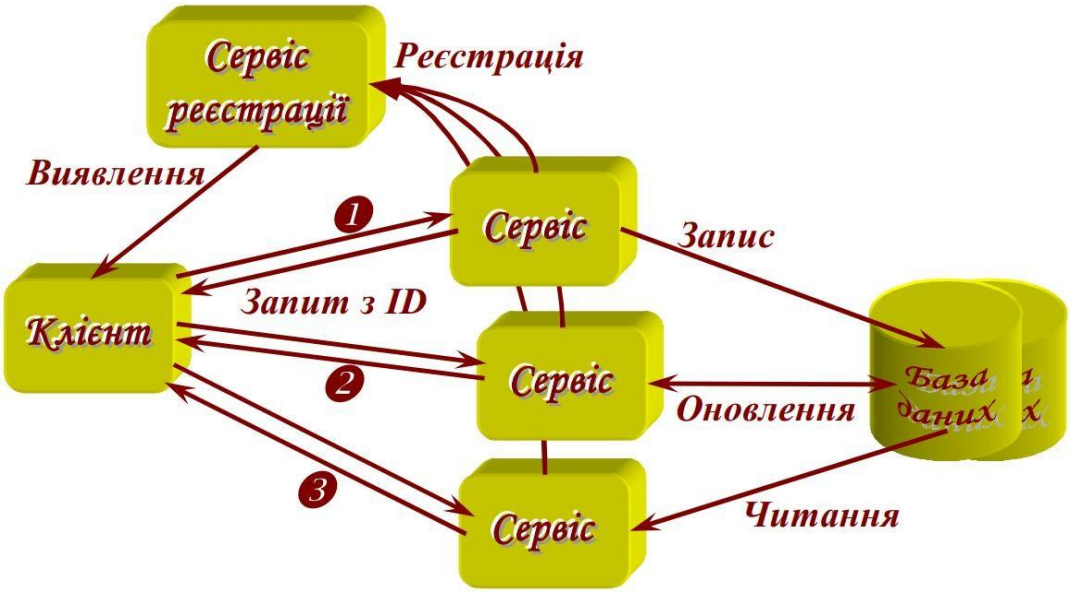

Рисунок 11 − Багатокрокова взаємодія клієнта й сервісів

Для максимального зменшення обсягу інформації про стани, якою обмінюються клієнт і сервіси, на кожному кроці обробки запиту істотні деталі інформації зберігаються в базі даних. При цьому всі обслуговуючі сервіси повинні мати можливість звернутися до БД і одержати необхідну інформацію, наприклад, на основі ідентифікаційного номера клієнта/запиту.

Веб-сервіси – є технологією, розробленою для підтримки взаємодії між розподіленими системами за допомогою обчислювальної мережі, і на сьогоднішній день – така технологія найчастіше зустрічається при реалізації сервісно-орієнтованої архітектури. У основі технології лежить процес обміну повідомленнями між сервісами, які ідентифікуються своїми мережними адресами, і інтерфейс яких описується у форматі документів мовою eXtensible Markup Language (XML), точніше на її спеціальному «діалекті» – Web Service Description Language (WSDL). Інші програмні системи можуть взаємодіяти з веб-сервісами за допомогою повідомлень, які базуються на іншому «діалекті» XML Simple Object Access Protocol (SOAP). Передача повідомлень відбувається з використанням протоколів HTTP, XML, XSD, WSDL, SOAP, UDDI.

3.4 Основні стандарти, протоколи і специфікації підтримки сервісів

Для підтримки представлення, пошуку й обміну інформацією між веб-сервісами визначені три основних стандарти – це WSDL, UDDI і SOAP (див. рис. 10).

Крім основних стандартів, архітектуру веб-сервісів складає маса протоколів і специфікацій. Їх можна поділити на чотири частини (процес, опис, повідомлення, зв’язок), які утворюють стік протоколів, у якому кожен горішній рівень опирається на нижній рівень (рис. 12).

Веб-сервіси надають детальну інформацію про свої функції й інтерфейси, але не про деталі реалізації і про власну платформу. Таким чином, клієнт і сервіс, підтримуючи спільні протоколи комунікації, взаємодіють незалежно від платформ, на яких вони виконуються, і мов програмування, на яких вони

написані. Це робить веб-сервіси особливо пристосованими для розподіленого гетерогенного середовища.

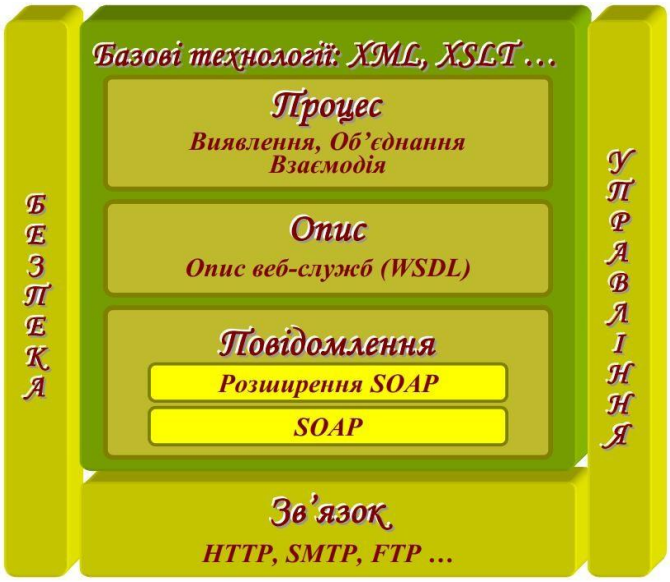

Рисунок 12 − Стік протоколів веб-сервісів

SOA і веб-сервіси можуть використовуватись спільно, як це часто й трапляється, але вони взаємно незалежні. Сервісна орієнтація – це архітектурний стиль, а веб-сервіси – технологія виконання.

Зазвичай вважається, що SOA призначена лише для розподілених систем. Але вона може використовуватися і для одиночних комп'ютерів, коли сервіси взаємодіють по внутрішніх каналах зв'язку. Можливе використання SOA і в кластерах персональних комп'ютерів, де сервіси взаємодіють за допомогою високошвидкісної локальної мережі. У свою чергу веб-сервіси добре підходять для побудови SOA-середовища, але при цьому не існує обов'язкових вимог втілення принципів SOA.

3.5 Формування концепції Грід-сервісів

Поширення мови XML і веб-сервісів дало старт процесу зближення технологій Грід і веб-сервісів, таке зближення зумовило формування концеп-

ції Грід-сервісів. Як уже згадувалося вище (див. стор. 49), у рамках Global Grid Forum почалися роботи зі стандартизації в новій області. Запропонована форумом архітектура поєднує основні технології Грід з веб-сервісами, що дозволяє створювати каркаси інфраструктури для інтеграції, віртуалізації й управління різнорідними ресурсами у віртуальній організації.

Як відомо, робота Грід-систем опирається на програмне забезпечення проміжного рівня, тобто програмні компоненти і протоколи, які забезпечують необхідний контрольований доступ до ресурсів. На першому етапі свого існування Грід-системи створювались або на основі спеціально розроблених загальнодоступних компонентів, або на основі закритих (пропрієтарних) технологій. Хоча різні вільні і комерційні рішення були успішні у своїх галузях застосування Грід, кожне зі своїми сильними й слабкими сторонами, вони мали обмежений потенціал як підґрунтя для Грід нового покоління, який повинен бути масштабованим і інтероперабельним, щоб задовольняти потреби широкомасштабних наукових і виробничих проектів.

Поступово прийшло розуміння того, що є значне перекриття між цілями обчислювального Грід і перевагами середовищ, основаних на принципах SOA і веб-сервісах. Швидкий прогрес у технології веб-сервісів і в розробці відповідних стандартів забезпечили еволюційний шлях від жорсткої й вузько-спрямованої архітектури Грід-систем першого покоління до стандартизованих, сервіс-орієнтованих Грід, які гарантують стабільно високу якість обслуговування користувачів.

Доволі схематично, структура найпростішого сервісно-орієнтованого Грід представлена на рис. 13. Тут сервіси використовуються й для віртуалізації ресурсів, і для забезпечення інших функціональних можливостей Грід.

Алгоритм роботи такого сервісно-орієнтованого Грід доволі простий:

а) програмне забезпечення консолі отримує від сервісу реєстрації інформацію про існуючі Грід-ресурси;

б) користувач через консоль входить у контакт із сервісами, які віртуалізують кожен ресурс для періодичного одержання даних про роботу ресурсів і повідомлень про зміни в їхньому стані;

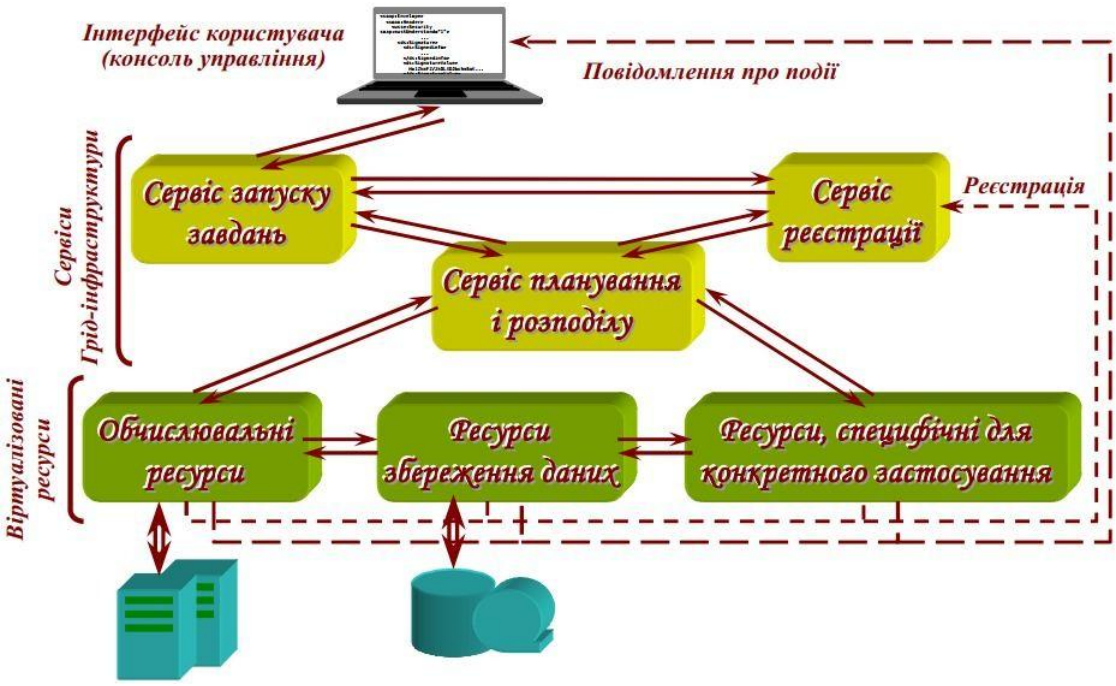

Рисунок 13 − Спрощена структура сервісно-орієнтованого Грід

в) запит на запуск завдання спрямовується в службу запуску, яка у свою чергу передає запит службі розподілу завдань;

г) служба розподілу звертається до служби, яка представляє застосування, з інформацією про вимоги до ресурсів, які необхідні для виконання завдання;

д) служба розподілу запитує в служби реєстрації інформацію про всі підходящі ресурси в Грід і безпосередньо контактує з ними на предмет доступності;

е) якщо потрібні ресурси доступні, планувальник вибирає найкращу доступну сукупність ресурсів і передає інформацію про неї сервісу застосування із запитом на початок виконання;

ж) у противному випадку планувальник ставить завдання в чергу й виконує його, коли необхідні ресурси стають доступними;

з) коли виконання завдання закінчується, сервіс застосування повідомляє про результат планувальнику, який сповіщає про це сервіс запуску завдань;

и) сервіс запуску завдань, у свою чергу, повідомляє користувача.

Цілком очевидно, що така схема роботи Грід дуже спрощена, а функціонування реального Грід промислового рівня є набагато складнішим, чим це показано на схемі.

Поява й бурхливий розвиток технології веб-сервісів поступово зумовили розуміння того факту, що відсутність стандартизації в найбільш важливих сферах побудови розподілених обчислювальних систем призводить до несумісності розроблювальних вирішень. Наприклад, відсутність єдиних протоколів авторизації й автентифікації користувачів спочатку приводила до того, що кожен розроблювач самостійно намагався щось придумати в цій сфері. Тому, коли виникала необхідність взаємодії користувача із зовнішньою сервісно-орієнтованою системою (навіть такою простою, як на рис. 13) доводилося витрачати багато зусиль для забезпечення стабільної роботи спільного вирішення.

3.6 Стандарт Open Grid Services Architecture

Особливо гостро ця проблема виявилася при спробі тісного зближення SOA і обчислювальних Грід-систем. Спроба усунути виниклі труднощі була зроблена згадуваною вище (див. стор. 49) організацією GGF, і втілилася в розробці стандарту OGSA. Створений і рекомендований до впровадження стандарт фіксує архітектуру OGSA і описує загальні принципи побудови сервісно-орієнтованого Грід. Опис виконується в термінах необхідних функціональних можливостей Грід, наприклад, виконання завдань, управління ресурсами й даними, забезпечення безпеки. Основною метою введення стандарту й інших документів, пов’язаних з OGSA, є стандартизація інтерфейсів і поведінка основного (базового) набору Грід-сервісів. Взаємодія Грід-сервісів як один з одним, так і з Грід-застосуваннями не повинна залежати від їхніх конкретних реалізацій. Так, зокрема, OGSA:

- визначає стандартні механізми для створення, іменування й виявлення постійних і тимчасових екземплярів сервісів;
- забезпечує прозорість місцезнаходження і взаємодії по різних прото-

колах для екземплярів сервісів;

– описує загальну модель ресурсів, яка є абстрактним представленням як фізичних ресурсів (таких як процесори, процеси, диски й файлові системи), так і логічних (екземплярів сервісів).

У рамках OGSA принципово різняться поняття сервісу й екземпляра сервісу. Сервіс – це абстрактне поняття, обумовлене своїм інтерфейсом і способом взаємодії з іншими сервісами. Тобто передбачається, що зовсім необов'язкові ні конкретна реалізація сервісу, ні його реальне функціонування. Екземпляром же того або іншого сервісу є реально функціонуючий сервіс. Тобто відмінність між сервісом і екземпляром сервісу майже така, як між класом і об'єктом у ООП.

При роботі над OGSA розробники намагалися врахувати низку вимог, отриманих у результаті аналізу різних сценаріїв використання Грід-систем. У число таких вимог входять:

– інтероперабельність (віртуалізація ресурсів, загальні способи керування й виявлення ресурсів, стандартизація протоколів);
– поділюваний доступ до ресурсів;
– оптимізація виділення ресурсів;
– можливість запуску завдань на віддалених обчислювачах;
– можливість маніпуляції даними;
– зниження вартості адміністрування великих гетерогенних систем;
– безпека (автентифікація й авторизація, можливість інтеграції з різними системами безпеки, делегування прав)
– надійність;
– гарантії якості обслуговування;
– простота використання і розширюваність (можливість розширення спектра застосування);
– масштабованість (відсутність вузьких місць, які перешкоджають нарощуванню обсягу ресурсів).

Як видно перед розроблювачами стандарту стояло досить масштабне й складне завдання. В обмеженому обсязі конспекту неможливо дати повний

опис всіх положень стандарту Відкритої Архітектури Грід-сервісів, але є сенс хоча б коротко згадати деякі з них.

Так OGSA визначає Грід-сервіс як веб-сервіс, який надає набір коректних інтерфейсів, визначених мовою WSDL, і дотримує специфічні конвенції для їхнього створення й композиції складних розподілених систем. Інтерфейси сервісів визначають способи виявлення, динамічного створення служб, управління життєвим циклом, повідомлення, керованості. Угоди по інтерфейсах визначають спосіб іменування і можливість модернізації Грід-служб. Крім того OGSA в обов'язковому порядку визначає механізми відновлення «знань» клієнта про службу. До числа таких знань, наприклад, відносяться відомості про операції, які підтримуються сервісом, про мережні протоколи, які клієнт може використовувати для зв'язку з цією службою й багато чого іншого.

Але практично відразу при розробці OGSA стало ясно, що механічне перенесення архітектури SOA на Грід-служби не веде до очікуваних результатів. Однією з причин невідповідності можливостей SOA вимогам нового стандарту стала, наприклад, та обставина, що традиційні веб-сервіси дозволяли виявляти й ініціювати тільки постійні служби. А стандарт OGSA, зі свого боку, передбачав обов'язкову підтримку тимчасових екземплярів сервісів, які б ініціювалися і завершувалися динамічно. Як приклад таких сервісів можна привести тимчасовий екземпляр служби, що створюється для звертання до сховища даних з метою резервування мережних або процесорних ресурсів.

У зв'язку з тим, що існуючі стандарти веб-сервісів (WSDL, SOAP, UDDI) не могли забезпечити всіх вимог, які ставилися розробниками до функціональних можливостей Грід-сервісів, для створення базової моделі Грід розроблювачами OGSA була організована робоча група Інфраструктури Відкритих Грід-сервісів – Open Grid Service Infrastructure (OGSI).

Основними завданнями, які ставилися перед цією робочою групою, були модифікація й розширення відповідних стандартів. Результатом роботи групи став випуск специфікації OGSI, у якій визначалися механізми створення, управління й обміну даними між Грід-сервісами. Іншими словами специ-

фікація OGSI є формальним описом Грід-сервісів. Основні інтерфейси моделі OGSA – OGSI (у термінології мови WSDL інтерфейси називають PortType) і їхній опис наведені в табл. 1.

Таблиця 1 – Інтерфейси Грід-служб по OGSA – OGSI

PortType (інтерфейс)	Операція	Опис
GridService	FindServiceData	Фактично, реалізація створення й забезпечення доступу до ресурсів, які приховані за службою.
	SetServiceData	
	SetTerminationTime	
	Destroy	
Factory	CreateService	
HandleMap	FindByHandle	
NotificationSource	SubscribeToNotificationTopic	Система сповіщення.
NotificationSink	DeliverNotification	
Registry	RegisterService	Реєстрація дескрипторів Грід-служби
	UnregisterService	

OGSA спільно зі специфікацією OGSI описує те, як створюється Грід-служба, як вона називається, як визначається її час життя тощо. OGSA пропонує базову поведінку, але не визначає конфігурацію служби і її розгортання в Грід-інфраструктурі. Інакше кажучи, OGSA не торкається питань реалізації, парадигми програмування, конкретних мов, механізмів і інструментальних засобів реалізації або операційного середовища. Це дозволяє моделі Грід-служб абстрагуватися від особливостей програмного й апаратного забезпечення, пропонуючи лише набір стандартних інтерфейсів високого рівня. Зокрема, згідно з OGSA, будь-який Грід-сервіс повинен реалізовувати тип GridService, який служить основним визначенням інтерфейсів в OGSA. Він у деякому значенні є аналогом базового класу Object в об’єктно-орієнтованих мовах. Така аналогія проявляється в тім, що GridService інкапсулює базову поведінку компонентної моделі. Базовими засобами комунікації GridService є

загальноприйняті механізми передачі повідомлень і механізм віддалених викликів процедур RPC. Орієнтація механізмів передачі повідомлень на передачу документів призводить до того, що вхідними й вихідними об'єктами є XML-документи. У свою чергу механізм RPC використовує більш строго визначені API і при цьому забезпечує більш високу продуктивність.

У рамках OGSA екземпляри служб (не плутати з сервісами) створюються або за допомогою операції CreateService, або вручну. Знищення екземплярів сервісів виконується динамічно після закінчення часу, на який вони створювалися. У будь-який довільний момент часу знищення динамічно створеного екземпляра можливо виконати за допомогою операції Destroy.

Операція CreateService (відповідно до стандарту POSIX) повертає дескриптор Грід-служби – Grid Service Handle (GSH). Цей дескриптор глобально є унікальним покажчиком (URL), який визначає ім'я екземпляра служби й відрізняє його від інших екземплярів. GSH не несе в собі всієї інформації, достатньої для того, щоб клієнт міг прямо взаємодіяти з екземпляром служби. Тому GSH необхідно відобразити на Grid Service Reference (GSR) – посилання на Грід-службу, яка містить інформацію, необхідну клієнту для зв'язку зі службою. Така двоступінчаста структура дозволяє GSH залишатися незмінним протягом усього терміну існування екземпляра Грід-сервісу, що дозволяє реалізувати механізм сервісів без станів. З іншого боку посилання на Грід-службу – GSR у загальному випадку може змінитися, наприклад, в разі «переносу» служби на інший комп'ютер. Цілком очевидно, що при виникненні подібної ситуації, клієнт повинен виконати відображення GSH служби на нове посилання.

Модель повідомлень, як частина OGSA, призначена для опису механізмів доставки повідомлень від джерела до адресату, наборів даних, структури служби і змін в її стані. Тут під джерелом повідомлень розуміється екземпляр Грід-служби, який реалізує інтерфейс NotificationSource і посилає повідомлення з інформуванням. У свою чергу, адресат – це екземпляр Грід-служби, який реалізує інтерфейс NotificationSink і одержує відправлене джерелом повідомлення з інформуванням. Для ініціювання повідомлення від конкретної

Грід-служби, користувач викликає операцію SubscribeToNotificationTopic у відповідності до інтерфейсу джерела повідомлення, передаючи йому дескриптор GSH служби-одержувача повідомлення.

У результаті спільних зусиль розробників стандарту OGSA, робочої групи OGSI з її відповідною специфікацією і розробників веб-сервісів вийшла багаторівнева структура (рис. 14) для побудови Грід-систем.

Рисунок 14 − Схематичне представлення рівнів інфраструктури,
яка призначена для побудови Грід

Взаємозв’язок між цими трьома рівнями представлення інфраструктури, що працює на основі сервісних принципів, повністю відповідає тому, що відбувається при створенні фізичних конструкцій. Спочатку засобами OGSA фіксується архітектурний задум, потім з використанням OGSI виконується інженерний проект, а потім з компонентів Globus Toolkit будується «спорудження» з Грід-сервісами.

Таким чином, з огляду на внесені специфікацією OGSI доповнення до OGSA, можна вже дати більш точне визначення поняттю Грід-сервіс.

Грід-сервіс – це (можливо тимчасова) служба на базі Грід-протоколів, описана за допомогою WSDL. Кожен інтерфейс Грід-служби визначає набір операцій, які ініціюються шляхом передачі відповідної послідовності повідомлень.

Однак при спробах створення Грід-систем за моделлю OGSA – OGSI, практично відразу стало зрозуміло, що спільне використання веб-сервісів і Грід-сервісів в одному середовищі є неможливим, через несумісність базових стандартів. Використання OGSI у незрозумілій якості, чи то проміжного шару, який служить для узгодження вимог OGSA з існуючими веб-службами, чи то латки, яка дозволяє веб-службам виконувати не властиві їм дії, практично не дало очікуваного результату. Та й сама специфікація OGSI не була добре сприйнята ні практиками-розробниками Грід-систем, ні веб-співтовариством, оскільки, на думку фахівців, мала низку істотних недоліків, основними з яких були:

– надмірна складність і громіздкість специфікації;
– не відповідність її стандартним засобам створення веб-служб;
– сильна об’єктна орієнтованість, у той час як, за спільною думкою, найбільш ефективною була сервіс-орієнтована архітектура.

Опинившись у такому, як здавалося безвихідному положенні, у додаток до GGF ще безліч комерційних і некомерційних організацій, з метою вирішення виниклих проблем, об’єднали свої зусилля в рамках різних консорціумів і взялися до розробки нового покоління стандартів веб-сервісів. Але й у цьому випадку всупереч великим очікуванням їхні зусилля дали не зовсім той результат, на який розраховували з самого початку. Причиною цього став той факт, що кожен консорціум, не погоджуючи свої дії з іншими розробниками, намагався створити свій власний пакет стандартів, що закономірно призвело до великої кількості різних несумісних стандартів, спрямованих на вирішення тих самих завдань.

Як уже згадувалося вище (стор. 49), піонером в галузі стандартизації Грід-технологій стала GGF – громадська організація, яка поєднує понад п’ять тис. фахівців, які працюють в галузі розподілених обчислень і технологій Грід. Основна мета GGF – сприяння розвитку, реалізації технологій Грід і розгортанню додатків шляхом створення основних документів (технічних специфікацій, описів досвіду створення застосувань, керівництв тощо).

Трохи пізніше до розв’язання проблеми стандартів Грід крім GGF під-

ключився консорціум Enterprise Grid Alliance – Альянс Ініціатив Грід-розробників (EGA). Цілі цього альянсу значною мірою відрізнялися від цілей GGF. Альянс був зацікавлений у розвитку ідей Грід і в застосуванні Грід-технологій не при вирішенні наукових або технічних завдань, які вимагають значних обчислювальних потужностей, а навпаки – у бізнес додатках, для яких необхідні великі обсяги баз даних. Тому в центрі уваги альянсу виявилися статичні Грід-структури, які розташовані в одному географічному місці і є складовою звичайного корпоративного центру обробки даних.

Говорячи про ці консорціуми не можна не звернути уваги на список компаній-учасників, які входили або підтримували GGF і EGA. Список очолюють такі гранди IT-індустрії, як IBM, Intel, Microsoft, HP, Fujitsu Technology Solutions, NEC, Network Appliance, Oracle, Sun Microsystems, EMC ну й ціла низка менших за розміром компаній, які долучилися до них. Причому одні компанії підтримували перший напрямок розвитку Грід, а інші – другий.

На щастя, незважаючи на розходження в підходах до концепції Грід, ці дві провідні Грід-організації оголосили про своє об’єднання. У результаті була створена нова організація під назвою Відкритий Грід Форум – Open Grid Forum (OGF). Таке об’єднання привело до значного прискорення впровадження Грід-технологий в усьому світі.

OGF успадкував від EGA всі досягнення цього альянсу в галузі промислової експертизи і орієнтацію на швидкі практичні результати, а від GGF – багатий досвід відкритого співробітництва в галузі досліджень Грід-систем і в розробці стандартів. OGF є відкритим форумом для всіх ключових фігур і організацій Грід-співтовариства. Форум сприяє виявленню потенційних проблем у Грід-технологіях, на основі обговорень у робочих групах, допомагає розв’язанню цих проблем, а також виконує роботи з розробки специфікацій для прискорення впровадження Грід.

Як відкрита організація зі стандартизації, OGF тісно співпрацює з іншими організаціями, які розробляють і впроваджують ті або інші нові стандарти. Список організацій, які виконують роботи зі стандартизації і з якими співпрацює OGF, доволі великий, сюди входять: Organization for the Ad-

vancement of Structured Information Standards (OASIS), World Wide Web Consortium (W3C), Distributed Management Task Force (DMTF), Web Services Interoperability Organization (WS-I), Internet2 і Liberty Alliance. Така співпраця повною мірою сприяє узгодженню існуючих промислових стандартів і розробці нових специфікацій.

Створення OGF і його тісна співдружність із спорідненими організаціями дозволило зблизити дві раніше незалежні гілки веб-служб. З одного боку – це сімейство специфікацій Web Services Distributed Management (WSDM), яке включало стандарти Web Services Resource Framework (WSRF) і WS-Notification і активно підтримувалося IBM, HP, а також рядом інших організацій. З іншого боку – це сімейство специфікацій WS-Management, які включали WS-Eventing, WS-Transfer і WS-Enumeration, і прихильниками яких були Microsoft, Intel і інші розробники стандартів. У результаті був розроблений спільний стандартний набір специфікацій веб-служб і вироблена спільна схема використання цих специфікацій. Нові розроблювальні специфікації в остаточному вигляді вже підтримують всі базові концепції, уведені спочатку в OGSI і згодом реалізовані в WSRF/WS-*.

Таким чином, уже в значній мірі була вирішена проблема розробки всеосяжних стандартів управління різними системами, які базуються на веб-службах, у тому числі й управління Грід-системами.

3.7 Грід-специфікації Web Services Resource Framework і WS-Notification

Найбільше визнання й поширення серед розробників одержали такі стандарти веб-сервісів:

– WS-Security – стандарт, призначення якого – рекомендації із забезпечення безпеки веб-сервісів, основною задачею WS-Security є забезпечення цілісності, конфіденційності й автентичності повідомлення і його відправника при одночасному збереженні відкритості для розширень;

– WS-Addressing – специфікації, що дозволяють виконувати маршрутизацію й адресацію SOAP-повідомлень між службами й клієнтами, ця специфікація дозволяє включати в заголовки SOAP-повідомлень інформацію про маршрутизацію. З одного боку вона дозволяє описувати адреси кінцевих точок, які є постачальниками або споживачами повідомлень, а з іншого боку – дозволяє зв'язувати конкретне повідомлення з кінцевою точкою. При цьому важливо розуміти, що заголовок SOAP-повідомлення є частиною протоколу прикладного рівня, таким чином WS-Addressing – це спосіб відв'язати інформацію про маршрут повідомлення від транспортного рівня моделі OSI і передати її на прикладний рівень;
– WSRF, WS-Notification – рекомендації зі створення й робота зі станами веб-сервісів.

Як видно зі сказаного раніше, мета, переслідувана при створенні цих специфікацій, полягала в зближенні OGSA з веб-сервісами і з SOA. За допомогою засобів, які відповідають цим специфікаціям, створюється і реалізується новий підхід до моделювання і управління станом у контексті веб-сервісів. Іншими словами, веб-служби одержали можливість реалізувати стан, а це саме те, що відрізняє Грід-сервіси від веб-сервісів. Подібний підхід дозволяє користувачам, перебуваючи в контексті веб-сервісів, контролювати й змінювати стан доступних їм ресурсів.

До стандартів WSRF були включені, наприклад, такі специфікації:

– WS-Resource Lifetime – специфікація визначає способи керування життєвим циклом ресурсу і описує веб-сервіси для ліквідації ресурсу, тобто фактично визначається механізм припинення існування якогось WS-Resource (включаючи обмін повідомленнями), це дозволяє негайно або через зазначений час видалити ресурс;
– WS-Resource Properties – ця частина визначає способи запитування й модифікації ресурсів, описуваних XML-документами Resource Property, специфікує механізм асоціювання екземпляра сервісу з інтерфейсом, який описує веб-сервіс (включаючи обмін повідомленнями)

і додатково дозволяє виділяти, змінювати й знищувати властивості ресурсу WS-Resource;

– WS-ServiceGroup – визначає способи подання і управління колекціями веб-сервісів і/або WS-Ресурсами, тобто специфікує інтерфейс до набору гетерогенних веб-сервісів;
– WS-BaseFaults – специфікація визначає базовий XML-тип, використовуваний при обміні повідомленнями в веб-сервісах для інформування про збої, іншими словами визначає механізм обробки повідомлень про помилки;
– WS-RenewableReferences – частина стандарту, яка визначає механізм розширення звичайної системи адресації WS-Addressing, прийнятої у веб-сервісах.

Слід відзначити, що специфікації WS-Notification безпосередньо не входять до складу набору WSRF, але розроблені на його основі й визначають засновані на принципах підписки/публікації, механізми обробки повідомлень про події. Набір WS-Notification складається із трьох специфікацій:

– WS-BaseNotification – ця специфікація визначає інтерфейси постачальника (NotificationProducers) і споживача (NotificationConsumers) асинхронних повідомлень, а також основні виконувані функції при розсиланні повідомлень і передплаті на них, процесів припинення/поновлення підписок і контролю строку підписки.
– WS-BrokeredNotification – специфікація дозволяє об’єкту, який не відноситься до веб-сервісу, створювати асинхронні повідомлення і розсилати їх через особливу посередницьку службу – NotificationBroker.
– WS-Topics – специфікація для організації і категоризації тем для підписки, інакше кажучи, цей список дає уявлення про набір функціональностей, для яких цими специфікаціями визначаються ті або інші стандарти.

Цей список специфікацій далеко не повний, оскільки окремі галузі вимагають відразу декількох специфікацій. Широта тематичного охоплення

вражає уяву, але закінченість і повнота такого переліку залишаються під сумнівом.

Розробка WSRF з набором специфікацій WS-* дозволила повністю відмовитися від інтерфейсів (PortType), визначених в OGSI, а також від операцій, які ними пропонуються, і замінити їхніми аналогами зі специфікацій WSRF. Відповідність деяких специфікацій WSRF і операцій інтерфейсів OGSI наведені в табл. 2.

Таблиця 2 – Відповідність між специфікаціями OGSI і WSRF

OGSI	WSRF
Grid Service Reference	WS-Addressing Endpoint Reference
Grid Service Handle	WS-Addressing Endpoint Reference
HandleResolver portType	WS-RenewableReferences
Service data definition & access	WS-ResourceProperties
GridService lifetime management	WS-ResourceLifeCycle
Notification portTypes	WS-Notification
Factory portType	Treated as a pattern
ServiceGroup portTypes	WS-ServiceGroup
Base fault type	WS-BaseFaults

У результаті, розкритикована розробниками латка OGSI, була виключена зі структур для побудови Грід-систем, і їхня архітектура стала просто трирівневою так, як це показано на рис. 15.

Сказане вище дає можливість стверджувати, що розроблені на основі OGSA і WSRF Грід-сервіси – це сервіси, які підтримують надання повної інформації про поточний стан екземпляра сервісу, можливість надійного й безпечного виконання, управління часом життя, розсилання повідомлень про зміну стану екземпляра сервісу, управління політикою доступу до ресурсів, управління сертифікатами доступу й віртуалізації.

Рисунок 15 − Трирівнева структура для побудови Грід-систем

Грід-сервіси в обов'язковому порядку підтримують такі стандартні інтерфейси:

– пошук – цей інтерфейс забезпечує Грід-застосуванням необхідні механізми для пошуку доступних сервісів і визначення їхніх характеристик;
– динамічне створення сервісів – надає можливість динамічного створення й управління сервісами – це один з базових принципів OGSA, який вимагає наявності сервісів створення нових сервісів;
– управління часом життя – потрібний тому, що розподілена система повинна забезпечувати можливість знищення екземпляра Грід-сервісу;
– повідомлення – для забезпечення роботи Грід-застосування набори Грід-сервісів повинні мати можливість асинхронно повідомляти один одного про зміни в їхньому стані.

У всякому випадку, підхід на основі OGSA припускає використання стандартів, які гарантують функціональну сумісність різних реалізацій Грід-систем. Використання специфікацій OGSA, які постійно вдосконалюються, приводить до того, що програмні засоби проміжного рівня різних постачальників одержують OGSA-сумісні функціональні можливості й інтерфейси. А це, в свою чергу, надає розробникам Грід-систем можливості поєднувати в єдине ціле компоненти різних постачальників ПО для побудови масштабова-

них, функціонально-сумісних (інтероперабельних), надійних і легко керованих Грід-систем.

Сьогодні при розробці й введенні робочими групами GGF/OGF нових стандартів, OGSA обов'язково враховує як усі раніше існуючі, так і всі стандарти зі всіх сегментів веб-сервісного співтовариства, які з'являються знову. Це повною мірою забезпечує потреби різних категорій користувачів і розробників Грід-систем. Наприклад, на першому етапі специфікації OGSA, методи керування Грід-системами визначалися на основі веб-сервісного стандарту Web Services Distributed Management (WSDM) міжнародної організації OASIS, хоча також існувала й можливість використання альтернативного набору специфікацій, заснованого на стандарті асоціації DMTF. У свою чергу, стандарти WSRF, які обумовлюють способи представлення і управління ресурсами/сервісами зі станом, і специфікації WS-Notification, які забезпечують загальний механізм повідомлень і підписки на повідомлення про події, були із самого початку розроблені й затверджені як стандарт OASIS для потреб побудови Грід-систем.

3.8 Структура Globus Resource Allocation Manager на основі Грід-сервісів

На сьогодні реалізація моделі OGSA за допомогою WSRF і супутніх стандартів WS-Addressing і WS-Notification найпоширеніша в середовищі Грід. Власне кажучи, на цій основі в свій час й створювався проект Globus із розробки й одержання інфраструктури для розподілених обчислень. Але в процесі розвитку проекту, основний акцент був перенесений з підтримки високопродуктивних обчислень у бік сервісів підтримки віртуальних організацій. Загалом метою створення Globus було надання можливості застосуванням працювати з розподіленими різнорідними обчислювальними ресурсами як з єдиною віртуальною машиною. Основна спрямованість проекту – обчислювальні Грід-системи.

Базовим елементом системи виступає Globus Toolkit, який описує базо-

ві сервіси й можливості, необхідні для створення обчислювальних Грід-систем. Система Globus надає високорівневим застосуванням доступ до Грід-сервісів, кожен з яких може використовуватися застосуванням або розробником для досягнення власних цілей, тому окремі сервіси повинні бути ізольовані й мати чітко визначені програмні інтерфейси.

Раніш (розділ 2.4.1 , стор. 53) вже розглядалася загальна архітектура ППЗ Globus Toolkit, але спираючись на Грід-сервіси, його структура може бути представлена так, як це зображено на рис. 16.

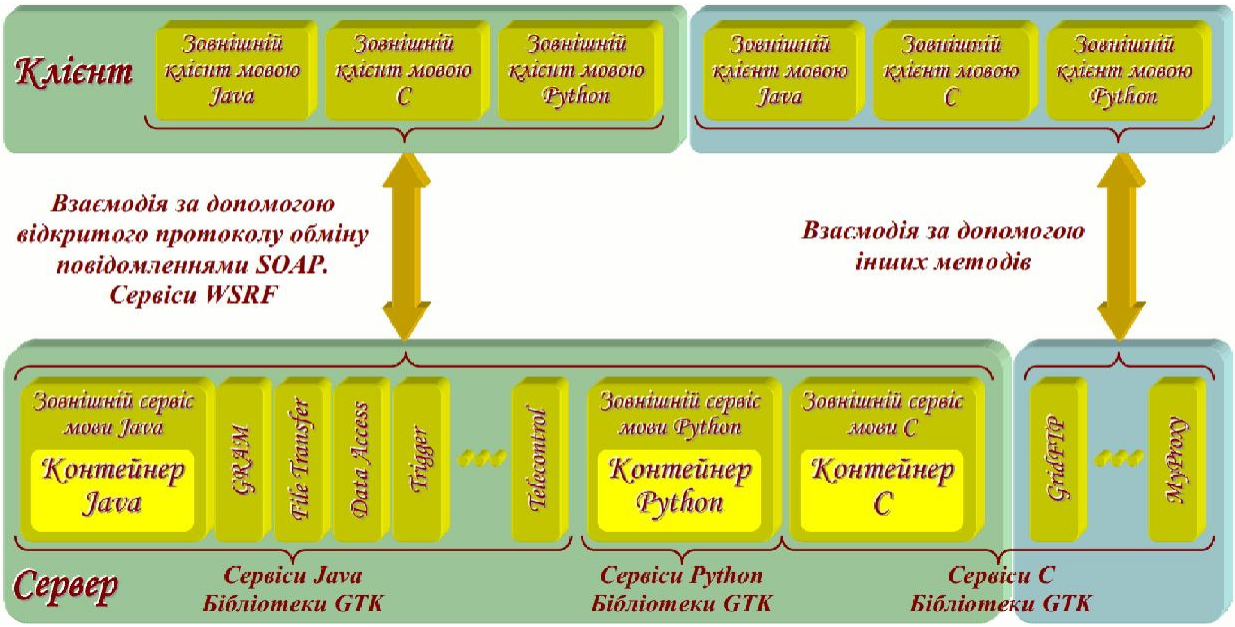

Рисунок 16 − Структура ППЗ Globus Toolkit на основі Грід-сервісів

Тут системою Globus надаються такі базові Грід-сервіси:

– сервіс і протокол GRAM – використовується для розподілу обчислювальних ресурсів і контролю обчислень, з використанням даних ресурсів;
– розширена версія протоколу передачі файлів GridFTP, яка використовується для організації доступу до даних, включаючи питання безпеки й паралелізму високошвидкісної передачі даних;
– контейнери для користувальницьких сервісів, які підтримують авте-

нтифікацію, управління станом, пошук і т.ін., а також забезпечують підтримку стандартів WSRF, WS-Security, WS-Notification;

- сервіси автентифікації й безпеки з'єднань GSI;
- розподілений доступ до інформації про структуру й стан системи розподілених обчислень;
- віддалений доступ до даних за допомогою послідовних і паралельних інтерфейсів;
- створення, кешування і пошук ресурсів, які зараз виконуються.
- бібліотеки, для забезпечення взаємодії сторонніх застосувань з GTK і/або з користувальницькими сервісами.

Однією з найважливіших частин системи Globus є сервіс GRAM – менеджер розподілу ресурсів Globus. Це сервіс призначений для створення/видалення процесів. Він встановлюється на кожному обчислювальному вузлі Грід-системи, роль такого вузла може виконувати як проста робоча станція (навіть персональний комп'ютер), так і потужний обчислювальний кластер або суперкомп'ютер. Місце GRAM в спільній архітектурі засобів управління ресурсами Globus показане на рис. 8, а його місце серед Грід-сервісів GTK на рис. 16.

Як видно з цього рисунка, сервіс GRAM на кожному обчислювальному вузлі підтримує роботу контейнерів, які відповідають за виконання програм мовами програмування Java, C, Python тощо, так само, як з іншого боку він через власний інтерфейс взаємодіє з великою кількістю локальних систем управління ресурсами.

На рис. 17 наведена структурологічна схема роботи GRAM.

Щоб на якому-небудь обчислювальному вузлі можна було у віддаленому режимі запускати на виконання програми, на ньому повинен виконуватися спеціальний процес, який називається Gatekeeper. Gatekeeper працює в привілейованому режимі і призначений для виконання такої низки функції:

- проведення взаємної автентифікації між ресурсом і клієнтом;
- аналіз підмножини RSL-запитів, які надходять до нього від високорівневого менеджера ресурсів;

– відображення клієнтського запиту на обліковий запис деякого локального користувача;
– запуск від імені такого локального користувача спеціального процесу, який називається Job Manager, і передачу йому переліку ресурсів, які вимагаються користувачем.

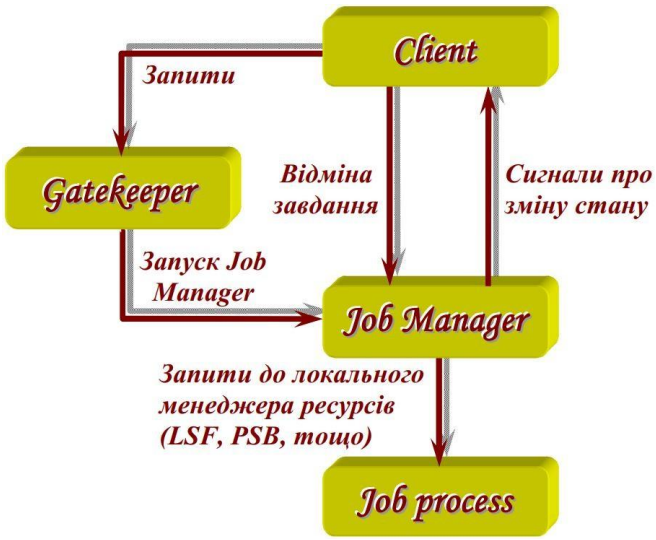

Рисунок 17 – Загальна структура і взаємодія елементів Globus Resource Allocation Manager

Після того, як Gatekeeper виконає свою роботу, Job Manager запускає завдання (процес або кілька процесів) і здійснює його подальший моніторинг, повідомляючи клієнта про помилки й інші події, які трапляються в системі. Для кожного користувача, незалежно від кількості надісланих ним завдань, Gatekeeper запускає лише один Job Manager, який і управляє всіма завданнями даного користувача. Job Manager завершує свою роботу тоді, коли вичерпаються і завершаться всі завдання користувача.

Більш наочно зрозуміти принципи керування GRAM дозволяє наведена на рис. 18 докладна організація GRAM вже на базі Грід-сервісів і взаємодії між ними. Реалізація GRAM надана на прикладі контейнера мови Java, але для інших контейнерів реалізація така ж сама.

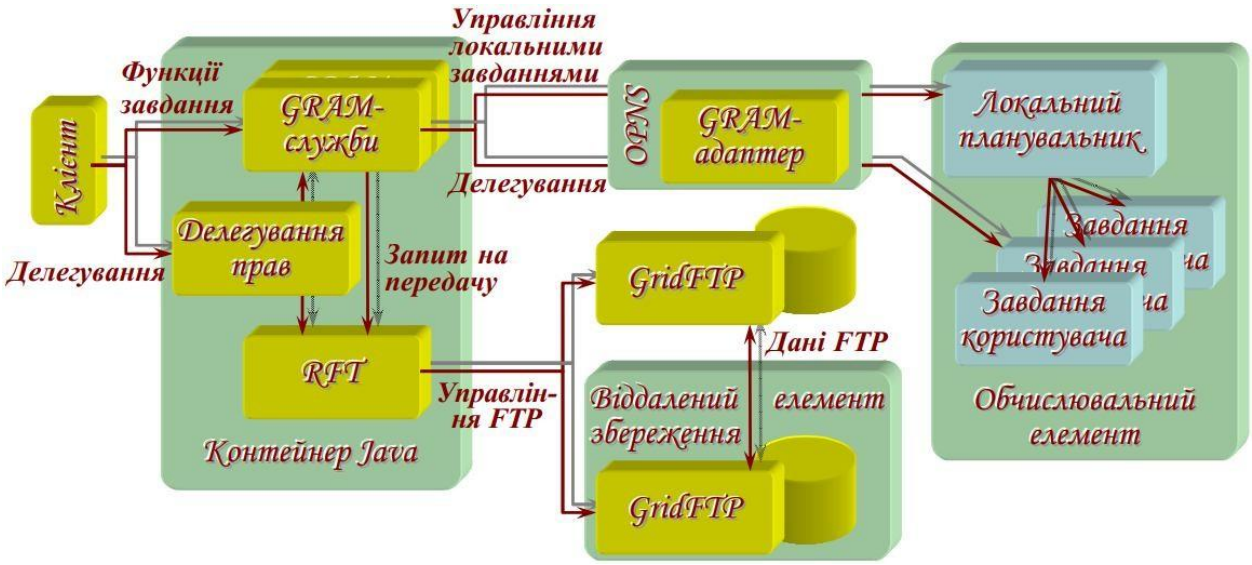

Рисунок 18 − Компоненти GRAM на прикладі Java контейнера

Основними елементами GRAM з цієї точки зору є:

– специфічні GRAM-служби для створення, спостереження й управління роботами;
– загальноцільові служби для передачі прав і надійної передачі даних RFT;
– планувальні GRAM адаптери, які основані на відкритих продуктах для мережного програмного забезпечення – Open Products, Networks Software (OPNS), призначені для передачі запитів у відповідні локальні планувальники;
– GridFTP сервери для виконання конвеєрних команд;
– програми командного рядка (табл. 3).

Таблиця 3 – Команди програмних оболонок Грід-систем

Команди	Опис [1)]
cas-*	Загальна служба авторизації
globus-credential-*	Інсталяція й відновлення прав у службі передачі прав
myproxy-*	Служба MyProxy

Кінець таблиці 3

Команди	Опис [1)]
grid-ca-sign	Простий видавець для генерації X.509 прав
gsi*	OpenSSH с ssh, scp, stpf клієнтами
grid-*	Управління X.509 проксі мандатами й файлами розміщення
globus-url-copy	GridFTP клієнт
rft*	Клієнт надійної передачі файлів
globus-rls-*	Клієнт RLS, адміністрація, сервер
globusrun-ws	Клієнт GRAM
mds-servicegroup-add	Додавання входу в MDS групу служб
globus-*-container	Запуск і зупинка контейнера Globus
wsrf-*	Взаємодія з WSRF властивостями ресурсів
wsn-*	Створення й управління WS-Notification subscriptions.

[1)] Кожен рядок таблиці містить деяке сімейство програм

Ці команди створюють інтерфейс користувача і надають можливість викликати компоненти GRAM безпосередньо із програми користувача, які написані на Java, C або Python.

Як приклад команди завдання роботи в GRAM від клієнта, можна надати команду для виконання програми «/bin/touth» з аргументом «touched_it», яка виглядає так:

```
% globusrun-ws -submit -job-command /bin/touch touched_it
```

Компоненти цієї команди задають назву команди `globusrun-ws`; прапор завдання – `submit`, який вказує, що це й є завдання; прапор роботи – `job-command`, який вказує, що залишок рядка команди задає ім’я програми і її аргумент. Якщо немає помилок, то завдання буде прийняте й програма запущена. Після закінчення завдання виводиться інформація про його статус:

```
Job ID: uuid:c51fe35a-4fa3-11d9-9cfc-000874404099
Termination time:     02/17/2019 20:47 GMT
Current job state:    Active
Current job state:    CleanUp
Current job state:    Done
Destroying job...     Done.
```

Після того, як завдання запущене, користувач залишається постійно підключеним до нього. Це необхідно, наприклад, для одержання статусної інформації, а також для того, щоб при необхідності припинити виконання завдання. Інтерактивний режим корисний у багатьох обставинах, наприклад, у пакетному режимі користувач на свій розсуд у будь-який момент може запросити статусну інформацію. Види статусної інформації зазначені в табл. 4.

Таблиця 4 – Стани робіт в GRAM

Стан	Опис
Unsubmitted	Робота ще не задана
StageLn	Робота очікує етапу виконання або вхідних файлів для закінчення
Pending	Локальний планувальник ще не спланував роботу на виконання
Active	Робота виконується
Suspended	Виконання роботи було припинено
StageOut	Виконання роботи завершене, вихідні файли виведені
CleanUp	Виконання роботи завершене, виконується очищення задач
Done	Робота завершена успішно
Failed	Робота зазнала невдачі

У наведеному прикладі команди запуску програми передбачалося, що програма, виконання якої передбачається, розташована на виконуючому комп’ютері, вхідні дані до неї передаються як аргумент, а результати збері-

гаються у файлах на виконуючому комп'ютері. Однак, бувають ситуації, коли програму, яка зараз буде виконуватись, або інші потрібні їй файли необхідно передати в цільовий комп'ютер або одержати від нього результат. Для цього в описі завдання використовуються спеціальні директиви файлових передач, які засновані на RFT синтаксисі, у них вказуються адреси передавальної й приймаючої сторони (URL). Приклад завдання на передачу файлів наведений нижче.

```
<fileStageIn>
  <transfer>
    <sourceUrl>
      gsiftp://submit.host:2888/tmp/mySourceFile
    </sourceUrl>
    <destinationUrl>
      file:///${GLOBUS_USER_HOME}/my_file
    </destinationUrl>
  </transfer>
</fileStageIn>
```

Різні розробники часто створюють деякий власний діалект мови RSL. Сьогодні, на жаль, існує багато різних форматів синтаксису, тому за відомостями про структуру мови обов'язково треба звертатися до документації конкретного розробника.

3.9 Кластерна система Condor

Як вже згадувалось раніше компонент GTK GRAM постійно взаємодіє з локальними засобами управління ресурсами вузла, серед яких доволі великою популярністю користується вільно доступний менеджер ресурсів Condor.

Кластерна система Condor була створена групою розробників Університету Wisconsin-Madison, де й була понад 20 років тому розгорнута її перша конфігурація. На сьогоднішній день в університеті велика кількість настільних Unix станцій включена в мережу Condor і надає доступ для роботи користувачам з усього світу. Система вільно поширюється в завантажувальних модулях.

3.9.1 Основні можливості Condor

У багатьох випадках, особливо якщо мова йде про обчислювальні задачі, користувачам набагато важливіше не швидкість виконання одного завдання, а число завдань, які можна виконати за доволі тривалий час. Інакше кажучи, число операцій за місяць або рік. Такий постулат лежить в основі технології ефективного використання наявних комп'ютерних ресурсів – High Throughput Computing (HTC). Система Condor – це ПЗ для підтримки середовища HTC, утвореного станціями на платформі Unix і Windows-NT. Незважаючи на те, що Condor може управляти спеціалізованими кластерами з робочих станцій, його велика перевага – це здатність розподіляти звичайні комп'ютерні ресурси, доступні в будь-якій лабораторії або офісі. Іноді ще Condor називають «мисливець за вільними станціями». Це тому, що замість того щоб запускати завдання на власній машині, користувач звертається до системи, яка шукає тимчасово вільні машини в мережі і запускає на них завдання. Коли машина перестане бути вільною (користувач, який повернувся з обіду натиснув клавішу або кнопку миші, або машина одержала команду вивантаження ОС), Condor перериває виконання завдань і здійснює їхню міграцію на іншу вільну машину, а потім перезапускає завдання на ній з перерваного місця. В разі, коли вільних машин немає, завдання ставиться в чергу де й чекає вивільнення ресурсів. Реалізований в Condor механізм управління завданнями, передбачає супровід черг, складання розкладів, схему пріоритетів і класифікацію ресурсів, дозволяє користувачеві бути в курсі про стан своїх завдань, не піклуючись про їхню подальшу долю.

Запропонована в Condor парадигма роботи орієнтована на виконання тривалих завдань, час обчислення яких вимірюється годинами. Наприклад, якщо необхідно п'ять годин обчислення, то завдання може почати роботу на машині «А», а після трьох годин мігрувати на машину «В», де завдання й завершиться через дві години, сповіщення про це одержить користувач завдання. Цілком можливий і більш глибокий розподіл загального часу виконання, наприклад, на 250 різних квантів. При цьому Condor не вимагає приєднання

машин до складу мережних файлових систем, таких як NFS або AFS, а всі станції можуть розміщатися в різних доменах. Крім звичайного режиму обробки контрольних точок із запам'ятовуванням стану, Condor може періодично зберігати поточний стан, що особливо корисно для забезпечення надійності роботи у випадку збою комп'ютерів з пулу або страховки від більш прозаїчних речей на кшталт мимовільного вимикання станції без операції «shutdown».

Крім виконання міграції на вільні машини Condor забезпечує управління розподіленими ресурсами. На відміну від інших систем управління пакетною обробкою (СУПО), у яких адміністратору або користувачу потрібно вручну редагувати реквізити завдання в черзі, щоб задовольнити вимоги виконуючого комп'ютера і запустити завдання на обчислення, Condor повністю автоматизує процес розподілу завдань, використовуючи для цього об'єктну технологію ClassAd. Ця технологія працює подібно до «Дошки оголошень» (звідки й назва). Всі машини, які доступні пулу Condor, повідомляють свої можливості в рубриці «Пропозиція»: обсяг вільного дискового простору, тип і швидкодія процесора, обсяг оперативної пам'яті, середня завантаженість і т.ін. З іншого боку, користувач описує потреби свого завдання в рубриці «Запит ресурсів». Condor працює в ролі брокера, який задовольняє заявки з рубрики «Запит ресурсів» ресурсами, представленими в рубриці «Пропозиція». Одночасно система забезпечує урахування і ведення декількох рівнів пріоритетів з аналізу заявок: пріоритет призначення ресурсів, пріоритет використання і пріоритет серед машин, які задовольняють однакові заявки.

При роботі Condor від користувача не потрібно спеціальної реєстрації (account або login) на машинах, де буде виконуватися його завдання – система сама виконує реєстрацію за допомогою технології віддаленого виклику RSC (Remote System Call), що дозволяє перехоплювати системні виклики при виконанні операцій читання/запису файлів і перенаправляти їх на машину звідки було запущене завдання. Таким чином, користувач може не хвилюватися про доступність файлів на віддаленій машині або одержання для себе account. Система не вимагає використання деякого спеціального ПЗ й здатна

виконувати звичайні Unix і Windows програми, а користувачеві потрібно тільки перезібрати своє завдання з бібліотеками Condor. Права власників робочих станцій, які включені у пул Condor ні в якій мірі не обмежуються і вони можуть використовувати свої машини так, як це їм необхідно, і при цьому з їх боку не потрібно ні яких спеціальних зусиль по додатковому адмініструванню.

3.9.2 Архітектура Condor

3.9.2.1 Апаратна структура

Апаратна архітектура системи включає три типи комп'ютерів (рис. 19), об'єднаних у єдиний пул Condor: центральний менеджер (Central Manager), виконуючі машини (Execute Machine) і машини, з яких здійснюється запуск завдань (Submit Machine).

Рисунок 19 – Апаратна архітектура Condor

Центральний сервер – це один (будь-який) з комп'ютерів у пулі, який

повинен виконувати функції менеджера, збирати інформацію про всі машини і бути брокером між наявними ресурсами й користувачем. Збій у роботі комп'ютера, на якому встановлений центральний менеджер, спричиняє зупинку в роботі всього комплексу.

Виконуючі машини призначені для виконання обчислень. Будь-яка машина в пулі, у тому числі центральний сервер і машина для запуску завдань, може бути сконфігурована ще й для виконання обчислювальних завдань Condor.

Машина для запуску завдань це знов-таки будь-яка машина з пулу, з якої здійснюється запуск завдання на виконання. Бажано, щоб на цій машині було досить ресурсів для роботи керуючих демонів. Кожне завдання, яке перериває ться для обробки контрольної точки, має асоційований файл розміром з файл самого завдання, тому на диску машини для запуску мусить бути достатньо вільного місця для збереження цього файла.

Крім цих машин у пул може включатись ще й «Сервер контрольних точок» (Checkpoint Server), на якому зберігаються всі робочі файли для обробки контрольних точок перерваних завдань.

3.9.2.2 Програмна архітектура Condor

З програмної точки зору система Condor складається з деякої кількості процесів-демонів різного призначення.

Демон condor_master – це демон, який контролює роботу всіх інших демонів Condor, запущених на кожній машині пулу. Демон виконує адміністративні команди системи. Даний демон повинен запускатись на кожній машині з пулу незалежно від її призначення.

Демон condor_startd призначений для представлення ресурсів у пулі. Він запускається і працює на кожній виконуючій машині і у випадку її готовності до виконання завдання запускає демон condor_starter.

condor_starter – це демон для запуску завдань і моніторингу процесу їхнього виконання на конкретній машині. Після завершення завдання демон

посилає повідомлення машині, з якої запущено завдання, і припиняє свою роботу.

Демон condor_schedd призначений управляти ресурсами, необхідними для пулу Condor і мусить працювати на кожній машині, для запуску. При запуску завдання відбувається звертання до condor_schedd, який розміщає завдання в черзі завдань. При збої демона condor_schedd ніяка подальша робота системи неможлива.

Демон condor_shadow, який працює на машині, для запуску завдання виконує функції менеджера ресурсів. Всі системні виклики з віддалених машин пересилаються демону condor_shadow, виконуються на його машині, а результати роботи відправляються назад до віддаленої машини і завдання.

Збір інформації про стан пулу Condor виконує спеціальний демон – condor_collector. Всі інші демони періодично посилають цьому демонові дані про свій стан.

Демон condor_negotiator – це демон керування станом Condor. Періодично демон запускає цикл погоджування системи, протягом якого збираються дані про стан ресурсів, про поточний стан кожного демона condor_schedd з метою зміни пріоритетів і т.ін.

Демон condor_ckpt_server реалізує функції сервера обробки контрольних точок, його задачею є збереження поточного стану завдання.

При інсталяції Condor насамперед розгортається ПЗ на комп’ютері, який відіграє роль центрального сервера. На ньому на виконання запускаються демони condor_collector і condor_negotiator. Демони за мету мають виконання функцій зв’язку між всіма наявними машинами і всіма завданнями, які очікують виконання. У випадку збою цієї машини-сервера всі активні на цей момент завдання будуть продовжувати виконуватись до завершення, але нові завдання вже запускатися не будуть. Крім того, інструменти системи Condor, у своїй переважній більшості, стають недоступними.

Всі демони Condor рекомендується запускати з правами root, у противному випадку система не гарантує надійної роботи. На всіх машинах пулу також корисно створити користувача з ім’ям «Condor». При цьому, демони

будуть створювати файли (наприклад log-файли), власником яких буде цей користувач, а його каталог home буде застосовуватися для зазначення місця розташування файлів Condor. У каталозі, який зазначається в параметрі LOCAL DIR файла конфігурації, на кожній машині пулу треба створити підкаталоги «spool», «log» і «execute». Каталог execute використовується для зберігання завдань Condor, які виконуються на даній машині, spool – необхідний для зберігання черг і файлів історії, а також файлів контрольних точок для всіх завдань від машини, яка їх запускає, log – це місце розміщення log-файлів кожного демона Condor.

3.9.2.3 Керування ресурсами

Як уже відзначалося, система Condor працює на кшталт брокера між ресурсами системи – пулом вільних машин і користувачами, які запускають свої завдання в системі. Крім наявних ресурсів, машини ще й повідомляють про умови за яких вони буду виконувати завдання від Condor і який тип завдань для них кращий. Наприклад, машина з ім’ям «ws327b-5» буде обробляти завдання тільки вночі при фізичній відсутності власників. Крім того, переважне право займати своїми завданнями цю машину буде належати, наприклад, викладачам кафедри інформатики.

З боку завдання користувача також виставляються власні вимоги до машини, наприклад наявність процесора, який добре виконує арифметичні дії з числами з плаваючою крапкою, або мають пам’ять не менше за 2 ГіБ. Всі ці вимоги поміщаються в ClassAd і доводяться до відома системи Condor. Інформацію з ClassAd можна вивести на монітор робочої станції користувача за допомогою статусних команд.

3.9.2.4 Підготовка завдання

Процес підготовки завдання передбачає опис системного оточення (Universe) системи і, можливо, зборку завдання разом з бібліотеками Condor,

за допомогою спеціальних команд компіляції.

Насамперед треба пам’ятати, що Condor виконує завдання в автоматичному режимі на тлі роботи обчислювального комплексу, тому треба перш за все переконатись, що програма здатна працювати без втручання користувача. Тобто тут треба подумати про перепризначення потоків stdout, stdin і stderr. Крім того, якщо є потреба у багаторазовому виконанні завдання на різних наборах вхідних даних, то для цього треба відповідним чином оформити завдання, щоб для кожного запуску була передбачена власна безліч вхідних і вихідних файлів. В системі Condor це дуже просто організувати так, як це зроблено у наступному прикладі завдання на виконання:

```
Executable = irbis
Requirements = Memory >= 32 && OpSys == «IRIX6» && Arch == «SGI»
Rank = Memory >= 64
Image_Size = 28 Meg
Input = in.$(Process)
Output = out.$(Process)
Error = err.$(Process)
Log = irbis.log
Queue 150
```

В цьому прикладі виконується запуск 150 разів програми «irbis» на обчислення. Відповідно до завдання програма повинна бути відкомпільована і зібрана на станції Silicon Graphics під управлінням ОС IRIX 6.x з пам’яттю не меншою за 32 МіБ і запушена на машині з пам’яттю не меншою за 64 МіБ. Файли stdin, stdout і stderr задані як: «in.xxx», «out.xxx» і «err.xxx» де xxx – поточний номер запуску програми від 000 до 150. Файл журналу «irbis.log» містить дані про міграції й переривання завдання.

При запуску завдань в Condor версії більшої за 6.1.12 можна використовувати п’ять різних режимів роботи: «Standard», «Vanilla», «PVM», «MPI» і «Globus». Режим «Standard» забезпечує максимально можливі способи міграції і надійності виконання завдання, однак, при його використанні є деякі обмеження на тип програм, які підлягають запуску. Режим «Vanilla» надає дещо бідніші можливості, але не накладає істотних обмежень на програми. Режим «PVM» призначений для програм, які використовують інтерфейс Parallel

Virtual Machine, а «MPI» потрібний для програм, які працюють під MPICH. Режим «Globus» дозволяє користувачам викликати систему Globus через інтерфейс Condor.

У режимі Standard здійснюється підтримка контрольних точок і виклик віддалених системних функцій (RSC), це дозволяє використовувати ресурси всього комп'ютерного пулу Condor через уніфікований інтерфейс. Однак для цього необхідно зібрати завдання з бібліотеками Condor за допомогою, наприклад, команди:

```
% condor_compile cc main.o tools.o -o program.
```

Можливо також використання будь-якого іншого компілятора gcc, g++, g77, cc, acc, c89, CC, f77, fort77 і завантажника.

Виклик віддалених системних функцій дозволяє створити для завдання максимально комфортне оточення, начебто мова йде про його виконання на машині користувача. При виконанні завдання на іншій машині, на машині, яка запустила завдання, стартує ще й додатковий процес – condor_shadow, який перехоплює кожну спробу завдання звернутися до системних функцій і виконує все сам, повертаючи завданню відповідний результат. Таким чином, якщо завданню треба, наприклад, відкрити файл, який є тільки на машині для запуску, демон condor_shadow знайде цей файл і переадресує звертання до нього на виконуючу машину, на якій у цей момент відбувається обчислення завдання.

Режим Vanilla призначений для програм, які не можна перезібрати заново з бібліотеками Condor (наприклад, через відсутність вихідних текстів). У цьому режимі не можливе використання контрольних точок і RSC. У випадку збою або вимикання виконуючої машини завдання буде або очікувати можливості поновлення її роботи, або буде заново запущене на іншій. Передача даних можлива через NFS або AFS.

Режим Globus необхідний для підключення до стандартного інтерфейсу Globus шляхом трансляції черги завдань в рядок мовою Globus RSL. Цей ря-

док передається у команду globusrun як її аргумент.

3.9.2.5 Звертання до Condor

За допомогою команди «submit» користувач розміщає своє завдання в системі Condor, указуючи виконуваний файл, імена файлів потокового введення/виведення для клавіатури й екрана (stdin і stdout) і адресу email для оповіщення про завершення завдання. Можна також указати, скільки разів треба виконувати програму, які дані при цьому треба використовувати, який тип машини потрібен для запуску завдання, ім’я виконуваного завдання, вихідні робочі каталоги, командний рядок для завдання. На основі цієї інформації створюється новий об’єкт ClassAd, який передається демонові condor_schedd, виконуваному на машині запуску.

Якщо потрібно одержати опис всіх можливих атрибутів для конкретної машини з пулу Condor, то можна викликати команду «status -l <ім'я машини>», наприклад так:

```
% condor_status -l ws319-4
```

і отримати щось подібне до:

```
MyType = «Machine»
TargetType = «Job»
Name = «dmitrii.cs.wisc.org»
Machine = «dmitrii.cs.wisc.org»
StartdIpAddr = «<128.105.83.11:32780>»
Arch = «INTEL»
OpSys = «SOLARIS251»
UidDomain = «cs.wisc.org»
FileSystemDomain = «cs.wisc.org»
State = «Unclaimed»
EnteredCurrentState = 892191963
Activity = «Idle»
EnteredCurrentActivity = 892191062
VirtualMemory = 185264
Disk = 35259
KFlops = 19992
Mips = 201
LoadAvg = 0.019531
```

```
CondorLoadAvg = 0.000000
KeyboardIdle = 5124
ConsoleIdle = 27592
Cpus = 1
Memory = 64
AFSCell = «cs.wisc.org»
START = LoadAvg -CondorLoadAvg <= 0.300000 && KeyboardIdle > 15 * 60
Requirements = TRUE
```

Тут поле «Activity» позначає стан машини: Idle – машина вільна, Busy – зайнята, Suspended – завдання припинене, Vacating – завдання в контрольній точці, Killing – завдання перерване. У полі «Arch» – відображається архітектура комп'ютера: «INTEL», «ALPHA», «SGI», «SUN4u» (Sun ULTRASPARC), «HPPA1» (PA-RISC 1.x PA-RISC 7000), «HPPA2» (PA-RISC 2.x PA-RISC 8000) і т.ін. Крім цього вказується середня завантаженість комп'ютера, кількість процесорів в ньому, обсяг вільного дискового простору, обсяг оперативної пам'яті, продуктивність в KFLOPS (за тестом Linpack) і MIPS (за тестом Dhrystone), ім'я машини, тип ОС, IP адреса. Вказується також стан комп'ютера в форматі:

- Owner – комп'ютер недоступний для Condor;
- Unclaimed – комп'ютер доступний з низьким пріоритетом;
- Matched – машина підходить для роботи, але планувальник Condor ще не додав її в перелік ресурсів,
- Claimed – комп'ютер присутній у пулі,
- Preempting – завдання вивантажене (можливо після контрольної точки) з іншої машини.

Система Condor не забезпечує автоматичну конвертацію на інші архітектури й за замовчуванням передбачається запуск завдання лише на виконуючій машині, яка має ту ж архітектуру, що й машина запуску. Якщо має місце розходження архітектур, наприклад завдання було запущено з комп'ютера з процесором Intel під управлінням ОС Linux, а в пулі Condor більшість станцій SPARC під управлінням ОС Solaris, то виконується компіляція й зборка завдання для виконання на іншій архітектурі. Для зміни архітектури потрібно долучити в опис завдання рядок:

```
Arch = «SUN4x» && OpSys == «SOLARIS251».
```

3.9.2.6 Управління процесом виконання завдання

У процесі виконання завдання користувач може спостерігати за ходом роботи за допомогою команд «q» і «status» і, можливо, модифікувати завдання. Якщо буде потрібно Condor може також посилати повідомлення про всі зміни стану завдання: обробка контрольної точки, міграція на іншу машину, розміщення в чергу очікування і т.ін. Повний список доступних машин можна одержати по команді «status -submitters»:

```
% condor_status -submitters
Name Machine Running IdleJobs MaxJobsRunning
ashoks@jules.ncsa.ui jules.ncsa 74 54 200
breach@cs.wisc.edu neufchatel 23 0 500
d.vlk@raven.cs.wisc raven.cs.w 1 48 200
ashoks@jules.ncsa.ui 74 54
jbasney@cs.wisc.edu 0 1
Total 109 103
```

Для ідентифікації виконуваних робіт можна використовувати команду:

```
% condor_q
```

і одержати приблизно таке:

```
Submitter: froth.cs.wisc.edu : <128.105.73.44:33847> : froth.cs.wisc.edu
ID      OWNER    SUBMITTED  CPU_USAGE          ST   PRI   SIZE  CMD
125.0   jbasney  4/10       15:35 0+00:00:00   U    -10   1.2   hello.remote
127.0   raman    4/11       15:35 0+00:00:00   R    0     1.4   hello
128.0   raman    4/11       15:35 0+00:02:33   I    0     1.4   hello

3 jobs; 1 unexpanded, 1 idle, 1 running, 0 malformed
```

Колонка ST (status) показує стан завдання в черзі:

– U – виконання без контрольних точок;

– R – виконання в цей момент;

– I – виконувалося раніше, а зараз у контрольній точці і очікує на вивільнення комп'ютера.

Цю ж інформацію можна одержати з файла стану «log».

При необхідності можна довідатися, на яких машинах виконується конкретне завдання із зазначеної машини, наприклад «ws327a-3@dit.edu»:

```
% condor_status -constraint
RemoteUser == «ws327a-3@dit.edu»
Name       Arch  OpSys       State    Activity   LoadAv  Mem   ActvtyTime
biron.cs.w INTEL SOLARIS251  Claimed  Busy       1.000   128   0+01:10:00
cambridge. INTEL LINUX       Claimed  Busy       0.988   64    0+00:15:00
falcons.cs INTEL SOLARIS251  Claimed  Busy       0.996   32    0+02:05:03
istat03.st INTEL SOLARIS27   Claimed  Busy       0.883   64    0+06:45:01
```

При виконанні завдання користувача йому призначається певний пріоритет, який може бути змінений командою «userprio». Чим менше значення пріоритету, тим краще й тим більше машин буде доступно для виконання. Спочатку пріоритет дорівнює 0.5 і змінюється залежно від запитуваних ресурсів. Чим менше у пулі машин, які відповідають вимогам завдання, тим вище значення пріоритету. Зміна відбувається автоматично, однак при конфігурації можна задавати періодичність корекції пріоритетів. З появою в пулі користувачів з вищим пріоритетом (з меншим його значенням) завдання користувача з меншим пріоритетом (з більшим значенням) будуть негайно вивантажені зі звільненням ресурсів для нового користувача. На доповнення до пріоритету користувача можна змінити пріоритет завдання усередині свого пакета завдань. Крім того, при запуску завдання йому можна позначити пріоритет «nice», таке завдання буде виконуватись на комп'ютерах, на яких немає інших Condor-завдань.

Після завершення роботи Condor сповіщає користувача по email, повідомляючи про кількість процесорів, задіяних при обчисленні і про загальний час перебування завдання в пулі.

Командою «rm», якщо з яких-небудь причин це потрібно, завдання можна зняти з обчислення, не чекаючи його завершення.

3.9.2.7 Виконання паралельних застосувань

Система Condor може бути використана для автоматичного створення середовища виконання паралельних завдань за технологією PVM шляхом динамічного надання вільних у цей момент машин. При цьому Condor-PVM діє в ролі менеджера ресурсів для демона PVM і, як тільки PVM – програмі потрібні додаткові вузли, формується запит для на пошук вільних машин з пулу і їхнє підключення до віртуальної машини PVM.

Серед загальних парадигм паралельного програмування у системі Condor реалізована найчастіше використовувана з них – «майстер-робітник». Як відомо, в цій парадигмі один вузол виконує роль майстра, який контролює процес виконання паралельних застосувань і виконуючого розподіл завдань на роботу іншим вузлам (робітникам). Робітники виконують обчислення й відсилають результат назад майстрові. В Condor роль майстра виконує машина для запуску завдання, а робітників – всі інші машини з пулу. Якщо робітник не в змозі виконати роботу з причини відключення або зайнятості, то майстер передає її іншому. Якщо майстер вирішує, що число робітників недостатньо для виконання всього обсягу робіт, він, через виклик pvm_addhosts(), запитує нових робітників потрібної архітектури, або намагається перерозподілити роботи між наявними робітниками, кожен з яких має власну продуктивність і «спеціалізацію».

Для роботи в режимі Condor-PVM не потрібно змін програми – використовується існуючий стандартний інтерфейс викликів PVM, таких як pvm_addhosts() і pvm_notify(). При описі завдання користувач лише вказує режим «universe = PVM», а система вже сама створює програму-майстер і умови для запуску робітників за допомогою команди pvm_spawn(). Фактично, звичайні PVM-програми мають повну двійкову сумісність із Condor.

Крім PVM система Condor підтримує ще й виконання паралельних додатків, які використовують парадигму MPI у версії від Argonne National Labs – версія MPICH, яка застосовується для виконання комунікацій між процесами в кластері робочих станцій цього дослідницького центру.

На відміну від PVM в реалізації Condor-MPICH поки що не існує диспетчеризації, тому можливе виникнення ситуації deadlock, коли жодне із завдань не зможе виконуватися.

Слід відзначити, що машини, призначені для виконання завдань MPI, повинні мати поділювану файлову систему, оскільки не можна використовувати RSC, як це передбачено в режимі Standard.

3.9.2.8 Обмеження Condor

Спочатку розробка Condor носила сугубо дослідницький характер, але під впливом зростаючого інтересу до розподілених обчислень (метакомп'ютингу) розробники одержали достатнє фінансування, що дозволило істотно розвинути функціональність системи й підвищити її надійність. А поки в поточних версіях є ряд обмежень.

У завданні допускається робота тільки одного процесу, тому не можна вказувати виклики fork(), exec(), system(). Не можна організувати взаємодію процесів через семафори, конвеєри й поділювану пам'ять. Не підтримуються такі функції як таймери, будильники й процеси з очікуванням: alarm(), getitimer() і sleep(). Незважаючи на те, що в Condor підтримуються сигнали і їхні дескриптори, є зарезервовані сигнали SIGUSR2 і SIGT-STP, які не можна використовувати в коді користувальницьких програм.

У системі не підтримуються команди міжпроцесорної взаємодії (IPC): socket(), send(), recv().

Не допускається використовувати множинні потоки на рівні ядра і застосовувати функції відображення файлів в пам'ять mmap() і munmap(). Не допускається блокування файлів, а всі файлові операції повинні бути ідемпотентними – «тільки читання» і «тільки запис», тому програми, які одночасно читають і пишуть в один файл, можуть працювати некоректно.

Якщо поточний робочий каталог на машині запуску, доступний через NFS, то в системи можуть бути проблеми з автоматичним монтуванням дисків (automounter), якщо буде потрібно розмонтувати каталог до завершення

завдання.

3.9.2.9 Додаткові обмеження Condor для ОС Windows

Кластерна система Condor з самого початку створювалася і розвивалася як система для комп'ютерів на платформі Unix і Unix-сумісних ОС. Однак операційна система Windows на сьогодні є стратегічною платформою для Condor. Розробники Condor постійно працюють над проблемою повної сумісності з Windows і намагаються зробити Condor таким же працездатним на Windows, як і на Unix.

Перенесення Condor з Unix на Windows є величезним завданням, оскільки багато компонентів Condor повинні тісно взаємодіяти з базовою операційною системою. Але, не дивлячись на величезні зусилля, навіть остання версія системи для Windows (8.9.0 від лютого 2019 року) є обрізаною версією. І хоча загалом, цей випуск для Windows працює так само, як і відповідний випуск Condor для Unix, однак у цій версії поки що:

- не реалізовані режими Standard і PVM, працює тільки режим Vanilla, а це означає відсутність можливостей роботи з контрольними точками, неможлива міграція і відсутні віддалені системні виклики RSC;
- задачі в режимі Globus не можуть бути запущені з платформи Windows, якщо тип мережі не є Condor;
- ще не підтримується доступ до файлів через мережевий ресурс, який вимагає автентифікації по протоколу Kerberos (наприклад, AFS);
- всі файли, необхідні для роботи завдання повинні розміщатися тільки на локальних дисках машини запуску.

Є ще ціла низка дрібніших проблем, які мабуть будуть вирішені розробниками у подальших версіях.

3.9.3 Конфігурування режимів роботи Condor

За замовчуванням, система Condor конфігурується так, щоб дозволити

будь-кому з бажаючих спостерігати за станом роботи в пулі, однак, при необхідності цей режим можна змінити, передбачивши, наприклад, різні рівні доступу. У системі можна встановити наступні рівні доступу: READ, WRITE, ADMINISTRATOR, OWNER, NEGOTIATOR.

Машини з рівнем доступу READ можуть тільки читати інформацію з Condor, наприклад: статус пулу, черга завдань і т.ін., нічого не змінюючи. Навппаки рівень WRITE – доступ для запису, надає можливість відновлення ClassAd і ініціацію інших машин на виконання завдання. Крім того, машини з рівнем WRITE можуть запитувати демон condor_startd для періодичного виконання запису по контрольній точці будь-якого завдання. На рівні ADMINISTRATOR можлива зміна пріоритетів користувача, активація або видалення машини з пулу, запит на реконфігурацію, рестарт і т.ін. OWNER – це рівень користувача-власника машини, режим найбільших обмежень для сторонніх користувачів. NEGOTIATOR – спеціальний рівень доступу, який означає, що спеціальні команди роботи з пулом (наприклад, опитування стану) будуть оброблятись тільки в разі, коли вони прийшли від центрального сервера. Можна вказати перелік команд, дозволених для виконання. Рівень NEGOTIATOR автоматично призначається системою, інші чотири рівні просто прописуються у файлі конфігурації. Права ADMINISTRATOR за замовчуванням належать тільки центральному менеджерові, OWNER – виконуючій машині.

3.10 Контрольні запитання і завдання для самостійної роботи

1. Надайте визначення сервісно-орієнтованої архітектури. Що є її типовими складовими? Чи можна на підґрунті цієї архітектури створювати надійні розподілені системи?
2. Що розуміється під сервісом (службою)? Чи можна різні сервіси об’єднувати між собою, а якщо так, то для чого це робиться? Що надають сервіси клієнтам? Опишіть простий цикл взаємодії сервісів.
3. Що таке слабкий зв’язок сервісів, які переваги має заснована на такому

принципі архітектура?

4. Докладно розкажіть про сервіси без станів і їхнє функціонування.
5. Які основні стандарти, протоколи і специфікації призначені для підтримки сервісів, як ці стандарти співвідносяться між собою?
6. Поясніть чому розробка Грід-систем спонукала відмовитися від веб-сервісів і перейти до формування концепції Грід-сервісів.
7. Спробуйте описати алгоритм роботи найпростішої сервісно-орієнтованої Грід-системи і поясніть чому навіть в такий простій системі не можна використовувати веб-сервіси.
8. Навіщо знадобилась розробка стандарту відкритої архітектури Грід-служб, чим в цьому стандарті відрізняються поняття сервісу й екземпляра сервісу?
9. Опишіть вимоги відкритої архітектури Грід-служб, які були враховані розробниками стандартів для неї.
10. Чому при розробці відкритої архітектури Грід-служб ще знадобилося розробляти додаткові специфікації в межах інфраструктури відкритих Грід-сервісів? Які механізми визначалися в специфікації інфраструктури відкритих Грід-сервісів?
11. Як виглядає багаторівнева структура для побудови Грід-систем на базі моделі відкритої архітектури/інфраструктури Грід-сервісів? Опишіть взаємодію рівнів такої структури і поясніть, чому спільне використання веб-сервісів і Грід-сервісів в одному середовищі неможливе.
12. Охарактеризуйте основні з істотних недоліків, які властиві моделі відкритої архітектури/інфраструктури Грід-сервісів, розкажіть про учасників спільноти фахівців, які брали участь у виправленні недоліків і до чого вони дійшли спільними зусиллями.
13. Назвіть і стисло охарактеризуйте стандарти веб-сервісів, які набули найбільшого визнання й поширення серед розробників.
14. Надайте стислий опис специфікацій, які були включені до стандартів WSRF. Назвіть і розкажіть про основні специфікації, які входять у набір WS-Notification.

15. Розкажіть про стандартні інтерфейси, які в обов'язковому порядку підтримують сучасні Грід-сервіси.
16. Що було метою створення проекту Globus, що є базовим елементом системи, як на основі Грід-сервісів представляється структура GlobusToolkit?
17. Для чого призначений менеджер розподілу ресурсів GRAM? Опишіть основні процеси, які функціонують в межах сервісу GRAM.
18. Що Ви можете сказати про основні елементи GRAM з цієї точки зору контейнерів різних мов? Опишіть взаємодію цих елементів.
19. Розкажіть про команди програмних оболонок Грід-систем на базі Globus Toolkit, наведіть деякі приклади застосування команд.
20. Охарактеризуйте основні можливості кластерної системи Condor.
21. Опишіть апаратну архітектуру системи Condor.
22. Як виглядає архітектура системи Condor з програмної точки зору? Розкажіть про призначення різноманітних процесів-демонів і їхню взаємодію.
23. В чому полягає процес підготовки завдання для системи Condor? Надайте приклад завдання.
24. Розкажіть про процес розміщення користувачем завдання в системі Condor за допомогою команди «submit».
25. Якими командами виконується спостереження і керування процесом виконання завдання в системі? Наведіть відповідні приклади.
26. Розкажіть про технології, які застосовуються для виконання паралельних застосувань в системі Condor.
27. В чому полягають обмеження системи Condor взагалі і для ОС Windows зокрема?
28. Як виконується конфігурування режимів роботи Condor?
29. Як виконується конфігурування режимів роботи Condor?

4 GRID І БАЗИ ДАНИХ. УПРАВЛІННЯ GRID-ОТОЧЕННЯМ

БД про стан засобів і пристроїв Грід, а також про хід виконання завдань на обчислювальних вузлах повинні зберігатись десь у структурі Грід. Тобто

повинна існувати і функціонувати деяка служба управління даними.

Як правило, у Грід-системах існують дві тісно пов'язані між собою служби, перша з яких працює з даними на файловому рівні, а друга, на противагу від неї, забезпечує управління системами БД, які оперують такими елементами як записи й поля.

4.1 Опис підсистеми управління даними

Підсистема управління даними (Data Management Subsystem, DMS) включає три сервіси, які підтримують доступ до файлів:

– ресурс зберігання даних (Storage Element, SE);

– сервіс каталогів (Catalog Services, CS);

– планувальник передачі даних (Data Scheduler, DS).

У розподіленому Грід-середовищі користувальницькі файли можуть зберігатися в багатьох екземплярах-репліках, розміщених у різних місцях. Задача CS і DS полягає в тому, щоб зробити процес управління репліками прозорим для користувача, так щоб застосування діставали доступ до файлів за їх глобальними для всього Грід іменам або дескрипторами метаданих. Тобто хоча й реально доступ до даних файлів відбувається через один із багатьох SE, які входять у Грід-систему, підсистема управління даними забезпечує глобальну файлову систему в масштабах усього Грід (концепція віртуалізації наборів даних). Клієнтське застосування для навігації по такій файловій системі може бути влаштоване як звичайна командна оболонка Unix/Linux – з використанням звичних команд зміни каталогів, перегляду файлів і т.ін. DMS в Грід не забезпечує захист файлів від несанкціонованого доступу, ця функція покладається на підсистему безпеки й контролю прав доступу, зокрема за допомогою списків контролю доступу (Access Control List, ACL).

Користувачі Грід-системи оперують логічними іменами файлів (LFN). Простір імен LFN є ієрархічним, точно так само як звичайна файлова система. Семантика простору імен LFN також практично така ж, як у файлової си-

стеми Unix/Linux. Отже, LFN – це логічне ім’я файла, тобто ідентифікатор файла, який є унікальним, але змінюваним: файли можуть бути перейменовані користувачем. Кожна ВО може мати свій власний простір логічних імен, якщо всі вони дотримуються цієї угоди.

Однак LFN не єдиний ідентифікатор, який пов’язується з файлом у Грід, хоча звичайному користувачеві інші імена файлів можуть знадобитися лише у вкрай рідких випадках. Прикладом форми LFN є наступний рядок:

```
/glite/myVO.org/production/run/07/123456/calibration/cal/cal-table100.
```

LFN повинен бути унікальним. В архітектурі Грід, інтерфейсом, який використовується, щоб управляти LFN є каталог файлів і реплік, він й забезпечує унікальність LFN. Якщо декілька ВО використовують той самий каталог спільно, то унікальність LFN повинна бути забезпечена для них всіх. Для цього ВО повинні домовитися про структуру імен у логічної ієрархії, наприклад, за допомогою префікса для кожної ВО. В такому випадку кожна ВО повинна забезпечити унікальність LFN лише у межах її власного простору імен. Це механізм, схожий на той, який використовується в інших розподілених файлових системах – таких як AFS.

У просторі імен LFN визначене поняття директорії. Директоріями можна управляти, як і в звичайній файловій системі: їх можна переглядати, додавати в них нові файли й піддиректорії або знищувати/перейменовувати вже існуючі. LFN містить повну ієрархію директорій, яким це ім’я належить. У попередньому прикладі LFN, одна з директорій така:

```
/glite/myVO.org/production/run/07/123456/calibration/.
```

Таким чином, у просторі LFN файли представлені у вигляді звичайної ієрархічної структури (каталоги, підкаталоги й т.д.). Користувачі можуть також копіювати логічні файли з інших логічних директорій у їхні власні директорії.

У логічному просторі імен файлів для кожного LFN можуть існувати

багато символічних посилань (співвідношення N:1). Якщо LFN переміщується або перейменовується, то за аналогією зі звичайною семантикою файлової системи символічне посилання залишається «повислим». З іншого боку, для логічних директорій символічні зв'язки, як правило, не підтримуються через труднощі відстеження таких директорій у середовищі розподілених просторів імен і каталогів. Щоб підтримувати систему LFN, ПЗ Грід повинні відображати логічні імена на фізичні, які забезпечують глобальну однозначну ідентифікацію файла, із вказівкою на його місце розташування й протокол виклику.

Розрізняють три типи фізичних імен файлів, пов'язаних між собою через відповідні інтерфейси (рис. 20).

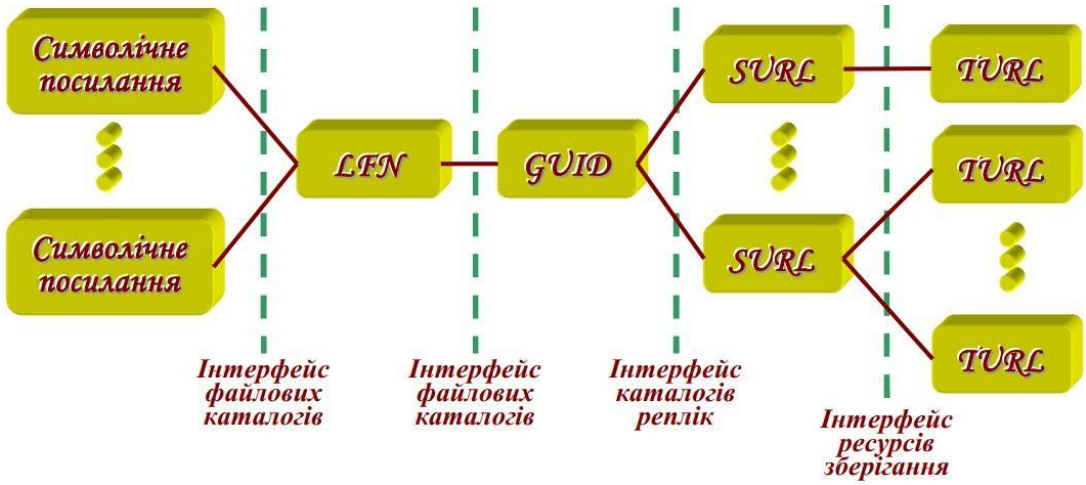

Рисунок 20 – Схема відображень різних просторів імен файлів у Грід-системах

Тут GUID (Grid Unique IDentifier) – глобальний унікальний ідентифікатор, з яким взаємно однозначно пов'язане кожне LFN. GUID незмінні, тобто вони не можуть змінюватись користувачем. Ці імена використовуються застосуваннями Грід як незмінні покажчики на файли. Якщо проводити аналогію з файловою системою Unix/Linux, GUID відповідає унікальному індексу файла (inode). Такий взаємооднозначний зв'язок LFN і GUID означає, що у віртуальній файловій системі Грід, на відміну від Unix/Linux, не дозволені жорсткі посилання, оскільки здійснення глобально розподіленої файлової си-

стеми із жорсткими посиланнями є важкою задачею. Відзначимо, що в той час як сукупність LFN має ієрархічну структуру, GUID не мають ніякої структури взагалі.

Другий тип фізичних імен – SURL (Site Universal Resource Locator; Site URL) обумовлює фізичне місце розташування файла або його репліки. Файл може мати багато реплік, таким чином, відображення між GUID і SURL – «один до багатьох». Кожна репліка файла має свій власний унікальний SURL. Як SURL застосовується повне ім'я, зрозуміле інтерфейсу диспетчера (SRM) елемента SE. Прикладом SURL може слугувати такий рядок:

```
srm://castorgrid.cern.ch:8443/srm/managerv1?SFN=/castor/cern.ch/file1.
```

Ім'я SRM неявно відповідає частині SURL, яка передує символам `?sfn` в цьому прикладі.

Нарешті TURL (Transport URL) – URL, який застосовується, для фактичної передачі файла, за допомогою будь-якого стандартного транспортного протоколу. Фізичне ім'я TURL починається з протоколу, який використовується для передачі або файла прямого доступу, або файла, доступ до якого здійснюється через деякий механізм уведення/виведення.

4.1.1 Ресурси зберігання даних

Прикладне завдання мусить могти звертатися до своїх даних, незалежно від фактичного місця розташування процесора, на якому воно виконується. При цьому необхідно забезпечити доступ до систем зберігання різного типу, які є в ресурсних центрах – від простих файлових систем до пристроїв масового зберігання даних. Щоб запобігти врахуванню особливостей окремих пристроїв зберігання даних, кожен ресурс зберігання, включений у Грід-інфраструктуру, повинен реалізовувати єдиний інтерфейс. У цей час найпоширенішим і перспективним стандартом для такого інтерфейсу є SRM, який надає більшість функціональних можливостей, необхідних для управління

даними в Грід.

Сукупність служб, необхідних для забезпечення доступу до файлів, які зберігаються в кожному з ресурсних центрів, й називається SE Грід-системи й складається з:

- власне пристрою зберігання з усіма відповідними апаратними засобами й драйверами;
- реалізації інтерфейсу SRM;
- служби передачі даних для низки транспортних протоколів;
- служби уведення/виведення файлів (I/O);
- модулів взаємодії з підсистемами реєстрації і безпеки.

Безпосередньо користувальницькі прикладні завдання під час виконання на робочому вузлі ресурсного центра використовують службу уведення/виведення, щоб звернутися до своїх даних на пристрої зберігання, а SRM і служби безпеки використовуються у фоновому режимі. Служба передачі файлів (File Transfer service, FTS) управляє потоками даних, які входять до SE. Вона аналогічна службі управління пакетними завданнями обчислювального ресурсу. Ресурс зберігання, на який передаються дані, завжди є локальним для даного екземпляра служби передачі, а ресурс зберігання, який є джерелом даних, може бути усередині або поза межами даного ресурсного центра. FTS функціонально доволі складна і може складатися з декількох компонентів. Наприклад, у ППЗ gLite служба передачі складається з наступних компонентів:

- служба розміщення файлів – зберігає всі записи про передачу даних у своїй базі даних і одержує нові запити, користувачі взаємодіють зі службою передачі винятково через клієнтів цієї служби;
- агент передачі – опитує базу даних служби розміщення про доступні передачі, для яких ресурс зберігання, керований цим агентом, є пунктом призначення.
- бібліотека передач файлів – забезпечує рівень управління поверх спеціалізованих Грід-протоколів передачі даних на великі відстані, вона може контролювати стан передачі і може скасовувати її.

Служба уведення/виведення як вхідні параметри приймає або LFN, або GUID, звіряється зі службою прав доступу до файлів на предмет дозволу користувачеві звертатися до файла, зіставляє GUID або LFN з SURL, який потім використовується, щоб одержати доступ до необхідного файла.

4.1.2 Каталоги у підсистемі управління даними

Створення глобальної файлової системи, типу AFS на основі відомих рішень для ресурсів, розподілених у масштабі однієї або декількох тісно зв'язаних організацій, у рамках глобального Грід практично нездійсниме (або, у всякому разі, накладає занадто жорсткі умови на провайдерів ресурсів). Замість цього використовуються глобальні каталоги, які дозволяють шукати файли і управляти ними і їхніми репліками в масштабах усього Грід, створюючи тим самим віртуальну файлову систему.

Концептуально розрізняють Каталог метаданих (Metadata Catalog, MC), Файловий каталог (File Catalog, FC), Каталог реплік (Replica Catalog, RC) і Об'єднаний Каталог (Combined Catalog, CC). Каталоги можуть існувати як у вигляді локалізованої, так і розподіленої версії.

MC забезпечує операції запису й одержання результатів запитів з метаданих. Метадані – це дані про дані, тобто вони повинні на щось посилатися, про що мають додаткову інформацію. У цьому випадку метадані посилаються на логічні імена файлів. Метадані сильно залежать від специфіки прикладної галузі. Тому загальний каталог повинен містити досить обмежений набір метаданих, але мати інтерфейси до MC конкретних ВО.

FC надає можливість провадити дії над логічними іменами файлів. Такими діями можуть бути створення директорій, перейменування файлів і створення символічних посилань.

RC забезпечує операції, пов'язані з дублюванням (реплікацією) файлів у Грід. Зокрема, він забезпечує перерахування, додавання й видалення копій (реплік) файлів на різних ресурсах зберігання даних, ідентифікованих за допомогою їх GUID/SURL.

Таким чином, інтерфейс RC дає доступ до відображення GUID/SURL, у той час як FC управляє тільки LFN, внутрішньо зберігаючи співвідношення з GUID.

CC забезпечує всі операції типу створення віртуальних (логічних) директорій, які мають потребу в синхронізованому доступі принаймні до двох каталогів, і створення й видалення нових файлів.

Каталоги повинні підтримувати масові операції з файлами (операції над сукупністю файлів) як частину свого інтерфейсу. Такі операції збільшують продуктивність і оптимізують взаємодію користувачів із Грід-сервісами.

Служби каталогів використовуються ресурсами зберігання даних, щоб одержувати відповідність GUID і LFN з SURL і перевіряти права доступу до файлів.

4.2 Підсистема інформаційного обслуговування і моніторингу Грід

Декілька осторонь від описаної системи збереження файлів користувачів стоїть система інформаційного обслуговування і моніторингу Грід. Хоча ця система і застосовує для свого функціонування ті ж самі ресурси збереження даних, але її дані групуються не в розподілену файлову систему, а в спеціальну розподілену БД. Таке розходження пов’язане з тим, що підсистема інформаційного обслуговування й моніторингу Грід призначена для вирішення задачі збору й управління даними про стан Грід, одержуючи інформацію від безлічі розподілених джерел – постачальників.

Раніш вже йшла мова про деякі елементи архітектури системи виявлення несправностей MDS (див. розділ 2.4.4 , стор. 60), але ця система застосовувалась лише у ранніх розробках. Зараз в нових Грід-системах використовується інша, хоча й у дечому схожа на неї архітектура – Grid Monitoring Architecture (GMA).

Через природу самих даних, які збираються в Грід, інформаційні системи загального призначення (бази даних і сервіси директорій) погано підхо-

дять для розподіленого моніторингу. Дані моніторингу про стан засобів і завдань системи мають обмежений і, як правило, короткий час життя (після чого вони стають недостовірними). Тому частота їхніх відновлень повинна бути досить високою, а звичайні БД оптимізуються на запити, а не на відновлення. В інформаційній системі моніторингу мусить забезпечуватися низька затримка даних при передачі від точки одержання до точки, де вони зберігаються. У свою чергу сторона, яка приймає дані, повинна підтримувати високу швидкість прийому, що зумовлюється частими відновленнями.

У Грід кількість доступної інформації про продуктивність елементів системи є дуже великою. До того ж пошук цієї інформації по всій розподіленій системі неминуче призводить до непередбачуваних затримок. Але ці потенційно великі затримки ні в якому разі не повинні впливати на ефективність запитів про інформацію. Бажано, щоб окремі пари постачальників/споживачів інформації мали можливість надавати «узгоджений спротив» вадам системи і щоб обсяг даних, які протікають через систему, контролювався точним і розподіленим способом, ґрунтуючись на поточних даних про локальне навантаження. Це призводить до того, що пошук в БД слід використовувати лише для пошуку відповідного джерела або приймача інформації, тоді як операції з більш передбачуваною затримкою слід використовувати для передачі фактичної інформації про функціонування ресурсів. Суть пропозиції з реалізації такої БД системи полягає в тім, щоб розділити збір даних і операції пошуку. Дані моніторингу мусять зберігаться розподілено – там же де й виробляються.

Це й покладено в основу Грід-сервісу GMA. Тут пропонується адресувати пошукові запити «реєстру метаданих», який виконує роль індексу розподіленого зберігання й дозволяє визначити джерело необхідних даних. Далі запит направляється за адресою місця зберігання й там вже провадиться більш вузький пошук.

4.2.1 Архітектура і реалізація підсистеми інформаційного обслуговування

Архітектура моніторингу системи моделює інформаційну інфраструктуру Грід як множину, яка складається з трьох типів компонентів (рис. 21):

– служба каталогів: підтримує публікацію і виявлення інформації;
– постачальник: робить доступними дані про функціонування ресурсу (джерело події функціонування);
– споживач: отримує дані про функціонування ресурсу (приймач події функціонування).

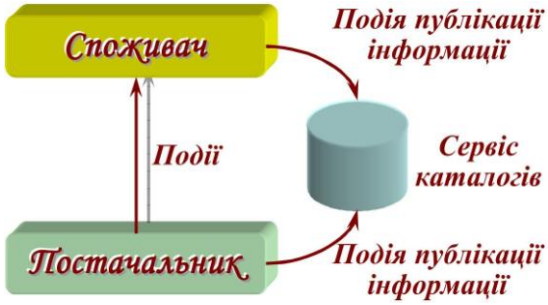

Рисунок 21 – Компоненти архітектури Грід-моніторингу

GMA призначена для обробки даних про функціонування ресурсів, які передаються як події з часовими позначками. Під подією розуміють типізовану колекцію даних з певною структурою, яка визначається схемою події. Дані про події функціонування завжди надсилаються безпосередньо від постачальника до споживача.

Постачальники (і споживачі, які приймають контрольні повідомлення) попередньо публікують дані про своє місцезнаходження в каталозі записів служби. Служба каталогів або реєстр використовується для пошуку постачальників і споживачів. Слід зауважити, що під терміном «служба каталогів» не розуміють ієрархічну службу, таку як LDAP, тут використовується будь-яка служба пошуку. Служба каталогів призначена для початкового завантаження даних про зв’язок між споживачами і постачальниками, тому що записи за-

повнюються інформацією про доступні мережні протоколи і механізми безпеки. Споживачі можуть скористатися службою каталогів для виявлення зацікавлених постачальників, а постачальники використовують службу каталогів для виявлення зацікавлених споживачів. В результаті або постачальник, або споживач ініціюють взаємодію з виявленим вузлом. У будь-якому випадку, передача даних про функціонування ресурсу відбувається безпосередньо між кожною парою споживач/постачальник без подальшого залучення служби каталогів.

GMA підтримує три типи взаємодії для передачі даних між постачальниками і споживачами: публікація/підписка, запит/відповідь і повідомлення.

Взаємодія GMA для публікації/підписки виконується в три етапи. На першому етапі – ініціатор взаємодії (це може бути або постачальник, або споживач) зв’язується з «сервером» (якщо ініціатором є споживач, сервер є постачальником, і навпаки), що вказує на зацікавленість у деякому наборі подій. Додатково на цій стадії також обговорюються параметри, необхідні для управління передачею даних. Сюди входять відомості про те, куди слід надсилати події функціонування, як кодувати або шифрувати події, а також як часто їх відправляти, розміри буфера, тайм-аути і т.ін. Це етап публікації. Початковий контакт і подальше спілкування на другому етапі здійснюється шляхом обміну контрольним повідомленням між ініціатором і сервером. Такі дії постачальника і споживача називаються підпискою. На цьому етапі взаємодії постачальник (який може бути ініціатором або сервером для взаємодії) відправляє одну або кілька подій виконання споживачу. Нарешті на завершальній стадії або постачальник, або споживач припиняє підписку, можливо з додатковими контрольними повідомленнями.

Для взаємодії запит/відповідь GMA ініціатор повинен бути споживачем. Взаємодія складається з двох етапів. Перший етап встановлює передачу, аналогічну першому етапу публікація/підписка. Потім замість відправлення однієї або декількох подій на виконання і відписки, постачальник передає всі події виконання споживачеві в єдиний відповіді. Ця взаємодія особливо добре відображає протоколи запиту/відповіді, такі як HTTP.

Взаємодія з повідомленнями GMA є одностадійною взаємодією, і ініціатор повинен бути постачальником. У цьому типі взаємодії постачальника/споживача постачальник передає всі події виконання споживачеві в одному повідомленні.

Служба каталогів GMA зберігає інформацію про постачальників і споживачів, які приймають запити. Коли постачальники та споживачі публікують в службі каталогів дані про свою наявність в системі, то, як правило, виконується й публікація інформації щодо типів подій, які ними виробляються або їм потрібні. Разом з цим публікуються метадані про прийняті в системі протоколи, механізми безпеки і таке інше. Публікація такої інформації, або реєстрація, дозволяє іншим постачальникам і споживачам виявляти доступні в даний час або прийняті в Грід типи даних подій, характеристики цих даних і способи отримання доступу до них. Служба каталогів не несе відповідальності за зберігання самих даних події – вона містить тільки публікації про те, які екземпляри подій можуть бути надані або прийняті. Схема події необов’язково доступна через службу каталогів.

Служба підтримує чотири функції:

– «Додавання»: запис додається до каталогу;
– «Оновлення»: запис у каталозі змінюється;
– «Видалення»: запис видаляється з каталогу.
– «Пошук»: виконується пошук постачальника або споживача на основі деяких критеріїв вибору.

Клієнт повинен вказувати, скільки співпадаючих подій слід йому повернути, одну або декілька, якщо вони доступні. Додаткове розширення дозволяє клієнту отримати декілька збігів, одночасно використовуючи запит «отримати наступний» у наступних пошуках.

GMA не визначає протоколи для каналів управління і для даних подій. Мережний протокол, який використовується для передачі інформації про управління між постачальником і споживачем, і мережний протокол, який використовується для передачі подій функціонування, можуть різнитись. Розробники Грід можуть підтримувати один або більше мережних протоколів,

наприклад, SOAP/HTTP, LDAP або XML/BXXP, вибираючи ті, які найкраще підходять для даної системи. GMA також не надає гарантії доставки подій.

Постачальником в GMA є будь-який компонент, який використовує інтерфейс постачальника для передачі подій споживачеві. Даний компонент може мати кілька інтерфейсів постачальника, кожен з яких діє і відправляє події незалежно.

Основні функції взаємодії, які зазвичай підтримуються постачальником такі:

– «Ведення реєстрації»: додавання/оновлення/видалення запису служби каталогів або опису події, які постачальник відправляє споживачеві, дія відповідає функціям службі каталогів «Додати», «Оновити» і «Видалити»;
– «Прийняти запит»: підтвердження запиту від споживача, у відповідь на запит споживачу надсилається одна або більше подій, відповідає функції споживача – «Ініціювати запит»;
– «Прийняти підписку»: прийняти запит на підписку від споживача, деталі про підписку повертаються у відповіді, при успішному встановленні підписки, постачальник посилає споживачеві події до припинення підписки, для споживача відповідає – «Ініціювати підписку»;
– «Підтвердити скасування підписки»: прийняти запит на відписку від споживача, при успішності операції відповідна підписка закривається і припиняється відправлення подій передплатнику, з боку споживача відповідає «Ініціювати скасування підписки»;
– «Пошук споживача»: пошук у службі каталогів на предмет наявності відповідного споживача, з боку директорії означає «Пошук служби»;
– «Оповістити»: надсилати споживачеві один набір подій, відповідає «Прийняти повідомлення» від споживача;
– «Ініціювати підписку»: запит до споживача на відправлення подій, деталі про підписку повертаються у відповіді, якщо підписка успішно встановлена, то постачальник посилає події споживачеві до при-

пинення підписки, з боку споживача є відповідна функція – «Прийняти підписку»;

– «Ініціювати скасування підписки»: припинення підписки зі споживачем, якщо це вдається, то підписка закривається, і події для цієї підписки більше не відправляються, відповідає функції «Припинення підписки» з боку споживача.

Постачальники можуть забезпечувати контроль доступу до даних події, дозволяючи різний доступ для різних класів користувачів. Оскільки ВО об’єднують декілька реальних організацій, які контролюють наявність ресурсів моніторингу, то можуть існувати різні політики доступу (можливо, брандмауери), різні частоти вимірювання, а також різні характеристики функціонування для споживачів «всередині» або «поза» організації, яка управляє ресурсом.

На додаток до основних функціональних можливостей постачальники GMA можуть надавати й інші послуги, наприклад, фільтрування подій, кешування, проміжну обробку даних на запит споживача і багато чого іншого.

Інформація про послуги, які підтримуються даним постачальником, публікується в каталозі на рівні з інформацією про подію.

Споживачем є будь-який компонент, який використовує інтерфейс споживача для отримання даних про подію від постачальника. Компонент може мати декілька інтерфейсів споживача, кожен з яких діє і отримує події незалежно.

Споживачі підтримують низку основних функцій взаємодії, які подібні основним функціям постачальників:

– «Пошук постачальника»: пошук по службі каталогів на предмет наявності в системі хоча б одного відповідного постачальника, з боку директорії – це функція «Пошук служби»;

– «Ініціювати запит»: запит на одну або більшу кількість подій від постачальника, які доставляються як частина відповіді, відповідає функції «Прийняти запит» з боку постачальника;

– «Ініціювати підписку»: запит на встановлення підписки з постачаль-

ником, подробиці про деталі підписки повертаються у відповіді постачальника, якщо підписка пройшла успішно, постачальник посилає події до моменту припинення підписки, функція відповідає запиту «Прийняти підписку» з боку постачальника;

– «Ініціювати скасування підписки»: припинення підписки, при успішності операції відповідна підписка закривається і подій для цієї підписки більше не приймається, з боку постачальника відповідає «Підтвердити скасування підписки».
– «Ведення реєстрації»: додавання/оновлення/видалення запису служби каталогів або опису події, які споживач прийме від постачальника, відповідає функціям службі каталогів «Додати», «Оновити» і «Видалити»;
– «Прийняти повідомлення»: прийняти один набір подій від постачальника, відповідає функції постачальника «Оповістити»;
– «Прийняти підписку»: прийняти запит на підписку від постачальника, подробиці про підписку повертаються у відповіді постачальника, якщо підписка успішно встановлена, то до моменту припинення підписки постачальник посилає події передплатнику, з боку постачальнику є відповідна функція – «Ініціювати підписку»;
– «Припинення підписки»: підтвердити запит про скасування підписки від постачальника, якщо це вдається, підписка закривається, і ніякі подій для цієї підписки більш не приймаються;
– «Знайти схему подій»: запит до сховища схем на предмет пошуку даного типу події, таке сховище схем може бути GMA-службою безпосередньо каталогів споживачів.

Наслідком поділу пошуку даних від їхньої передачі є те, що протоколи, які використовуються споживачем для виконання публікації/підписки, запиту/відповіді і опису взаємодії можна використовувати ще й для побудови посередників, які, наприклад, перенаправляють події, транслюють їх, фільтрують, кешують тощо. Комбінований постачальник/споживач (рис. 22) править за будівельний елемент для подібних якісніших послуг. Цей ресурс є єдиним

компонентом, він одночасно реалізує і інтерфейси постачальника і споживача, і змінені події. Інтерфейс споживача такого постачальника/споживача може збирати дані про події від декількох постачальників, використовувати ці дані для створення нових типів даних похідних подій і доставляти їх іншим споживачам через інтерфейс постачальника.

Рисунок 22 – Приклад комбінованого постачальника/споживача

Велика кількість Грід-сервісів правлять і за споживачів, і за постачальників моніторингу подій. Наприклад, архіви подій, скоріш за все, дійсно реалізують інтерфейси як постачальника, так і споживча. Використання подібних проміжних компонентів може значно зменшити навантаження на постачальників подій, що важливо для багатьох споживачів. До того ж це призводить до подальшого скорочення мережевого трафіка, тому що посередники розташовуються «поблизу» від споживачів даних.

Оскільки GMA є досить загальною, вона застосовна як для зберігання даних про Грід (які ресурси й сервіси доступні, які в них характеристики), так і для моніторингу виконання завдань.

Дані, які використовуються для побудови подій, можна зібрати з багатьох джерел. Устаткування або програмне забезпечення датчиків, які вимірюють показники функціонування в реальному часі, є одним з типів джерел даних. Іншим типом є база даних з інтерфейсом запитів, яка може надавати

архівовані дані. Цілі системи моніторингу такі, як мережева служба погоди може правити за джерело подій. Крім того, таймінги від таких інструментів, як Autopilot або NetLogger можуть забезпечити події, пов'язані з конкретним застосуванням.

Постачальник може пов'язуватись з одним джерелом, з усіма джерелами на даному хості, з усіма джерелами на даній підмережі, або з цілком довільною групою джерел. На рис. 23 показана одна з можливих конфігурацій, але архітектура дозволяє розробникам Грід-системи вибирати найкращу конфігурацію, яка відповідає їхнім потребам масштабованості та надійності системи. GMA не визначає ні відносини, ні інтерфейс між джерелами даних вимірювання і постачальником GMA.

4.3 Реляційна реалізація Grid Monitoring Architecture

В останні часи однією з найпоширеніших, принаймні в Європі, систем Грід-моніторингу є похідна від GMA система R-GMA.

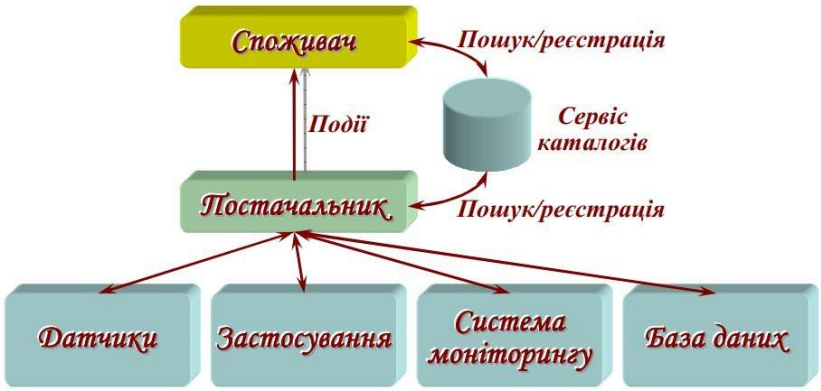

Рисунок 23 – Джерела даних подій

R-GMA – це реляційна реалізація GMA, яка розроблена в межах європейського Грід-проекту – DataGrid (EDG), якому міць і гнучкість надає реляційна модель. R-GMA створює враження, що в кожній ВО є СУБД. Однак слід відзначити, що те, що забезпечує ця система, є лише способом використ-

тання реляційної моделі в Грід-середовищі, а не розподіленою СУБД у загальному розумінні тому, що всі виробники інформації досить незалежні. Реляційність розуміється в тому значенні, що постачальники за допомогою SQL-запиту CREATE TABLE повідомляють про те, що вони мають опубліковувати і опубліковують це за допомогою SQL-запиту INSERT, а споживачі, в свою чергу, використовують SQL-запит SELECT для збору інформації про необхідні їм ресурси.

R-GMA побудована з використанням технології сервлетів і, елементарно адаптуючись до веб-сервісів, легко вбудовується в структуру OGSA.

R-GMA визначає п’ять різних типів постачальників: DataBaseProducer, StreamProducer, ResilientProducer, LatestProducer і CanonicalProducer. Всіх їх споживач бачить як звичайних постачальників, але вони мають різні характеристики. CanonicalProducer – у деяких відносинах найуніверсальніший, але характеризується тим, що не має користувальницького інтерфейсу для публікації даних за допомогою SQL-оператора INSERT. Замість цього він запускає цілий користувальницький код для відповіді на SQL-запит. Всі інші постачальники – «публікатори» – це означає, що всі вони мають інтерфейс для використання SQL-оператора INSERT.

Ці постачальники створюються як екземпляри з урахуванням опису інформації, яку вони пропонують за допомогою SQL-оператора CREATE TABLE і речення WHERE, яке містить предикат істинний для таблиці. Це речення має вигляд:

```
WHERE (<column_1>=<value_1> AND <column_2>=<value_2> AND ...).
```

Для публікації даних викликається метод, який приймає форму звичайної SQL-інструкції INSERT. Підтримуються три типи запитів: History, Latest і Continuous. Ще можна відмітити такий тип запитів, як Static, але ці запити типові для звичайних баз даних, не містять часових позначок і тому практично не розглядаються як елемент структури R-GMA. Запит History може розглядатися як найбільш традиційний, коли потрібно зробити запит про дані

протягом деякого відрізку часу – включаючи «увесь час». Запит Latest використовується для пошуку поточного значення яких-небудь даних і запит Continuous надає клієнтові всі співпадаючі для його запиту результати так, як вони опубліковані. Таким чином, запит Continuous діє як фільтр потоку даних.

DataBaseProducer підтримує запити типу History. Він зберігає в СУБД кожен запис. Це доволі повільно (у порівнянні з StreamProducer), але цей постачальник управляє з'єднаннями. StreamProducer підтримує безперервні запити і записує інформацію в структури пам'яті, звідки споживач може їх одержати. ResilientStreamProducer схожий на StreamProducer, але створюється резервна копія інформації, яка зберігається на диску, щоб інформація не втрачалася у випадку збою системи. LatestProducer підтримує тільки недавні запити, зберігаючи тільки їхні записи в СУРБД.

Кожен запис має позначку часу, одне або кілька полів, які визначають те, що виміряється (наприклад, ім'я хосту) і одне або кілька полів, які надаються ресурсом, наприклад, середнє завантаження процесора за 1 хвилину. Позначка часу і визначальні поля за смислом близькі до первинного ключа, але тому, що немає ніякого способу довідатися, хто й що публікує у Грід, то поняття первинного ключа (як глобально унікального) не має жодного сенсу. LatestProducer мусить замінювати вже наявний запис на запис, який щойно надійшов і має таке саме значення поля. Заміна виконується щораз, коли позначка часу на новому записі така ж, як на старому або новіша.

Ще постачальників можна класифікувати за ознакою того, як вони формують інформацію (кортежі для таблиць). За цією ознакою постачальники дуже схожі на первинні і вторинні сервіси GIIS в системі MDS і відрізняють три класи постачальників:

– первинні;
– вторинні;
– постачальники на вимогу.

Основна різниця між ними полягає в механізмі формування кортежів. Первинні постачальники самі формують інформацію, зберігають її у своїй пам'яті й самі відповідають на запити споживачів. Вторинні постачальники

збирають інформацію від первинних, зберігають її й також самі відповідають на запити. У постачальників на вимогу немає внутрішньої пам’яті – дані формуються безпосередньо у відповідь на запит.

Постачальнику, особливо такому, який використовує СУБД, час від часу може знадобитися очистка. Для цього є механізм, який дозволяє за допомогою зазначеного користувачем SQL-речення WHERE, знайти й видалити непотрібні записи з таблиці. Механізм запускається й виконується з інтервалами часу, які також вказуються користувачем. Наприклад, з якоїсь таблиці так можна видалити записи зроблені більше тижня тому, або таблиця може містити лише останні сто кортежів, або просто зберігати один запис на добу. Загалом кажучи, точне значення понять «старий кортеж», і «поточний стан» (для запитів типу Latest) обумовлюється за допомогою двох типів періодів зберігання кортежів. LatestRetentionPeriod (період зберігання останніх кортежів) обумовлює – після якого часу зберігання кортеж не може розглядатися як Latest. Первинний постачальник додає цей інтервал часу до часової позначки й вставляє його в кожен кортеж, який публікується. Значення позначки залишається в кортежі до моменту, коли вторинний постачальник заново опублікує кортеж. HistoryRetentionPeriod (період зберігання історії) – оголошується первинними й вторинними постачальниками для кожної таблиці, у якій вони публікують кортежі. Таким чином, первинні й вторинні постачальники мають два логічних набори кортежів: один для запитів типу, а інший – для запитів і. Постачальники зобов’язані зберігати нову версію будь-якого кортежу, який не перевищив LatestRetentionPeriod, і всі версії будь-яких кортежів, які не перевищили HistoryRetentionPeriod. Всі первинні й вторинні постачальники підтримують запити Continuous, однак не всі постачальники обов’язково підтримують History й Latest запити. HistoryRetentionPeriod може бути довше або коротше, ніж LatestRetentionPeriod. HistoryRetentionPeriod – це властивість постачальників, тоді як LatestRetentionPeriod – властивість кортежу.

Ще одним важливим компонентом системи є архіватор, структура якого – це об’єднаний постачальник/споживач. Користувачеві лише потрібно

повідомити архіватор, що збирати, й він зробить це від імені користувача. Архіватор працює, дістаючи контроль над існуючим постачальником і створюючи споживача для кожної таблиці, яку потрібно архівувати. Потім цей споживач підключається до всіх придатних постачальників і через архіватор починає потокову передачу даних від них новому постачальнику. Джерелом вхідних даних для архіватора завжди правлять потоки від StreamProducer або ResilientStreamProducer.

В системі для кожної ВО створюється реєстр, у якому зберігається інформація про всіх доступних постачальників. Спеціальний компонент системи – посередник, який криється за інтерфейсом споживача, за допомогою стандартного SQL-запиту:

```
SELECT * FROM <table> WHERE <predicate>
```

і використовуючи реєстр від імені користувача, може переглянути віртуальну БД, знайти необхідних постачальників і об’єднати інформацію від них.

R-GMA заснована на технології сервлетів. Основна частина кожного компонента реалізується в сервлеті. Для зв’язку з сервлетами доступна безліч API мовами Java, C++, C, Python і Perl. Основні з цих цілком написані на Java і C++. API на C звертається до C++, а API на Python і Perl генеруються за посередництвом SWIG. Більшість кодів дуже компактна і написана мовою Java з використанням контейнера сервлетів Tomcat. API сервлетів не підтримують безпеку їхньої роботи, ця функція покладається на інші компоненти програмного забезпечення EDG і виконується завдяки сервісам безпеки Грід.

Рис. 24 демонструє зв’язки між API і сервлетами. При створенні джерела його реєстраційні дані через сервлет постачальника відправляються до реєстру (рис. 24 а). Реєстр записує подробиці про виробника, які містять опис і перегляд опублікованих даних, але не самі дані. Опис даних насправді зберігається у вигляді посилання на таблицю в схемі. На практиці схема пов’язана з реєстром. Потім, коли постачальник публікує дані, вони передаються сервлету локального постачальника (рис. 24 б).

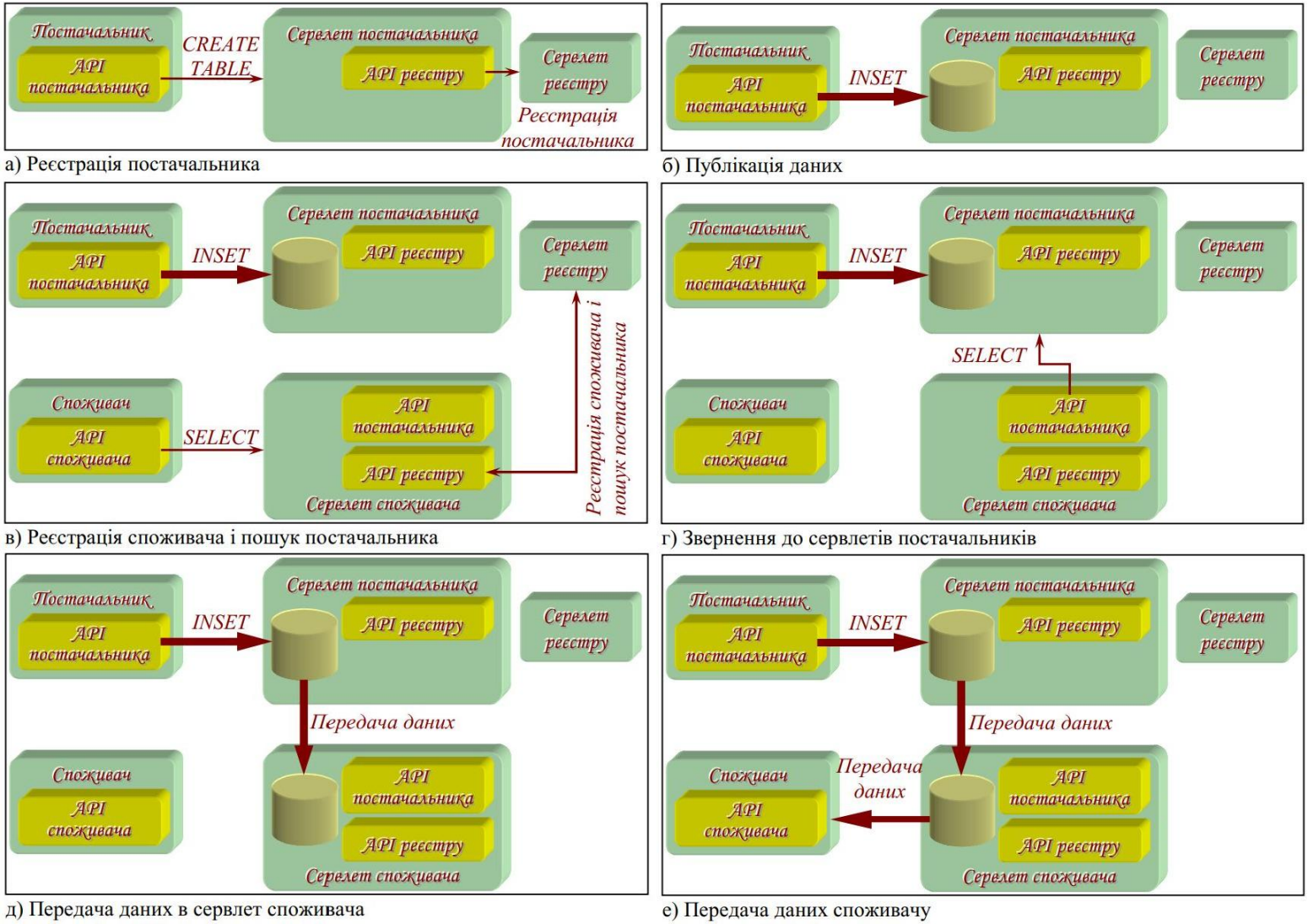

Рисунок 24 – Спілкування між API і сервлетами в реляційній архітектурі моніторингу Грід

При створенні споживача його реєстраційні дані також відправляються до реєстру, але вже через сервлет споживача (рис. 24 в). В реєстрі записуються подробиці про тип даних, які цікавлять споживача. Після чого реєстр назад в сервлет споживача повертає список постачальників, які відповідають критеріям вибору.

Одержавши список, сервлет споживача звертається до сервлетів відповідних постачальників, щоб ініціювати передачу даних із сервлетів постачальників у сервлет споживача, як показано на рис. 24 г), 24 д).

Нарешті ці дані стають доступними споживачеві на сервлеті споживача, який, з погляду мережі, найближчий до споживача (рис. 24 е).

Оскільки відомості про споживачів і критерії їхнього вибору зберігаються в реєстрі, сервлети споживачів автоматично повідомляються при реєстрації нових постачальників, які відповідають їхнім критеріям вибору.

Для надійності система використовує «м'яку» реєстрацію стану. Як виробники, так і споживачі зобов'язуються зв'язуватися зі своїм сервлетом через певні проміжки часу. Відповідні помітки часу зберігаються в реєстрі, і якщо до цього часу не відбулося якого-небудь звернення від клієнта, виробник або споживач реєстрацію не виконує. Сервлети постачальників і споживачів постійно відслідковують час останнього одержання події від свого клієнта й забезпечують своєчасне відновлення помітки часу в реєстрі.

Для роботи з R-GMA користувач може використовувати або інтерфейс командного рядка, або графічний інтерфейс спеціального браузера. Ці інтерфейси, зокрема, дозволяють:

- переглядати схему даних в R-GMA, тим самим визначати доступні таблиці;
- переглядати списки постачальників, які публікують дані в таблицях;
- спрямовувати запити до таблиць за допомогою посередників;
- направляти запити до певних постачальників таблиць.

Щоб використовувати браузер R-GMA, коли він працює на безпечному HTTPS сервері, користувач мусить використовувати спеціальний електронний сертифікат, який імпортується у веб-браузер.

На завершення цього розділу в табл. 5 наведена відмінність призначення і функціонування елементів архітектур MDS і R-GMA

Таблиця 5 – Порівняння підсистем інформаційного обслуговування і моніторингу

Архітектура / Елемент	MDS	R-GMA
Збирач інформації	Постачальник інформації (IP)	Постачальник
Інформаційний сервер	GRIS	Сервлет постачальника
Сукупний інформаційний сервер	GIIS	(Немає)
Сервер каталогів	GIIS	Реєстр

4.4 Контрольні запитання і завдання для самостійної роботи

1. З яких сервісів складається підсистема управління даними і для чого вона призначена?
2. Надайте визначення логічного імені файлів. Охарактеризуйте простір логічних імен у підсистемі управління даними.
3. Згадайте, які три типи фізичних імен файлів існують в системі. Докладно розкажіть про інтерфейси, якими вони пов’язані між собою і з логічними іменами.
4. Що називається елементом збереження даних Грід-системи і з чого він складається?
5. Які типи каталогів розрізняють в глобальному Грід, у якому вигляді вони існують? Докладно розкажіть про призначення всіх чотирьох типів каталогів.
6. Що відрізняє систему інформаційного обслуговування і моніторингу Грід від системи збереження файлів користувачів?

7. Що і чому в сучасному ППЗ покладено в основу Грід-сервісу інформаційного обслуговування й моніторингу?
8. Що моделює і для чого призначена сучасна архітектура моніторингу стану системи? Хто такі постачальники і споживачі, що таке подія?
9. Докладно розкажіть про три типи взаємодії для передачі даних між постачальниками і споживачами.
10. Які основні функції взаємодії зазвичай підтримуються постачальником? Надайте розгорнуту характеристику цих функцій.
11. Яку низку основних функцій взаємодії підтримують споживачі? Поясніть застосування цих функцій.
12. Навіщо потрібні посередники, як вони створюються і як функціонують?
13. В чому полягає перевага реляційної реалізації архітектури моніторингу Грід-систем? Чому така архітектура не є розподіленою СУБД у загальному розумінні?
14. Охарактеризуйте всі п’ять різних типів постачальників, які визначаються реляційною архітектурою моніторингу.
15. На якій технології базується реляційна архітектура моніторингу Грід-систем? Як на основі цієї технології взаємодіють постачальник і споживач?

5 БЕЗПЕКА ФАЙЛОВОЇ СИСТЕМИ. СЕРТИФІКАТ ВІДКРИТИХ КЛЮЧІВ

Грід-системи за своєю суттю є відкритими системами оскільки створюються з урахуванням вимог стандарту OGSA. Основний принцип відкритих систем полягає у тім, що створюється середовище для додатків, яке складається з програмного забезпечення, апаратних засобів, служб зв’язку, інтерфейсів, форматів даних і протоколів, що у сукупності забезпечує переносимість, взаємозв’язок і масштабованість. Невід’ємною частиною відкритих систем є профіль – сукупність стандартів і їхніх вимог. Відкритість систем досягається лише за рахунок використання загальновизнаних загальнодоступних стандартів.

З огляду на значне і незмінне зростання обчислювальних технологій і систем, уразливість інформації в таких системах значно підвищується. Це сильно непокоїть як користувачів, так і виробників програмно-апаратних засобів.

5.1 Підсистема безпеки файлової системи Грід

У загальному випадку, під безпекою інформації розуміють умови зберігання, обробки і передачі інформації, при яких забезпечується її захист від погроз знищення, зміни і розкрадання. Окремим випадком зміни інформації є порушення її цілісності.

У будь-якій платформі, на якій будується відкрита система, можна виділити 6 зв’язаних між собою служб:

– служби операційної системи (функціональна галузь програмування);
– служби інтерфейсу «людина-машина» (користувальницький інтерфейс);
– служба підтримки й управління даними;
– служба обміну даними;
– служба машинної графіки;
– служба мережного забезпечення.

Є сенс розглянути всі шість типів служб з погляду безпеки інформації і можливих механізмів її підвищення.

У функціональній галузі ПЗ повинна підтримуватися модульність. При цьому кожен модуль повинен проходити ідентифікацію, автентифікацію і перевірку на цілісність. Це дозволяє забезпечувати безпечну зміну або додавання нових модулів. Принцип захисту інформації тут може бути аналогічним системі сертифікатів X.509. Ці сертифікати дозволяють використовувати усередині будь-який алгоритм електронного цифрового підпису. За допомогою цих сертифікатів можна легко модифікувати роботу всієї системи шляхом заміни і модифікації окремих модулів. Головне, щоб система зв’язку між модулями була відкритою і здатною до використання сторонніми виробниками.

В сфері користувальницького інтерфейсу людина-машина дуже важливо, щоб процес захисту інформації якнайменше заважав користувачеві. Якщо

захист буде заважати, то користувач завжди знайде спосіб його обійти. Проте, користувача і у цій сфері потрібно ідентифікувати й автентифікувати. На сьогоднішній момент існує дуже багато варіантів захисту інформації в користувальницькому інтерфейсі, але загальновизнаних стандартів немає. Найчастіше необхідно просто на початку роботи ввести ім’я й пароль.

Дуже важливо, щоб постійно розроблялися нові й удосконалювалися старі стандарти на інтерфейси пристроїв автентифікації (таких як дактилоскопічних або пристроїв-зчитувачів ключів)

З погляду підтримки і управління даними, мати можливість обробки блоків даних повинні програмні комплекси і модулі будь-яких виробників. Тут безумовно, важлива конфіденційність даних і/або вірогідність даних після операцій з ними. Ця задача вирішується шляхом впровадження електронного цифрового підпису для досягнення вірогідності даних або шифрування цілих блоків даних для забезпечення конфіденційності. Але, як і раніше тут повинні використовуватися єдині прийняті суспільством відкриті стандарти.

Обмін даними – одна з найважливіших служб. Завдяки їй різні користувачі і програмні модулі можуть обмінюватися потрібною інформацією. Для коректної роботи в розрізі ІБ ця служба повинна складатися з модулів, кожен з яких має коректний сертифікат. Це відноситься не тільки до модуля шифрування, але й до проміжних модулів, які можуть знадобитися. Наприклад, до модуля, який просто підготовляє інформацію до шифрування перед обміном. Іншими словами – немає малозначущих етапів, а захищеність всієї системи завжди дорівнює захищеності найслабшої ланки системи.

Машинна графіка – мабуть єдина служба в розглянутій моделі платформи для відкритих систем, яка не має потреби в захисті інформації, тому що повинна передавати сигнал користувачеві винятково у відкритому вигляді. Іноді, щоправда, і в машинній графіці потрібна безпека. Але це буває вкрай рідко й відноситься скоріше до пристроїв відображення інформації, аніж до програмного комплексу.

Гостро стоїть проблема безпеки з’єднання між віддаленими комп’ютерами в Грід-системі. В цей час вони працюють у мережі Інтернет, а вона по-

будована на стеку протоколів TCP/IP. Але при проектуванні цих протоколів, зовсім не було приділено уваги безпеці. Інакше кажучи, на рівні TCP/IP безпеки немає взагалі. Для реалізації безпечного зв'язку необхідно використовувати деяку надбудову на стек TCP/IP. Саме так і роблять сьогодні, тому що безпека має все більшу значущість. Шифрування трафіка відбувається часто-густо. Навіть незважаючи на те, що це помітно збільшує трафік. Часто використовується протокол SSL. Хоч SSL і прийнятий консорціумом W3C, багато хто вважає цей протокол слабким і недостатньо захищеним. Це дійсно так, тому більшість розробників віддає перевагу протоколам похідним від SSL або подібним до нього. І знову ж спостерігається нагальна потреба розробляти нові стандарти і створювати нові профілі захисту інформації для забезпечення задовільного рівня ІБ.

Зі сказаного вище, стає очевидним, що безпека в Грід-системах – це стан захищеності інформаційного середовища систем, який використовується для Грід-обчислень, а також комплекс заходів, спрямованих на забезпечення цього стану.

Згадуваний раніше, один з лідерів у сфері розробки ППЗ Грід для представників великого бізнесу – асоціація Enterprise Grid Alliance, починаючи з 2004 року, постійно публікує матеріали, у яких викладаються вимоги до безпеки Грід-інфраструктур.

Деякі вимоги до безпеки, описувані в документах, цілком застосовні й до традиційних архітектур. Просто при проектуванні Грід-інфраструктур їхнє дотримання обіцяє набагато більшу вигоду. Приміром, у системі зберігання може перебувати важлива інформація, доступ до якої необхідно надати лише одному застосуванню, у той час як інші програми теж звертаються до цього ж самого ресурсу. Розподілені обчислення за своєю природою передбачають створення умов, при яких кілька застосувань підключаються до одного ресурсу зберігання. Таким чином, питанням безпеки тут необхідно приділяти дуже серйозну увагу.

Інші вимоги відносяться винятково до Грід-середовищ, більшість із них відноситься до класу «одиниць управління Грід» (Grid Management Entity,

GME), які відповідають за операції в розподілених середовищах. GME налаштовують конфігурацію й підтримують роботу компонентів Грід-середовищ, таких як сервери й дискові масиви, управляють робочим навантаженням і «переводять у резерв» компоненти, які виконали свою роботу.

У документах так само описуються різні ризики і їхній вплив на Грід-середовища. До таких ризиків відносяться різні атаки, спрямовані на одержання контролю над доступом. У ході цих атак неавторизовані користувачі або компоненти підключаються до Грід-мережі. Атаки, націлені на відмову в обслуговуванні, які паралізують засоби управління мережею. Нарешті, повторне використання об'єктів, при якому неавторизований користувач одержує доступ до компонента Грід-середовища, яке належним чином не переведене в автономний стан і не «ізольоване» від мережі.

За великим рахунком засоби безпеки Грід повинні підтримувати добрий десяток засобів захисту інформації.

Насамперед, мова йде про автентифікацію – надання способу підключення різних механізмів автентифікації й методів їхнього використання в різних ситуаціях.

Потім система безпеки повинна забезпечувати передачу прав – надання засобів, які дозволяють здійснювати передачу прав доступу від запитуючої сторони до викликуваної служби.

Наступне – це єдиноразовий вхід, під цим розуміється звільнення суб'єктів, які вже виконали процедуру автентифікації, від необхідності її повторення при кожній спробі доступу до ресурсів на якийсь час.

Важливим є забезпечення життєвого циклу мандатів і його відновлення тому, що у багатьох випадках можлива ситуація, коли процес, ініційований суб'єктом, виконується довше, ніж час дії виданого мандата. Тому необхідно попереджати про це суб'єкта або передбачити відновлення мандата, для того, щоб робота могла бути закінчена.

Обов'язково повинна підтримуватися авторизація – дозвіл доступу до служб на підставі політик авторизації, пов'язаних із службами (хто і на яких підставах може здійснювати доступ), і надання можливості визивній стороні

задавати політики виконання (кому клієнт довіряє виконання).

Дуже важливо забезпечувати конфіденційність, тобто запобігання витоку (розголошення) якої-небудь інформації.

Система безпеки відповідає за цілісність даних – забезпечення виявлення несанкціонованих змін.

Обмін політиками, тобто надання визивній і викликуваній сторонам можливості з обміну інформацією про політики безпеки для створення безпечного середовища обміну інформацією, так само є однією з обов'язкових вимог до системи безпеки.

Система безпеки повинна реалізовувати засоби, що дозволяють визначати й підтримувати необхідний рівень забезпечення безпеки всієї системи в цілому.

Нарешті повинна забезпечуватися безпечна проникність мережних екранів (firewalls), оскільки вони є основним бар'єром при передачі даних у динамічних, кросдоменних Грід-системах, тому при проектуванні системи необхідно забезпечувати можливість вільної передачі даних через екран без зміни їхніх політик безпеки.

Більшість із перерахованих вище вимог увійшли в стандарт OGSA, а точніше кажучи у його розділ, який визначає інфраструктуру безпеки GSI. У свою чергу, GSI ґрунтується на надійній і широко використовуваній технології відкритих криптографічних ключів (Public Key Infrastructure, PKI).

На кожному рівні архітектури OGSA Грід рішення забезпечення безпеки мають свою специфіку. При цьому варто враховувати той факт, що повна система безпеки Грід повинна успішно взаємодіяти із уже існуючими локальними рішеннями. Однак, різнорідність локальних рішень істотно ускладнює створення комплексних систем безпеки. Тому мабуть єдиним можливим шляхом усунення труднощів такого роду є використання тих самих стандартизованих підходів при реалізації локальних рішень забезпечення безпеки, що і при реалізації Грід.

Найчастіше при розробці Грід-систем рекомендується диференціація вимог ІБ на базові (доступність, цілісність, конфіденційність) і розвинені

(єдиний вхід, делегування повноважень, динамічне встановлення відносин довіри й т.п.).

Виконати ці вимоги можна з використанням моделі, яка має п’ять рівнів, а саме:

– локального захисту;
– трансляції ідентифікаторів суб’єктів;
– комунікаційної безпеки;
– функцій безпеки;
– застосувань безпеки.

На рівні локального захисту застосовуються механізми безпеки, наявні в ланці управління. Ці механізми діють не тільки для локальних, але й для віддалених (з інших ланок управління) суб’єктів. Загалом кажучи, у різних ланках управління механізми безпеки можуть бути різними.

Наступний рівень надає функції для відображення (трансляції) ідентифікаторів суб’єктів з однієї ланки управління, наприклад, віддаленої, на ідентифікатори з іншої ланки – локальної. Після цього до віддалених суб’єктів можуть застосовуватися локальні політики й механізми безпеки.

Комунікаційна безпека, з виконанням всіх базових вимог, забезпечується на транспортному рівні стеку TCP/IP, а також на рівні окремих повідомлень. Для реалізації використовуються специфікації WS-Security.

Рівень функцій безпеки з використанням нижчерозташованих рівнів забезпечує виконання розвинутих вимог безпеки. Для реалізації корисні такі специфікації, як WS-Policy, WS-Trust, WS-Federation і т.ін.

На рівні застосувань безпеки функції попереднього рівня стають доступними у вигляді сервісів безпеки й можуть використовуватися як з локальної, так і з віддалених ланок управління. Можливе створення ієрархії сервісів безпеки.

Така багаторівнева архітектура безпеки відповідає духу сервісно-орієнтованої архітектури, знижує складність систем, приховує деталі локальних реалізацій, забезпечує взаємну сумісність і інтеграцію засобів ІБ для Грід-конфігурацій і сприяє підтримці діючих і розвитку перспективних стан-

дартів.

Інтеграція інфраструктури безпеки з архітектурою відкритих Грід-сервісів дає можливість використовувати методи веб-сервісів для визначення і публікації політик, дозволяючи застосуванням автоматично визначати, які політики й механізми безпеки їм потрібні.

Розробники різних Грід-середовищ, як типове ППЗ часто використовують вільно розповсюджуваний пакет з відкритим вихідним кодом для Грід-інфраструктур – Advanced Resource Connector (ARC) колаборації NorduGrid. На рис. 25 відображено структуру апаратного й програмного забезпечення і їхню взаємодію на деякому віддаленому обчислювальному вузлі. Після проходження процедури автентифікації, розподілений програмний продукт (РПП) користувача Грід-системи запускається від імені користувача Грід на ідентифікованому кластері. Звичайно РПП для організації обміну повідомленнями в розподіленому середовищі задіє MPI, а конкретний екземпляр РПП виконується в ОС конкретного вузла.

Рисунок 25 – Типова схема взаємодії ПЗ на обчислювальному вузлі

З погляду власника кластера, перші три рівні в системі ПЗ: РПП, ARC і MPI, як інноваційні складові мають високу динаміку змін. Тому особливо жорстких вимог з ІБ до них ставити недоцільно. Та й відповідальність за їхню безпечну роботу лежить на розробниках і користувачах. З іншого боку,

ОС і мережні середовища мають традиційні функції і складають основу ІБ. Для них доцільно жорстко регламентувати вимоги, оскільки від них залежить функціонування Грід-середовища в цілому. Тому ОС і встановлене ПЗ мережі і апаратного забезпечення повинні відповідати стандарту загальних критеріїв безпеки інформаційних технологій – Common Criteria for Information Technology Security Evaluation (Common Criteria або CC) і пропонованих ним гарантованих параметрів відповідності Evaluation Assurance Levels (EAL) не нижче рівня 4+. У цьому випадку один конкретний РПП й може завершитися крахом, але не повинен вплинути на роботу Грід-середовища.

5.2 Сертифікати відкритих ключів

Спрощено структуру Грід можна представити таким чином (рис. 26), тобто при взаємодії користувача з обчислювальними кластерами в Грід хоча й створюється повна ілюзія монопольного володіння ними, але взаємодія виконується посередництвом ВО.

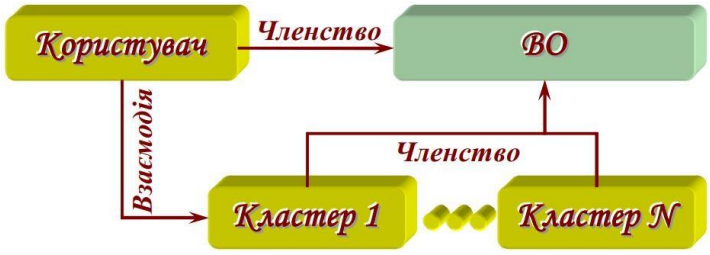

Рисунок 26 – Спрощена структура взаємодії користувача з обчислювальним кластером в Грід

Тому технології ІБ в Грід-системах базуються не на прямих відносинах довіри між традиційними організаціями, оскільки подібний підхід цілком позбавлений гнучкості. Мостом між учасниками певного співтовариства або проекту є віртуальна організація, а посвідченнями особи – сертифікати відкритих ключів, які забезпечують єдиний вхід у сервісно-орієнтоване Грід-

середовище, делегування повноважень і відображення особистостей. При цьому підтримуються такі стандартні прикладні програмні інтерфейси, як Generic Security Service API (GSS-API) до яких відноситься й ARC.

5.2.1 Ідентифікація користувачів і вузлів Грід

Як ідентифікатори користувачів і ресурсів в GSI використовуються цифрові сертифікати стандарту X.509 організації International Telecommunication Union (ITU). У роботі із сертифікатами X.509 і в процедурі видачі/одержання сертифікатів задіяні три сторони:

Перша сторона – це Certificate Authority (CA) Центр Сертифікації, тобто спеціальна організація, яка має повноваження видавати (підписувати електронним способом) цифрові сертифікати. CA власно й є гарантом дотримання правил інформаційної безпеки в Грід-мережах. Зазвичай різні CA не залежать один від одного. Відносини між CA і його клієнтами регулюються спеціальним документом і довіра до сертифікату будується на довірі до CA, який підписав цей сертифікат.

Центр видачі персональних і сервісних сертифікатів (центр сертифікації) – це компонент Грід-мережі, відповідальний за управління криптографічними ключами користувачів і служб. Інформація про служби Грід-мережі зберігається у вигляді сервісних сертифікатів. Відкриті ключі й інша інформація про користувачів зберігається центрами сертифікації у вигляді цифрових персональних сертифікатів формату X.509.

Сервісні сертифікати необхідні для автентифікації служб Грід-мережі між собою і їхньою коректною роботою. Персональні сертифікати користувачів використовуються для авторизації користувачів у Грід-мережі і для запуску завдань у цій мережі від імені користувача.

Центр видачі сертифікатів виконує наступні функції:

– реєструє електронні цифрові підписи;
– створює по обігу користувачів закриті й відкриті ключі ЕЦП;
– припиняє і відновляє дію сертифікатів, ключів, підписів, а також

анулює їх;

– веде реєстр сертифікатів ключів підписів, забезпечує актуальність реєстру і можливість вільного доступу користувачів до реєстру;
– видає сертифікати ключів підписів у вигляді електронних документів з інформацією про їхню дійсність.

Центр видачі сертифікатів вирішує проблеми, пов'язані з авторизацією користувачів у Грід-мережах. Підписуючи електронний сертифікат користувача, центр сертифікації, тим самим, дає користувачеві набір прав на використання Грід. Видаючи повноваження на використання ресурсів Грід-мережі тільки тим користувачам, які мають на це дозвіл, центр видачі сертифікатів вирішує питання, пов'язані із забезпеченням конфіденційності і цілісності даних. Неавторизований користувач не може одержати доступ до служб Грід і порушити їхнє функціонування. Таким чином, допуск до Грід одержують тільки користувачі з певним набором прав.

Другою стороною, яка задіяна в роботі із сертифікатами, є власник сертифіката, який є користувачем Грід або Грід-ресурс, який користується сертифікаційними послугами CA. CA включає в сертифікат дані, надавані власником (ім'я, організація й інша інформація) і завіряє його своїм цифровим підписом. Зокрема власникові сертифікату присвоюється унікальне ім'я (Distinguished Name, DN).

Нарешті третім учасником в процесі сертифікації є користувачі або ресурси, які виконують автентифікацію інших Грід-объектів, і покладаються на інформацію із сертифікату при одержанні його від об'єктів, які ідентифікуються. Ці учасники можуть приймати або відкидати сертифікати, підписані якою-небудь CA.

В GSI використовуються два типи сертифікатів X.509:

По-перше – це Сертифікат користувача (User Certificate, UC), який мусить мати кожен користувач, що працює з Грід-системою. Сертифікат користувача містить інформацію про ім'я користувача, організацію, до якої він належить і CA, який видав даний сертифікат.

По-друге – це Сертифікат вузла (Host Certificate, HC), який мусить мати

кожен вузол (Грід-сервіс або ресурс) Грід-системи. Сертифікат вузла практично аналогічний сертифікату користувача, але у ньому замість імені користувача вказується доменне ім’я конкретного Грід-вузла.

Сертифікати стандарту X.509 містять відкритий ключ і деякі інші дані про користувача або Грід-вузол. Дійсні сертифікати зберігаються в спеціальному репозиторії центра сертифікації, там же зберігається список відкликаних сертифікатів (Certificate Revocation List, CRL). Сертифікаційний центр засвідчує приналежність сертифіката даному користувачеві або Грід-вузлу, які визначаються своїми унікальними іменами (Distinguished Name, DN).

Автентифікація в PKI зводиться до наступних кроків:

а) об’єкт А (користувач або Грід-вузол) хоче автентифікувати об’єкт Б;

б) Б посилає свій сертифікат до А, той перевіряє правильність сертифіката й підпис (або ланцюжок підписів) сертифікаційного центру;

в) А посилає Б довільну фразу із проханням зашифрувати її закритим ключем Б;

г) Б шифрує дані, що надійшли, а відповідь відсилає до А;

д) А розшифровує відповідь Б з допомогою переданого раніше відкритого ключа і порівнює результат з еталонною фразою;

е) якщо порівняння виконане успішно, то це означає, що Б дійсно володіє закритим ключем, який відповідає сертифікату.

Власник сертифіката – може використовувати закритий ключ і сертифікат для шифрування даних і для цифрового підпису. Цифровий підпис це хеш даних, зашифрований закритим ключем (хеш – це число, яке зіставляється до даних таким чином, щоб імовірність появи різних даних з однаковим хешем була близька до нуля, а відновити дані за ним було б практично неможливо). Підпис перевіряється за допомогою відкритого ключа.

5.2.2 Делегування прав і використання доручень

Для забезпечення безпеки в Грід-системах використовуються не самі користувальницькі сертифікати, а спеціальні доручення (proxy certificate,

проксі-сертифікат), з коротким (десь до 24 годин) терміном дії. Ці доручення виписує сам користувач на підґрунті власного постійного сертифіката, діючи в цьому випадку як «сертифікаційний центр» для самого себе. З їхньою допомогою Грід-сервіси й виконують дії від імені користувача – власника сертифіката, наприклад запускають завдання на обчислювальному ресурсі.

Важливою умовою для ефективної роботи розподілених систем є можливість делегування прав користувача Грід-сервісам. Справа в тому, що практично будь-який запит користувача проходить не через один, а через кілька сервісів. І якби не було механізму делегування, користувачеві було б необхідно проходити автентифікацію на кожному сервісі в ланцюжку тих з них, які обробляють даний запит. Це означає, що користувач після відправлення завдання на виконання мусить невідривно знаходитись біля свого комп'ютера й відповідати на запити від кожного сервісу в обробному ланцюжку про свою автентифікацію. Фактично це унеможливлює роботу в Грід-середовищі, особливо в разі, коли виконується запуск великої кількості завдань.

Делегування ж прав дозволяє уникнути цієї проблеми. Процес делегації складається з таких кроків:

– сервіс, якому делегуються права (делегат) створює пару ключів (відкритий і закритий);
– відкритий ключ відсилається тому сервісу, який делегує права (тобто користувачу-власнику вихідного сертифіката або попереднику у ланцюжку обробних сервісів);
– підписаний відкритий ключ (сертифікат) вертається делегатові разом з усім ланцюжком сертифікації.

Це забезпечує те, що користувачеві досить виконати автентифікацію тільки на першому сервісі в ланцюжку, який обробляє запит.

Проксі-сертифікат не може відкликатися, тому створення довгострокового доручення дуже небажано. Однак, при виконанні операцій, які вимагають досить великого проміжку часу, може минути термін дії такого доручення, що призведе до некоректного завершення операції. Щоб виключити подібні випадки використовують спеціальний сервіс, який на запит від користу-

вача автоматично відновляє термін дії проксі-сертифіката без втручання користувача. За своєчасним відновленням проксі-сертифіката стежить брокер ресурсів підсистеми управління завантаженням. При використанні сервісу поновлення доручень, користувач цілком довіряє йому свої повноваження на увесь час дії довгострокового сертифіката.

5.2.3 Сервіс управління віртуальними організаціями й авторизація користувачів

Як відзначалося раніш, ВО є одним з ключових понять Грід-технологій. ВО поєднує співтовариство користувачів, які спільно працюють у деякій прикладній галузі. Таким чином, користувачі, які належать до однієї ВО, ставлять до Грід-ресурсів приблизно однорідні вимоги і мусять обслуговуватися на уніфікованому рівні, що значно спрощує управління Грід. Кожна ВО має власну політику, відповідно до якої і надаються ресурси користувачам даної ВО.

З технічної точки зору відомості про ВО, зареєстровані в цій Грід-інфраструктурі, і про їхній склад зберігаються в базі даних, яка використовується ресурсними центрами для вирішення питання про допуск завдань конкретного користувача на даний ресурс. Грід-ресурси надаються на основі як політики самої ВО, так і локальної політики ресурсних центрів – у цьому й полягає процес авторизації. Для гнучкого управління правами різних користувачів ВО може мати внутрішню структуру, тобто містити різні групи користувачів, а окремим користувачам можуть приписуватися різні ролі. Цим різним групам і ролям у процесі авторизації надаються різні права доступу до Грід-ресурсів (згідно з політикою ресурсних центрів).

У деяких ППЗ (наприклад, gLite) інформацію для авторизації користувачів надає окремий сервіс – VOMS (Virtual Organization Membership Service – Сервіс керування членством у віртуальних організаціях). Сервер VOMS, як системи зберігання даних, використовує реляційні бази даних MySQL або ORACLE і ґрунтується на додаванні некритичних розширень до користува-

льницького проксі-сертифіката. Ці додавання й містять відомості про користувача, необхідні для його авторизації.

5.3 Стандартні протоколи забезпечення безпеки

Насправді існує безліч стандартів забезпечення безпеки, але жоден з них не задовольняє вимоги систем безпеки Грід повною мірою. Тому є сенс хоча б назвати основні з існуючих стандартів і коротко розглянути ступінь їхньої відповідності вимогам, які ставляться Грід-системами.

Протокол Kerberos – стандарт Інженерної ради Інтернету (Internet Engineering Task Force, IETF), який підтримує безпеку системи через установлення дійсності, цілісності й конфіденційності повідомлення, створеного при використанні методу поділюваної секретної криптографії. Цей протокол забезпечує однократну авторизацію і гнучкий захист повідомлень. Однак, виконання вимоги інтеграції з локальними розв’язаннями проблеми безпеки для цього протоколу утруднено, тому що різні реалізації Kerberos мають тенденцію заміняти локальні вирішення для безпеки системи.

Протокол TLS (Transport Layer Security) (з початку відомий як SSL) – IETF стандарт для встановлення дійсності, цілісності й конфіденційності повідомлення, яке створено з використанням технології криптографії з відкритим ключем. Використання цього протоколу утруднено при реалізації однократної авторизації й делегування.

Набір IETF стандартів PKI, які описують протоколи й синтаксис управління сертифікатами X.509 в інфраструктурах систем безпеки з використанням технології відкритих ключів. Стандарти X.509 здебільшого використовуються в сполученні з іншими стандартами безпечного зв’язку, наприклад, TLS.

Стандарт IETF CMS (Cryptographic Message syntax) визначає синтаксис, що дозволяє в цифровій формі підписати, підтвердити дійсність, або зашифрувати довільні повідомлення.

IETF стандарт GSS-API. GSS-API, визначає інтерфейс прикладних про-

грам, що забезпечує встановлення дійсності, цілісності й конфіденційності повідомлення. Він припускає, що дві сторони, які обмінюються повідомленнями, мають зв’язок, оснований на протоколі із забезпеченням надійності доставки інформаційних пакетів (наприклад, TCP/IP). Даний стандарт використовується при реалізації багатьох з основних механізмів безпеки і рекомендується для застосування в розподіленому обчислювальному середовищі. Розширенням стандарту є стандарт IDUP-GSS-API (Independent Data Unit Protection Generic Security Service API), який забезпечує підтримку захисту незалежних одиниць даних.

Підсумовуючи сказане, слід відзначити, що для відкритих Грід-систем ще й досі не прийнята достатня кількість відповідних стандартів, які б повністю задовольняли потреби в захисті інформації в таких системах.

5.4 Контрольні запитання і завдання для самостійної роботи

1. Назвіть шість зв’язаних між собою служб, які характерні для відкритих систем на будь-якій платформі. Стисло охарактеризуйте кожну з служб.
2. Чому засоби безпеки Грід повинні підтримувати добрий десяток засобів захисту інформації? Спробуйте назвати і охарактеризувати як можна більше таких засобів.
3. Як рекомендується диференціювати вимоги до інформаційної безпеки при розробці Грід-систем? Розкажіть про рівні моделі, з використанням якої можна виконати ці вимоги.
4. Назвіть і опишіть типове ППЗ, яке найчастіше використовують розробники різних Грід-середовищ.
5. Чому технології ІБ в Грід-системах базуються не на прямих відносинах довіри між традиційними організаціями, що є мостом між учасниками Грід?
6. Розкажіть про ідентифікацію користувачів і вузлів Грід, поясніть які сторони беруть участь в цьому процесі, які типи сертифікатів видаються учасникам.
7. Поясніть чому в Грід-системах використовуються не самі користуваль-

ницькі сертифікати, а спеціальні доручення (проксі сертифікати). Хто і як створює ці доручення?

8. Розкажіть про сервіс управління віртуальними організаціями і авторизації користувачів.
9. Стисло охарактеризуйте стандартні протоколи забезпечення безпеки Грід.

6 СИСТЕМА ПІДТРИМКИ ФУНКЦІОНУВАННЯ: ПОСЛУГА ПРОТОКОЛЮВАННЯ ПРОЦЕСУ ВИКОНАННЯ ЗАВДАНЬ

В умовах розподіленої інфраструктури й децентралізованої обробки запитів особливим чином повинні вирішуватися питання забезпечення безпеки, надійності, моніторингу процесів, обслуговування великих користувальницьких колективів. У розробках сучасного ППЗ (наприклад, gLite) зроблений істотний крок у цьому напрямку, що дає підставу виділити окрему групу технологій підтримки функціонування Грід. У той же час слід зазначити, що більшість із цих технологій застосовується поки лише в контексті системи управління завантаженням (Workload Management System, WMS), хоча перспективи їхнього використання в системі керування даними й в інформаційній системі досить очевидні.

Тому є сенс коротко розглянути дві служби, які реалізують такі підтримуючі технології.

6.1 Служба протоколювання процесу обробки завдань

Підсистема протоколювання (Logging and Bookkeeping, LB) відслідковує процес виконання завдань, керований підсистемою WMS управління завантаженням, відслідковує кроки обробки завдань, які виконуються в різних вузлах Грід-системи, фіксує події, що відбуваються із завданнями, (запуск, розподіл у придатний ресурсний центр, початок виконання і таке інше) і запам’ятовує їх. Вона збирає повідомлення про події від різних компонентів

WMS і обробляє їх, щоб надати користувачеві узагальнений поточний стан (статус) завдання.

Інформація про події (протокол) поставляється компонентами менеджера завантаження (Workload Manager, WM) і ПЗ управляючого вузла обчислювального кластера (Computing Element, CE), для чого в ці компоненти вбудовуються звертання до LB.

В остаточному підсумку події збираються на сервері зберігання протоколів, але реалізується це у два етапи. Спочатку події передаються в локальну службу (Locallogger), яка виконується в безпосередній фізичній близькості від джерела події, що дозволяє уникнути яких-небудь мережних проблем при передачі. Локальний демон записує подію у файл на диску й повертає джерелу підтвердження про успішність операції.

Далі на другому етапі за доставку події відповідає інший демон Interlogger. Одержуючи подію або прямо від демона Locallogger, або, у випадку його падіння, з диска, він пересилає її в кінцеву точку призначення – одному із серверів зберігання (Bookkeeper server), причому надійність на цьому етапі забезпечується механізмом повторної передачі. Можна помітити, що в цілому двоетапний спосіб доставки повідомлень відповідає схемі, застосовуваній в R-GMA: демони LB відіграють роль постачальника, а сервер зберігання – споживача.

В інфраструктурі може бути кілька серверів зберігання, однак події від всіх завдань одного користувача надходять тільки на один з них – сервер статично визначений в конфігурації. Сервер Bookkeeper виконує декілька функцій. По-перше він «укрупнює» події, надаючи загальну картину зміни станів завдання (Submitted, Running, Done...). Крім того, Bookkeeper зберігає різні атрибути завдання: його опис мовою опису завдань (Job Description Language, JDL); CE, на якому воно виконувалося; коди завершення й т.ін. Як протокол станів, так і протокол подій можна одержати або за допомогою спеціального інтерфейсу WM, або через повідомлення при певних змінах стану, наприклад, при закінченні завдання.

Користувач не бере участі у зборі даних від компонентів WMS, а прос-

то звертається до LB за необхідною інформацією. LB надає інтерфейси для запитів інформації про завдання, а також для реєстрації запитів на одержання повідомлень про зміни стану завдань.

Підсистема протоколювання підтримує два типи запитів – запити про стан завдань, які повертають детальний опис станів одного або більшої кількості завдань, і запити про події, які повертають інформацію про події, яка одержується LB від компонентів WMS. Як правило, запити про завдання використовуються, щоб простежити штатну обробку завдань; запити про події використовуються, головним чином, для того, щоб простежити аварійну поведінку.

Кожен запит містить декілька умов. Наприклад, вказується конкретний ідентифікаційний номер завдання (він привласнюється завданню підсистемою завантаження і повідомляється користувачеві, який направив це завдання), ім’я власника завдання (як воно зазначено в сертифікаті власника), специфічний стан завдання й так далі. Підсистема протоколювання перетворює умови в повідомлення запиту, обробляє його і чекає відповіді від відповідного компонента Грід. Ця відповідь потім передається користувачеві синхронним чином.

Інший спосіб взаємодії користувачів з підсистемою протоколювання полягає в процесі проходження реєстрації для одержання повідомлень. Їх поставляють слухаючому клієнтові асинхронно, коли відбувається певна подія (звичайно – зміна стану завдання). Основна мета цієї функціональної можливості підсистеми протоколювання – уникнути непотрібного завантаження сервера LB численними повторними запитами, причому в більшості випадків – з однаковим результатом.

Використовуючи клієнта повідомлень, користувач реєструється на сервері LB, для одержання повідомлень. При цьому він мусить визначити умови, при яких повідомлення будуть йому висилатися. Запит на реєстрацію посилається серверу LB у той самий спосіб, що й синхронні запити, і зберігається там. У відповідь, сервер надає і повертає унікальний ідентифікаційний номер запиту на повідомлення, за допомогою якого користувач надалі може зверта-

тися до цього сервера, наприклад для зміни умов, які викликають повідомлення, продовження періоду дії реєстрації або її скасувань і навіть для зміни адресата повідомлень.

Безсумнівно, що інформація про завдання, яка зберігається на сервері LB, повинна бути доступна тільки для власника завдання (користувача, який відправляв завдання) і, можливо, для тих користувачів, яких власник указав у спеціальному списку контролю доступу (Access Control List, ACL). Користувачі з ACL можуть бути визначені безпосередньо за їхніми іменами з відповідних сертифікатів, або за назвами груп віртуальних організацій, або ще й за назвами цілих ВО.

Для забезпечення контролю за доступом до інформації всі компоненти LB повинні взаємодіяти на основі взаємної автентифікації, а користувачі, які виконують запити на сервер LB, повинні мати дійсний сертифікат. Всі повідомлення, послані по мережі, шифруються, щоб їхній вміст був недоступним для сторонніх.

Підсистема LB одержує повідомлення від безлічі джерел повідомлень, пов’язаних з виконанням завдань у Грід-системі. Нижче наведений перелік деяких з цих можливих джерел:

– користувальницький інтерфейс реєструє завдання в LB і надає інформацію щодо передачі завдання ресурсу-брокеру;
– брокер ресурсів реєструє різні події в міру проходження завдання через компоненти WMS, а також іншу важливу інформацію, пов’язану із завданням (наприклад, вибір CE, на якому буде виконуватися завдання);
– обчислювальний елемент забезпечує інформацію про хід виконання завдання;
– крім зазначених вище компонентів WMS, генерувати події може також спеціальний фрагмент користувальницького коду в самому завданні, так званий тег користувача (User Tag);
– завдання з контрольними точками (checkpointable) також можуть використовувати LB, щоб стежити за просуванням завдання;

– нарешті, сама LB може бути джерелом повідомлень, наприклад, при зміні ACL – списків управління доступом до інформації про завдання.

6.2 Служба обліку споживання ресурсів

Служба обліку споживання ресурсів – DataGrid Accounting System (DGAS) є засобом збору інформації про використання ресурсів користувачами Грід. Інформаційною одиницею в цій системі є запис використання ресурсів (Usage Records, UR), який створюється для кожного обробленого завдання й містить такі дані:

а) ідентифікатор завдання;

б) ресурсний центр, у якому воно виконувалося;

в) використані при виконанні ресурси;

г) час:

1) запуску на обчислення;
2) надходження до черги локального менеджера;
3) закінчення;

д) час створення запису.

Збір цих даних здійснює демон Gianduia, який працює в середовищі CE і взаємодіє з його локальним менеджером. Механізм збору виглядає в такий спосіб. Коли завдання починає виконуватися на виконавчому комп'ютері, його стартовий скрипт Job Wrapper створює в спеціальній директорії файл, який містить ім'я завдання, його ідентифікатор і адресу сервера, на який спрямовується формований запис використання. Виявляючи такий файл, демон Gianduia шукає облікову інформацію про завдання в лог-файлі локального менеджера. Коли завдання закінчується, вся облікова інформація вибирається із цього лог-файла й приєднується до файла, який був створений спочатку.

Записи використання, які збираються в системі, зберігаються в децентралізованій інфраструктурі, що утворена з багатьох серверів, на яких установлена служба Home Location Register (HLR). Організація інфраструктури зберігання має в DGAS особливості, зумовлені наступною обставиною.

Обслуговування запитів на видачу інформації в інфраструктурі з багатьох серверів істотно спрощується, якщо інформація розподіляється між ними таким чином, що всі дані, які належать певному користувачу, сконцентровані лише на одному з серверів. Саме такий підхід й застосовується в DGAS: кожен користувач приписаний до своєї «домашньої» служби HLR. Однак, DGAS призначена для обслуговування не тільки користувачів, але й провайдерів ресурсів. Якщо користувачі зацікавлені в одержанні відомостей тільки про свої завдання, то провайдерам потрібні дані про всі завдання, які були оброблені на їхніх ресурсах. При орієнтації тільки на користувальницьке обслуговування провайдери були б змушені опитувати кожен сервер.

У зв’язку із цим в інфраструктурі зберігання підтримуються два види служб HLR: для користувачів і для провайдерів. Записи використання, які збираються компонентами DGAS на CE, спочатку поставляються в одну з можливих служб HLR, а потім дублюються нею в іншу. Підтримується кілька сценаріїв поставки:

– спочатку в користувальницький HLR:

CE ⇨ User HLR ⇨ Resource HLR;

– спочатку в HLR провайдерів:

CE ⇨ Resource HLR ⇨ User HLR;

– тільки в HLR провайдерів:

CE ⇨ Resource HLR;

В складі DGAS є ще одна служба, дані якої становлять інтерес для всіх учасників Грід – це служба розцінок (Price Authority, PA). Служба підтримує базу даних цін ресурсів і зберігає історію зміни цін. Ціни в ній можуть встановлюватися адміністратором вручну, але можуть також і обчислюватися програмно по різних алгоритмах, наприклад по алгоритму торгів. В PA реалізований механізм додавання і вбудовування алгоритмів установлення цін у формі динамічної бібліотеки.

Наявність розцінок ресурсів у службі PA, з одного боку, і кількості споживаних завданнями ресурсів у запису HLR, з іншого, дають можливість обчислити вартість завдань. Таким чином, реалізовані в DGAS служби до-

зволяють акумулювати інформацію про використання ресурсів Грід окремими користувачами, групами користувачів і віртуальних організацій. Зібрана інформація дозволяє побудувати загальну картину діяльності в Грід, на основі якої може формуватися політика розподілу ресурсів і формуватися інформація про вартість використання даного Грід-ресурсу даним користувачем, якщо взаємини користувачів і провайдерів ресурсів базуються на економічній моделі.

6.3 Контрольні запитання і завдання для самостійної роботи

1. Для чого призначена підсистема протоколювання процесу обробки завдань? Як функціонує ця підсистема?
2. Які типи запитів підтримує підсистема протоколювання? Розкажіть про ці запити.
3. Що є інформаційною одиницею в службі обліку споживання ресурсів? Які дані містить запис використання ресурсів, в чому полягає механізм збору даних?
4. Опишіть механізм збереження і обслуговування записів використання в системі. Які два види служб зберігання підтримуються в інфраструктурі?

7 GRID-ПОРТАЛ ДЛЯ ДОСТУПУ КОРИСТУВАЧІВ ДО РЕСУРСІВ І ПРИКЛАДНИХ ПРОГРАМ GRID

Для входу в Грід-систему користувач мусить:

- бути легальним користувачем обчислювальних ресурсів у своїй організації;
- мати персональний цифровий сертифікат, підписаний центром сертифікації;
- бути зареєстрованим хоча б в одній віртуальній організації.

Якщо дотримані всі три умови, то доступ до Грід-системи виконується з будь-якої точки: обчислювальної системи, термінала й, навіть, з будь-якого

мобільного пристрою. Така точка мусить ще мати установлений користувальницький інтерфейс Грід-системи. Він може виконуватись різними способами – інтерфейс командного рядка або веб-інтерфейс, однак мусить надавати користувачу можливість повноцінно працювати з Грід-системою, тобто запускати нові завдання, управляти вже запущеними і одержувати результати роботи завдань, які завершилися. Найбільш стандартним є інтерфейс командного рядка (Command Line Interface, CLI), який дозволяє виконувати всі операції з управління завданнями й даними, а також адміністративні дії. Але велика кількість користувачів (особливо недосвідчених) вважають цей інтерфейс «недружнім» до користувача і занадто універсальним з відсутністю орієнтації на конкретне застосування. Таким користувачам набагато зручніше працювати із середовищем, яке орієнтоване на задачу розв'язувану ними і яке надає зручний графічний інтерфейс. В цьому разі вимоги користувачів повністю забезпечуються веб-інтерфейсами, які дозволяють працювати із Грід-системою безпосередньо із браузера.

Веб-орієнтовані обчислювальні портали (Грід-портали) є ефективним інструментом для забезпечення користувачів обчислювальних Грід простим інтерфейсом для доступу до інформації й використання Грід-ресурсів.

Портали можуть розглядатися як забезпечення веб-інтерфейсу для розподілених систем. Зазвичай портали будуються на основі трьохярусної архітектури, яка складається з:

- клієнтів (перший рівень);
- брокерів або серверів (середній рівень);
- репозиторіїв об'єктів (обчислювальні сервери, бази даних або будь-які інші ресурси або сервіси).

Це загальна архітектура й використовуючи її, можна будувати портали, які підтримують різноманіття практично будь-яких прикладних галузей, наприклад, портали галузей науки, обчислювальні портали, портали покупок, портали освіти і так далі. Однак, щоб зробити це ефективно, потрібен набір відповідних інструментальних засобів, які спеціально розробляються й підготовляються для кожної прикладної галузі.

Веб-портал дає можливість ученим і дослідникам мати доступ до інформаційних ресурсів, специфічних для їхньої предметної області, за допомогою веб-інтерфейсу. На відміну від звичайних тематичних веб-порталів портал Грід може також забезпечувати доступ до обчислювальних ресурсів Грід (наприклад, підтверджувати автентичність користувачів, давати їм дозвіл на доступ до віддалених ресурсів, допомагати приймати рішення щодо планування завдань, одержувати й обробляти інформацію про ресурси, які зберігаються у віддалених базах даних). Портал Грід може також бути індивідуалізований за допомогою профілів, які створюються й зберігаються для кожного користувача порталу.

В основному портал Грід діє як інформаційний накопичувач, який одержує доступ до різних інформаційних ресурсів.

7.1 Архітектура Грід-порталу на прикладі порталу GENIUS

Простота використання – обов'язкова умова доступності і інформаційної безпеки обчислювальних сервісів. Еталоном простоти є веб-доступ, тому природно скористатися веб-технологією і реалізувати інтерфейс з Грід-системами у вигляді веб-порталу, який надає сервіси автентифікації, авторизації, запуску і відстеження завдань. Такий підхід і реалізований у Грід-порталі GENIUS[1)].

Повнофункціональний Грід-портал GENIUS (Grid Enabled web eNvironment for site Independent User job Submission), розроблений в італійському інституті INFN (Istituto Nazionale di Fisica Nucleare – Національний інститут ядерної фізики Італії) активно використовується в рамках проекту GILDA (Grid INFN Laboratory for Dissemination Activities), який є віртуальною лабораторією для демонстрації можливостей технології Грід.

Випробний стенд GILDA складається з десятків сайтів на трьох континентах, він використовує гетерогенні апаратні засоби і діє як реальне середо-

[1)] Welcome to the GENIUS INFN GRID Portal. URL:https://genius.ct.infn.it (дата звернення 18.07.2018)

вище Грід. Стенд зібраний з тих же компонентів, які використовуються в потужніших Грід-проектах, включаючи системи тестування й моніторингу.

Конкретно ця версія програмного забезпечення Грід реалізована в рамках проекту європейського сегмента глобального Грід – European DataGrid (EDG). ПЗ містить усі сервіси, необхідні для виконання наступних завдань:

– автентифікація і авторизація користувачів;
– запуск завдань і ведення журналів;
– управління даними (публікація файлів, реплікація та видалення);
– створення і моніторинг інформації про Грід;
– підготовка і управління комп'ютерним обладнанням;
– взаємодія з ієрархічними системами масового зберігання (MSS);
– мережевий моніторинг.

Представлення ППЗ DataGrid у звичайній багатошаровій схемі «пісковий годинник» показано на рис. 27. Служби DataGrid покладаються на проміжне програмне забезпечення, яке надається інструментарієм Globus.

Рисунок 27 – Багатошарова структура ППЗ DataGrid,
включаючи портал GENIUS

Всі послуги, які надаються ППЗ DataGrid, доступні через портал GENIUS, який має трирівневу архітектуру (рис. 28):

– клієнт: робоча станція користувача, на якій запущений веб-браузер споріднений Mozilla;
– сервер: машина, на якій працює веб-сервер Apache, шар ПЗ Java/XML EnginFrame (розроблений компанією NICE srl), власне GENIUS і підтримується інтерфейс користувача EDG (оснащений сервісами ППЗ DataGrid і здатний подавати завдання та керувати даними у Грід);
– віддалені ресурси: власне Грід.

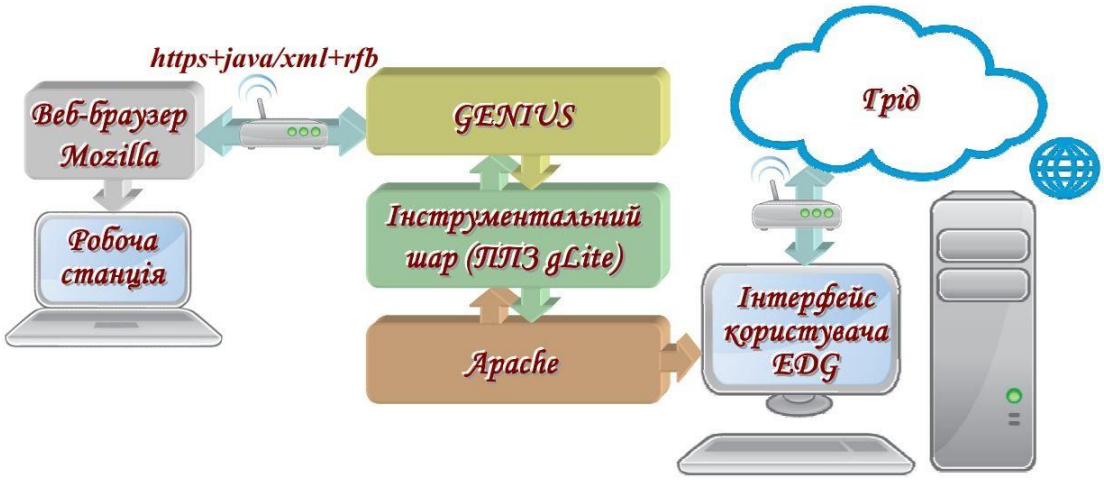

Рисунок 28 – Трирівнева архітектура Грід-порталу GENIUS

За операційну платформу GENIUS править Linux.

Щоб забезпечити захищений доступ до Грід-ресурсів, в GENIUS реалізована багаторівнева інфраструктура безпеки, яка включає протоколи HTTPS, SSL, користувальницькі рахунки на сервері, переспрямовування користувачів (після успішного входу на сервер) для обслуговування сервісом управління віртуальними організаціями.

Практично користувач входить у сітку з машини з інтерфейсом користувача, використовуючи свій цифровий особистий сертифікат.

За допомогою надаваних порталом GENIUS сервісів користувач може:

– опитувати доступні обчислювальні елементи;
– створювати завдання;
– передавати завдання на виконання, можливо, специфікуючи обчислювальні елементи;
– відслідковувати хід виконання завдань;
– «на льоту» переглядати вивід, породжуваний запущеними завданнями, і зберігати його на серверній і/або клієнтській системах.

GENIUS надає сервіс управління даними з можливістю їх інтерактивного аналізу і графічного відображення. За основу захисту переданих даних править протокол SSL.

Складною проблемою для Грід-порталів є універсальність, тому що високопродуктивні застосування дуже численні і різноманітні, до того ж постійно з'являються нові. Архітектура Грід-порталу мусить забезпечувати нескладне вбудовування користувальницьких застосувань в систему, а також криптографічний захист користувальницьких даних, у силу великого обсягу яких – краще за все вибірковий. Обидві задачі вирішуються на основі моделі портлетів.

Тут, як вирішення проблеми, пропонується трирівнева архітектура Грід-порталу: базовий рівень, рівень сервісів і рівень портлетів.

На базовому рівні реалізуються контейнер порталу, збереження даних, а також точка доступу до Грід-систем для вищерозташованих рівнів. Надаються такі базові портлети, як Login, Logout, персоналізація профілю користувача, настроювання зовнішнього вигляду і т.ін. Підтримуються прикладні програмні інтерфейси для управління завданнями з галузі високопродуктивних обчислень: створення завдань, опитування їхнього статусу, завантаження і вивантаження файлів, термінування завдань.

Рівень сервісів надає рівню портлетів такі послуги, як захист даних і управління завданнями. На цьому рівні генеруються криптографічні ключі, здійснюються шифрування й розшифрування даних, якими веб-навігатор обмінюється з сервером. Для завдань підтримується повний життєвий цикл.

Рівень портлетів містить численні застосування і функції управління

завданнями. Захисні функції представлені як на рівні сервісів, так і на рівні портлетів.

Модель безпеки портлетів складається із двох частин – навігаційної й серверної. У веб-навігаторі для захищеної передачі файлів використовується аплет SFTP. Серверна частина містить модуль захисту даних і, зокрема, реалізує симетричний алгоритм блочного шифрування Advanced Encryption Standard (AES). При обміні даними з користувальницьким веб-навігатором виконується вибіркове шифрування: зашифровуються описи завдань, які надходять у систему, вхідні і вихідні дані завдань, а також команди на термінування завдань.

Підсумовуючи, можна відзначити, що в основі нормального функціонування Грід-систем і їхньої інформаційної безпеки лежить сервіс-орієнтована архітектура, точніше кажучи, архітектура відкритих Грід-сервісів, яка забезпечує взаємну сумісність різних елементів, масштабованість, гнучкість і настроюваність.

Поняття віртуальної організації, динамічна підтримка стосунків довіри, використання сертифікатів для автентифікації, авторизації і делегування повноважень правлять за базу інфраструктури безпеки, вирішують проблеми автентифікації і управління доступом.

Програмне забезпечення проміжного рівня уніфікує інтерфейс до локальних сервісів, деякою мірою приховує розподілену природу Грід-конфігурацій.

Орієнтація на веб-технології, XML і асоційовані з ними специфікації дозволяє досягти стандартизації архітектурних елементів інформаційної безпеки, будувати великі розподілені системи апробованими методами з апробованих компонентів, забезпечує однакове розв’язання базових проблем безпеки, таких як реалізація захищених комунікацій, узгодження політик безпеки різних ланок управління.

Через велику динамічність Грід-конфігурацій для них потрібна активна безпека, що досягається засобами моніторингу і управління. Стандартизація цих засобів, їхня багаторівнева організація вирішують проблеми масштабо-

ваності й взаємної сумісності.

Простота використання – необхідна умова формування режиму інформаційної безпеки. Досягти її можна засобами веб-технологій, реалізувавши Грід-портал і організувавши користувальницький інтерфейс на основі веб-навігатора.

Перераховані архітектурні принципи і підходи забезпечують можливість досягнення необхідного рівня інформаційної безпеки Грід-систем.

7.2 Контрольні запитання і завдання для самостійної роботи

1. За яких умов і яким чином користувач отримує можливість доступу до Грід-системи, які інтерфейси надаються йому для цього?
2. Опишіть загальну архітектуру веб-орієнтованих обчислювальних порталів (Грід-порталів).
3. Які завдання виконуються сервісами ПЗ Грід-порталу GENIUS? Розкажіть про DataGrid – ППЗ порталу у відповідності до звичайної схеми «пісковий годинник».
4. З яких елементів створюється трирівнева архітектура порталу GENIUS, які дії користувач може виконати за допомогою надаваних порталом сервісів?
5. Які задачі вирішуються на основі моделі портлетів? Як трирівнева архітектура порталу (базовий рівень, сервіси і портлети) сприяє вирішенню проблем Грід-порталів?
6. Розкажіть яким чином сервіс-орієнтована архітектура відкритих Грід-сервісів забезпечує взаємну сумісність різних елементів Грід-систем: масштабованість, гнучкість, настроюваність.

8 GRID-ЗАСТОСУВАННЯ

Первісно технології Грід використовувалися для наукових і інженерних застосувань. Однак тепер вони стають основою для координованого спільно-

го використання ресурсів у динамічних віртуальних організаціях, які охоплюють багато підприємств у державному управлінні, у медицині, у промисловості, у бізнесі. Взагалі слід підкреслити, що з появою різних Грід-інфраструктур з'явились і почали активно використовуватися нові терміни e-Science, e-Health, e-Commerce, які підкреслюють найтісніший зв'язок у розвитку науки, медицини, бізнесу із сучасними інформаційними технологіями, у першу чергу, з Грід-технологіями. Таким чином, Грід служить універсальною ефективною інфраструктурою для високопродуктивних розподілених обчислень і обробки даних.

За застосування Грід правлять:

– складне моделювання на віддалених суперкомп'ютерах;
– спільна візуалізація дуже великих наборів наукових даних;
– розподілена обробка з метою аналізу даних;
– з'єднання наукового інструментарію з віддаленими комп'ютерами і архівами даних.

Серед основних напрямків використання Грід на даний момент можна виділити:

– організацію ефективного використання ресурсів для невеликих задач, із залученням комп'ютерних ресурсів, які тимчасово простоюють;
– розподілені суперобчислення для розв'язання дуже великих задач, які вимагають величезних процесорних ресурсів, пам'яті і т.ін.;
– обчислення із залученням великих обсягів географічно розподілених даних, наприклад, у метеорології, астрономії, фізиці високих енергій, медицині, науках про землю;
– колективні обчислення, у яких одночасно беруть участь користувачі з різних організацій.

Яскравим прикладом використання Грід-інфраструктури для розв'язання своїх завдань дослідниками з цілого ряду наукових і виробничих галузей у цей час є світова Грід-система EGEE – Enabling Grids for E-sciencE (рис. 29), тобто Грід пристосований для наукових розробок (e-Science), які потре-

бують інтенсивних обчислень у високорозвинених мережних середовищах, або використовують величезні набори даних для обчислень.

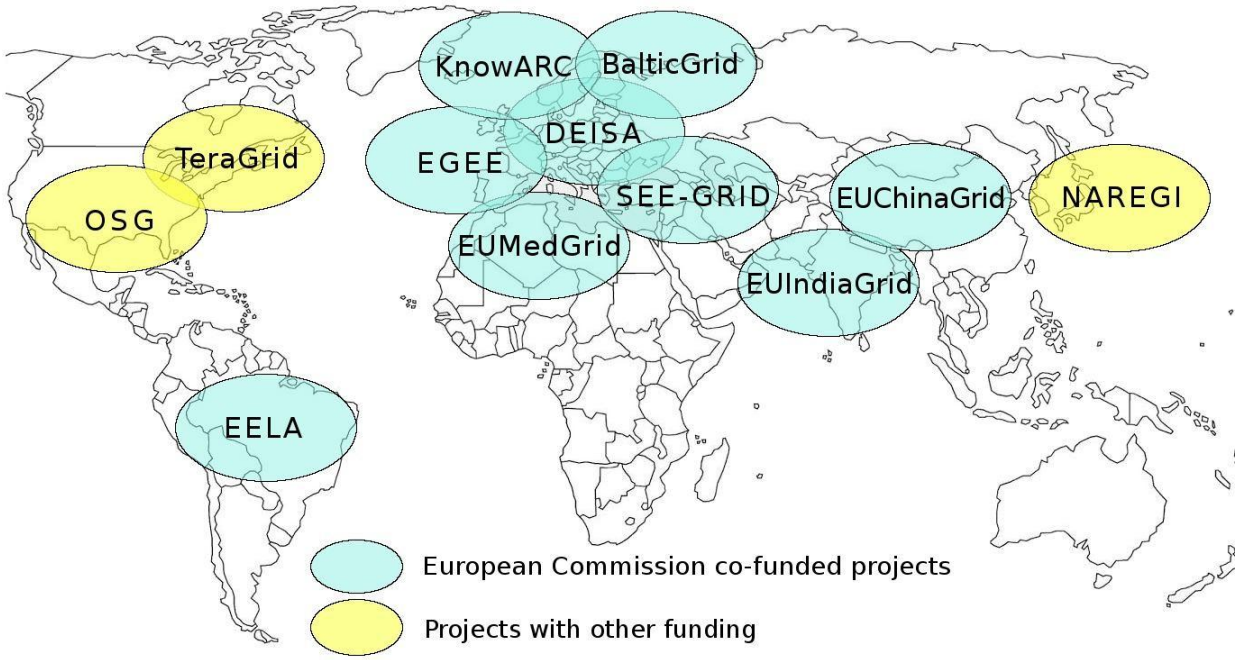

Рисунок 29 – Загальна інфраструктура проекту EGEE і споріднених проектів[1)]

Проект EGEE спочатку розроблявся лише для робіт у галузях фізики високих енергій і біомедичних наук. З часом проект почав підтримувати найширший діапазон галузей досліджень. Тепер до нього входять, наприклад, такі галузі застосування, як розробка мультимедійних засобів і технологій, астрофізика, археологія, обчислювальна хімія, фізика термоядерного синтезу, науки про Землю, матеріалознавство, моделювання процесів на фінансових ринках і інше. Дослідники об'єднуються у ВО і можуть з їхньою допомогою співробітничати, користуватися загальними ресурсами й мати доступ до загальних наборів даних через інфраструктуру EGEE.

Дуже важливо, що багато хто з учасників проекту після деякого часу, присвяченого процесу тестування своїх застосувань у Грід-середовищі, пере-

[1)] Кореньков В.В. Грід технологии: статус и перспективы. Вестник международной академии наук (Русская секция). 2010. №1. С. 41–44.

ходять до етапу практичної рутинної роботи і запуску задач з метою одержання нових результатів у своїх галузях. При цьому ефективність завантаження Грід-ресурсів EGEE сягає ~80-90%.

8.1 Основні користувачі Грід-інфраструктури EGEE

Загалом кажучи, Грід-інфраструктура EGEE ідеально пристосована для будь-яких наукових обчислень, які не можуть бути виконані на одному комп'ютері або вимагають обчислень, які можуть забрати непередбачений час для завершення навіть на окремому комп'ютерному кластері. Грід EGEE уже зараз має близько 20000 процесорів, які доступні 24 години на добу, 7 днів на тиждень, і надає більш ніж 5 Петабайтів пам'яті. Середня кількість завдань, які виконуються одночасно в системі, сягає 10000.

Завдяки Грід-інфраструктурі EGEE, вченим уже зараз доступні принципово нові можливості для проведення наукових досліджень, а наявність такого потужного інструменту повністю змінює методи їхньої роботи. Як саме використовується Грід – залежить від того, що саме потрібно користувачеві: доступ до великих обсягів пам'яті, або доступ до швидкісних каналів зв'язку, який забезпечує Грід-інфраструктура, або їх привертає доступна обчислювальна потужність Грід.

8.1.1 Застосування у фізиці високих енергій

Співтовариство фахівців з фізики високих енергій (ФВЕ) було одним з двох перших користувачів застосувань, для яких розроблявся EGEE, і зараз залишається одним з найкрупніших користувачів інфраструктури EGEE. Звідси EGEE одержує знання й досвід, винятково важливі для забезпечення високого рівня сервісів, орієнтованих на користувачів.

Головними користувачами є низка експериментів з CERN: у день вони виконують понад 20 тис. завдань і щорічно роблять багато сотень Терабайтів даних. В інших крупних експериментах: BaBar, CDF, H1, ZEUS і D0 також

застосовуються Грід-технології. Але у цих експериментах в інфраструктурі EGEE ведеться звичайна обробка фізичних даних.

Оскільки за своєю природою застосування для ФВЕ ставлять дуже високі вимоги до інфраструктури EGEE, вони дуже сильно сприяють поліпшенню сервісів EGEE і виробленню принципових підходів до їхнього розвитку. Це відноситься до всіх сервісів – від документації і підтримки користувачів до розробки ППЗ. Крім того, у ході експериментів у ФВЕ створюються цінні компоненти ППЗ, які можна вважати прототипами для всього співтовариства користувачів Грід-технологій. Досвід користувачів, які представляють ФВЕ, доступний всім іншим користувачам Грід EGEE. Галузь застосувань ФВЕ є однією з рушійних сил усередині EGEE тому, що співробітництво представників різних галузей науки дуже сильно сприяє прогресу Грід-технологій.

8.1.1.1 Експерименти на Великому Адронному Колайдері

Large Hadron Collider – Великий адронний колайдер (LHC) – найбільший у світі прискорювач елементарних частинок, створений в CERN (Європейської організації ядерних досліджень) у Женеві, Швейцарія. На ньому проводяться, наприклад такі експерименти: ALICE, ATLAS, CMS і LHCb. Вони використовують Грід-ресурси як проекту EGEE, так і споріднених йому проектів, як OSG у США і NDGF у Європі. Потік експериментальних даних, які обробляється при роботі прискорювача, надзвичайно великий – близько 15 Петабайтів у рік. Зосередження комп'ютерних ресурсів, необхідних для обробки такого потоку даних, безпосередньо в CERN є важко здійсненним завданням, як з технічних, так і з фінансових причин. У результаті й було створене розподілене по всьому світу середовище для обробки фізичних даних засобами промислового рівня. Застосування інфраструктури EGEE уже набуло масового характеру і стало звичайною практикою. Розробка наукової програми експериментів на LHC тепер у величезній мірі ґрунтується на застосуванні інфраструктури EGEE. Супутнім результатом є той факт, що в хо-

ді цієї діяльності перевіряється стійкість інфраструктури до стресових навантажень, що є найважливішим для роботи LHC. Із самого початку EGEE утворив стратегічний альянс із проектом LHC Computing Grid (LCG), який спрямований на створення необхідних комп'ютерних потужностей, потрібних для обробки цих даних. Цей альянс забезпечив велику кількість ресурсів для початкової інфраструктури EGEE, також як і застосування, які представляють експерименти LHC.

У кожного експерименту свої фізичні завдання, але в усіх них ведеться великомасштабне моделювання подій, які реєструються при реальному зіткненні високоенергетичних пучків протонів або важких іонів.

- ALICE (A Large Ion Collider Experiment – «Експеримент на великому іонному колайдері») – експеримент з вивчення фізики сильних взаємодій при надвисоких щільностях, де очікується утворення нового стану речовини: кварк-глюонної плазми.
- В експерименті ATLAS (A Toroidal LHC ApparatuS – «Тороїдальна установка на LHC») вивчаються глибинні основи будови речовини і фундаментальних сил, які сформували Всесвіт.
- CMS (Compact Muon Solenoid) – детектор для досліджень, у ході яких очікується виявлення бозона Хіггса і докази суперсиметрії.
- LHCb (The Large Hadron Collider Beauty Experiment) – експеримент з вивчення порушення симетрії заряду і парності. Цей ефект ймовірно є причиною нерівноваги між речовиною і антиречовиною при народженні Всесвіту.

У кожному експерименті LHC використовується своє середовище з багатьох Грід, яке склалося на основі передісторії експерименту і його обмежень. У двох найбільших експериментах LHC – ATLAS і CMS – беруть участь близько 200 інститутів; значна частина цих ресурсів знаходиться в США. В обох експериментах на додаток до EGEE використовується проект GRID3 і створені робочі інтерфейси до цих Грід. В експерименті ATLAS використовується також NorduGrid. В експериментах ALICE і LHCb ще до EGEE, у системах ALICE Environment (AliEn) і DiRAC була розроблена влас-

на Грід-подібна розподілена технологія. Ця розподілена технологія управління даними і завданнями дотепер частково задає напрямок розвитку програмного забезпечення gLite.

У проекті ARDA (A Realization of Distributed Analysis for LHC – реалізація розподіленого аналізу для LHC) розроблені прототипи систем аналізу, у яких є інтерфейс із ППЗ gLite, завдяки чому у фізиці високих енергій перейшли до обробки даних окремими користувачами. Це – децентралізоване використання Грід сотнями окремих користувачів на противагу десяткам керівників робіт.

8.1.1.2 Застосування для фізики високих енергій окрім LHC

Інші експерименти в області ФВЕ, які використовують інфраструктуру EGEE, також перебувають у стадії збору і обробки даних. Фізичні дослідження в них ведуться на найсучаснішому рівні, і там виникають ті ж задачі, з якими зштовхнувся й LHC. Крім того, вони цікавлять проект EGEE, ще й тому, що Грід у них застосовуються не в звичайний спосіб так, як на LHC, а їх лише планують до подібного застосування. Збір даних у цих експериментах ведеться вже кілька років, тому в них налагоджена вся послідовність обробки даних, і фізичні результати видаються регулярно.

От деякі із цих експериментів:

– BaBar – експеримент на Стенфордському лінійному прискорювачі в Каліфорнії. Ціль – вивчення порушення заряду і парності в розпаді b-мезонів;
– D0 – експеримент на колайдері «Теватрон» у Національній прискорювальній лабораторії ім. Енріко Фермі (Фермілаб) у місті Батавія (штат Іллінойс, США), у якому на субатомному рівні ведеться пошук зазначень на характер «стандартних блоків», які утворюють Всесвіт;
– CDF (Collider Detector at Fermilab – «Колайдерний детектор Фермілаб») – ідентифікація й вивчення властивостей частинок, з яких складається Всесвіт, і вивчення сил і взаємодій між ними.

Оскільки за своєю природою застосування для фізики високих енергій ставлять дуже високі вимоги до інфраструктури EGEE, вони дуже сильно сприяють поліпшенню сервісів EGEE і виробленню принципових підходів до їхнього розвитку. Це стосується всіх сервісів – від документації і підтримки користувачів до розробки ППЗ. Крім того, у ході експериментів у ФВЕ створюються дуже цінні компоненти ППЗ, які можна вважати прототипами для всього співтовариства користувачів Грід-технологій.

У всіх цих випадках EGEE продемонстрував, що може відповідати вимогам великих і розподілених наукових співтовариств із підвищеними запитами до потужності доступних комп'ютерних ресурсів і до обсягів пам'яті. Успішна підтримка застосувань ФВЕ стала наочним прикладом у демонстрації переваги підключення до інфраструктури EGEE для інших наукових співтовариств.

8.1.2 Застосування в галузі біомедицини

Біомедичні науки із самого початку були включені в проект EGEE і є однією з найважливіших галузей проекту. Стала експлуатація застосувань промислового рівня в цій галузі є звичайною практикою. Грід-технології дозволили біомедичному співтовариству провадити дослідження дуже широко географічно розподіленими колаборациями, які використовують спільні набори даних і виконують обчислення, які вимагають значної пропускної здатності. В інфраструктурі EGEE розгорнуті або розгортаються більш двох десятків окремих застосувань, які відносяться до наступних трьох напрямків: обробка медичних графічних даних, біомедичні дисципліни, розробка ліків. Для кожного з напрямків в інфраструктурі EGEE уже розгорнута безліч окремих застосувань.

Ці застосування висувають особливі вимоги до ППЗ, особливо в плані безпеки (чутливість даних), до управління даними (складна структура і розподіленість даних) і до виконання безлічі простих завдань, які вимагають інтенсивної роботи з даними.

Біомедичні застосування стабільно працюють в інфраструктурі EGEE у режимі звичайної експлуатації: у рік виконується близько 15 тис. завдань, а в процесі розробки одного з ліків за місяць був проведений аналіз молекулярного докінгу, який зажадав би 80 років роботи єдиного ЦП.

Нижче наведений огляд деяких біомедичних застосувань, розміщених в інфраструктурі EGEE.

По-перше – це завдання сектора обробки медичних графічних даних, тобто комп'ютерний аналіз цифрових медичних зображень. У нього входять: інтегровані медичні дані, медичні алгоритми, які вимагають значних комп'ютерних ресурсів, обробка великих обсягів даних, статистичні дослідження на великих вибірках населення. Серед них можна відзначити цілу низку застосувань.

- GATE – моделювання на основі методу Монте-Карло і графічних даних обстеження пацієнта. Мета моделювання – планування сеансів радіотерапії. Грід-інфраструктура EGEE використовується для зменшення, до розумного з погляду клінічної практики, часу, необхідного для одержання досить достовірних результатів при моделюванні методом Монте-Карло.
- CDSS (Clinical Decision Support System – «Система підтримки прийняття клінічних рішень») на основі експертних знань класифікує зображення з метою допомоги лікарям у прийнятті ними рішень. Грід-інфраструктура EGEE використовується як для накопичення великих обсягів даних, так і для ефективного «навчання» на цих даних ПЗ, яке виконує класифікацію.
- Pharmacokinetics – призначений для вивчення по послідовності окремих зображень, отриманих за допомогою магнітного резонансу (МР), дифузії контрастних агентів у печінці. Артефакти, пов'язані з рухами пацієнта, не дозволяють безпосередньо порівнювати зображення. Однак, у Грід-інфраструктурі можна за розумний час виконати розпаралелений аналіз послідовності зображень.
- SiMRI3D – моделювання на основі зображень, отриманих за допомо-

гою МР. Мета моделювання – одержати штучні, але цілком реалістичні тривимірні МР-зображення для аналізу зображень від добре відомих об’єктів, вивчення артефактів і подальшого розвитку й удосконалювання технології МР- зображень.

– gPTM3D – інтерактивне відновлення тривимірних медичних зображень, наприклад, зображень всього об’єму складних органів людини. В цьому разі вимоги до якості інтерактивної роботи сервісів такі, що деякі сайти в Грід-інфраструктурі повинні встановити високий пріоритет для таких завдань.
– Bronze Standard – оцінка алгоритмів одержання медичних зображень. Обсяг даних і вартість обчислень недоступні звичайним комп’ютерам, але в Грід-інфраструктурі це застосування працює досить легко.
– Пакет програмного забезпечення SPM застосовується в нейрологічних дослідженнях для ранньої діагностики хвороби Альцгеймера. У його основі лежить порівняння даних пацієнта з великим набором даних від людей без цієї патології. Грід-технології надають легкий доступ до розподілених даних і розподілених обчислювальних ресурсів.

По-друге – це сектор біоінформатики, який займається аналізом послідовностей генів. У сферу його інтересів входять геноміка, протеоміка й філогенія.

– GPS@ (Grid Protein Sequence Analysis – «Аналіз білкових ланцюжків на основі Грід-технологій») – веб-портал, який надає зручний інтерфейс із цими біоінформаційними ресурсами в Грід-інфраструктурі EGEE. Доступний також прототип цього порталу, де є інтерфейс з 13 програмами в Грід-інфраструктурі з 46 програм оригінального порталу.
– xmipp_MLrefine – тривимірний структурний аналіз великих макромолекулярних комплексів. У процесі відновлення структури комплексів спільно використовується безліч різних зображень досліджуваного зразка. Ці зображення, однак, часто страждають від сильного шуму і, як результат, для складання моделі, найбільш відповідній експериментальним даним, треба зробити багато ітерацій.

– Зображення, отримані на електронному мікроскопі, страждають різними формами аберації. Розходження між теоретичним і експериментальним зображенням-проекцією математично описується функцією контрастного переносу (ФКП). Xmipp_assign_multiple_CTFs (Micrographia Contrast Transfer Function calculation) – застосування, яке виконує моделювання для знаходження ФКП.
– SPLATCHE (SPatiaL And Temporal Coalescences in Heterogeneous Environment – «Просторово-часові інтеграції в різнорідному навколишньому середовищі») – застосування з стільниковою структурою для моделювання еволюції генома людини. Воно дозволяє відновити розселення людини по Землі в географічно правдоподібних ландшафтах і моделювати молекулярну розмаїтість різних людських популяцій.

Третя гілка застосувань підтримує роботу сектора розробки ліків і зосереджена на прискоренні пошуку нових ліків за допомогою комп'ютерного моделювання структури і динаміки білків.

WISDOM (Wide In Silico Docking On Malaria) – розрахунки, які вимагають значних комп'ютерних ресурсів, для пошуку ліків від захворювань, переможених у високорозвинених країнах, і від нових захворювань, які щойно поширюються. Мета цих розрахунків докінгу молекул – визначити, наскільки ефективно конкретні ліки приєднуються до певних ділянок вірусу-мішені – тобто, іде пошук ліків, у яких ділянка докінгу представляється найбільш ефективною проти вірусу. Успішними виявилися застосування для пошуку засобів від малярії й пташиного грипу; планується пошук ліків і від інших вірусів.

GridGRAMM – простий інтерфейс для веб-розрахунку докінгу молекул. Розрахунки включають оцінку якості докінгу і різні методи доступу до тривимірної структури комплексу. Молекулярний докінг може застосовуватися для вивчення міжмолекулярних взаємодій, вивчення взаємодій між ензимами і субстратом, розробки ліків і розуміння патологічних мутацій.

GROCK (Grid Dock) – нескладна веб-реалізація відбору визначених міжмолекулярних взаємодій з величезного обсягу інформації. Користувачі мо-

жуть по відомих структурах дослідити одну молекулу у купі цілої бази даних.

Застосування Drug Discovery (пошук ліків), яке працює в рамках EGEE, призначене для пошуку принципово нових ліків від масових захворювань, наприклад від малярії – хвороби, від якої щорічно страждають 300 мільйонів чоловік, а мільйон вмирає. Причому ситуація погіршується через підвищення стійкості хвороби до існуючих лікувальних препаратів. Застосування було ініційоване і реалізоване Інститутом алгоритмів і наукових обчислень ім. Фраунгофера SCAI (Fraunhofer Institute for Algorithms and Scientific Computing) у Німеччині й Лабораторією корпускулярної фізики (IN2P3) в Клермон-Феррані, Франція. Воно дозволяє підвищити можливості доступу фармацевтичних компаній і академічних дослідницьких інститутів до різноманітної, складної і розподіленої інформації про хвороби і забезпечити можливість спільних досліджень з пошуку нових ліків.

Застосування ґрунтується на можливості обчислювати ймовірність того, що нові потенційні ліки ввійдуть у прямий контакт із активною частиною одного з паразитних білків малярії. Зазвичай такі обчислення виконуються на кластерах ПК і обмежуються приблизно 100000 кандидатами на нові ліки. У сеансі масової обробки даних, проведеному в Грід-середовищі EGEE у серпні 2005 року, досліджено понад 46 мільйонів кандидатів. У сеансі одночасно використовувалося 1000 обчислювальних машин в 15 країнах по усьому світі. На одному персональному комп’ютері для виконання такої роботи треба було б 80 років. Успіх роботи продемонстрував, яку допомогу може надати Грід у дослідженнях з пошуку ліків, значно прискорюючи весь процес розробки.

Протягом квітня 2006 року лабораторії Азії і Європи, використовуючи крім EGEE ще й Грід-інфраструктури AuverGrid і TWGrid, провели спільну роботу з аналізу 300000 можливих компонентів ліків проти вірусу пташиного грипу H5N1. Метою роботи був пошук можливих сполук, які могли б придушувати активність ферменту на поверхні вірусу грипу. У роботі були задіяні 2000 комп’ютерів, які працювали протягом 4 тижнів. Якби працював лише один комп’ютер, йому знадобилося б 100 років для розв’язання цього завдання. В реляційній базі даних було створено і збережено більше 60000 ви-

хідних файлів з обсягом інформації 600 Гб. Зараз виконується ідентифікація і класифікація потенційних компонентів ліків проти пташиного грипу.

Наступні кроки в розвитку застосувань з пошуку нових ліків включають класифікацію великої кількості даних, з метою ідентифікувати потенційні ліки, використовувані в лікуванні ряду захворювань, і зменшити прірву між такими «віртуальними кандидатами» і традиційною розробкою ліків. Це може призвести до появи значного числа фізичних молекул-кандидатів для ліків, які надалі можуть доводитись до рівня реальних терапевтичних компонентів.

8.1.3 Роботи в галузі астрофізики

У рамках EGEE діють дві основні астрофізичні ВО: проект Planck Європейського космічного агентства в галузі обчислювальної астрофізики й проект MAGIC в галузі астрофізики елементарних частинок. Це різні наукові проекти, але обидві ВО займаються загальними проблемами обчислень, які відрізняються величезними масштабами пошуку, збору і зберігання даних і моделювання.

Planck – супутник Європейського космічного агентства запущений в 2008 р. Він проводить надзвичайно масштабне мікрохвильове картографування всього неба, як геометрично, так і по ширині діапазону частот. Також найвищим є рівень точності, стійкості і чутливості вимірів. Картографування проводиться мінімум двічі.

Planck оснащений мікрохвильовими і субміліметровими детекторами, які об’єднані в блок високочастотного обладнання – High Frequency Instrument (HFI) і блок низькочастотного обладнання – Low Frequency Instrument (LFI). Детектори охоплюють діапазон частот від 30 до 850 ГГц. Центр обробки даних від LFI обробляє близько 100 МБ стислих даних за добу.

Одне з основних завдань центра обробки даних – розробка і здійснення повномасштабного моделювання місії Planck для перевірки каналів обробки даних. Моделююче програмне забезпечення повинне імітувати процес картографування разом з усіма можливими джерелами систематичного впливу.

Більш того, воно повинне враховувати всі особливості неба в мікрохвильовому діапазоні. Це моделювання є фундаментальною перевіркою всієї інфраструктури, яка виконує аналіз даних, і основою вимог до апаратних засобів, які будуть вести їхню обробку.

MAGIC – телескоп для візуалізації випромінювання Черенкова в атмосфері; він розташований на Канарських островах і працює з кінця 2004р. Телескоп вимірює, а однойменне Грід-застосування виконує моделювання характеристики злив частинок, породжених у верхніх шарах атмосфери високоенергетичними космічними частинками – зокрема, гамма-променями. Для аналізу даних потрібне велике моделювання того, як високоенергетичні космічні частинки породжують зливи частинок в атмосфері. Перше випробування роботи інфраструктури EGEE з даними MAGIC почалося в березні 2005 р. Зараз будується другий телескоп на тій же ділянці Ла Пальми в 85 метрах від MAGIC.

Є ще одна ВО, яка інтенсивно експлуатує засоби EGEE. Колаборация ANTARES займається створенням великомасштабного підводного детектора черенковського випромінювання (проект NEMO), розташованого в глибині Середземного моря. Відповідне Грід-застосування оптимізовано для виявлення мюонів, які виникають при взаємодії високоенергетичних нейтрино із частками суши або води.

8.1.4 Експерименти і застосування для наук про Землю і геофізики

Науки про Землю охоплюють широкий спектр тем, пов’язаних із земною корою, атмосферою, океаном і їхньою взаємодією, а також різними шарами атмосфери і кори.

Двома співтовариствами представників наук про Землю і геофізики в EGEE підтримуються п’ять напрямків спеціалізованих застосувань для: науки про Землю, гідрології, спостереження за поверхнею землі, кліматології і фізики твердої Землі. Ці два близьких співтовариства діють через дві віртуальні організації: ESR (Earth Science Research – «Дослідження в галузі наук про Землю») для академічних установ і пов’язаних з ними учасників і

EGEODE (Expanding GEOsciences on DEmand – «Розширення кола інтересів наук про Землю на вимогу»), засновану у Франції приватною компанією CGG (Compagnie Generale de Geophysique). Віртуальна організація EGEODE підтримує перше промислове застосування EGEE.

У кожній галузі ESR розгорнуте мінімум одне застосування в EGEE і GILDA.

Галузь спостережень за поверхнею землі: супутник GOME вимірює характеристики озонового шару, а також використовує дані супутникового експерименту ERS/SAR з виявлення витоків нафти. Грід-інфраструктура допомагає вести аналіз характеристик озонового шару: вона забезпечує єдине обчислювальне середовище для різних стадій роботи і легкий загальний доступ «виробників» і «споживачів» до даних.

В галузі фізики твердої Землі роботи зосереджені на механізмах землетрусів і їхньому чисельному моделюванні у складних тривимірних геологічних моделях.

Спеціалізація першого застосування в галузі гідрології в інфраструктурі EGEE – вивчення зумовленого господарською діяльністю людини вторгнення морської води в прибережний водоносний шар у басейні Середземного моря. Робота застосування базуються на тривимірній моделі руху ґрунтових вод і переносу солі з урахуванням щільності середовища і виконується методом Монте-Карло.

В галузі кліматології виконується перенесення застосування для розрахунків повеней з експериментальної системи CrossGrid в середовище EGEE. У застосуванні виконується ціла низка процесів моделювання, на початку якої використовуються експериментальні дані. Для одержання результатів застосовуються метеорологічні, гідрологічні й гідравлічні моделі.

Першим застосуванням промислового рівня, яке успішно працює в режимі рутинної експлуатації сервісу в інфраструктурі EGEE, став Geocluster – промислове програмне забезпечення для обробки сейсмічних даних і дослідження складу шарів земної кори. Доступ до нього можливий через ВО EGEODE, яка веде дослідження в галузі наук про Землю як для державних,

так і приватних організацій, які займаються дослідженнями і розробками, а також для академічних лабораторій.

8.1.5 Вирішення питань ядерного синтезу засобами Грід

Наразі цілком очевидно, що Грід-технології здатні задовольняти практично всі вимоги досліджень в галузі ядерного синтезу. Тому в інфраструктурі EGEE працюють застосування, які вирішують такі завдання: простежування випромінювання на основі великої статистики для обчислення траєкторії мікрохвильових пучків у плазмі; дослідження кінетичного перенесення і оптимізації установок магнітного конфайнмента (стелараторів).

В інфраструктурі EGEE успішно розміщені кілька обчислювальних завдань, які стосуються проекту ITER (the International Thermonuclear Experimental Reactor – «Міжнародний експериментальний термоядерний реактор») – це міжнародний проект досліджень і розробок, мета якого – показати наукову й технічну можливість використання енергії термоядерного синтезу. У цьому реакторі здійснюється синтез двох різних ізотопів водню (дейтерію, у малих кількостях присутнього в природі, і тритію, штучного ізотопу) для одержання гелію із супутнім виділенням величезної кількості енергії. Реактор побудований у м. Кадараш у Франції. Воднева плазма в ньому втримується у формі тора при температурі понад мільйон градусів, що, у принципі, приблизно дорівнює потужності термоядерного синтезу на рівні 500 МВт.

Керуючий комітет (Steering Committee), який діє в рамках Європейської угоди з розробок в галузі термоядерного синтезу (European Fusion Development Agreement (EFDA)), заснував групу, завдання якої – вивчити перспективу потреб в обчислювальних ресурсах у європейського співтовариства дослідників в галузі ядерного синтезу. Уже показана спроможність Грід-інфраструктури задовольняти ці потреби. ВО співтовариства ядерного синтезу нараховує 11 сайтів у чотирьох федераціях учасників, що складає приблизно 1100 ЦП. Щоб розмістити в Грід-інфраструктурі нові застосування для досліджень ядерного синтезу, асоціаціям, які входять в EFDA, запропоновано

розмістити всі їхні програми і застосування в Грід-інфраструктурі EGEE.

Зараз у Грід-інфраструктурі EGEE працюють наступні застосування для досліджень ядерного синтезу:

- Massive Ray Tracing – Простежування траєкторії променів на основі великої статистики розраховує траєкторію мікрохвильового пучка в плазмі. Пучок, використовуваний для нагрівання плазми, моделюється групою променів (звичайно їх 105). Програма обчислює траєкторію і поглинання в складних видах плазми для кожного променя окремо.
- Kinetic Transport – Кінетичний перенос розраховує явища кінетичного переносу за допомогою відстеження орбіт великої кількості незалежних частинок, які перетерплюють зіткнення з фоновою плазмою, яка характеризується температурою, щільністю і електричним полем. Остаточні траєкторії використовуються для розрахунку важливих властивостей явищ переносу в плазмі – потоку частинок, потоку тепла, часів конфайнмента, асиметрій і вигляду функції розподілу частинок.
- Stellarator – застосування для оптимізації стеларатора – установки постійного режиму для магнітного втримання плазми, де відбувається термоядерний синтез і немає струмів. Для цього застосування розроблений власний унікальний алгоритм. Можливі різні конфігурації магнітного поля стеларатора, і треба вибрати найкращу. Кожна конфігурація досліджується в одному процесорі Грід; далі алгоритм застосування шукає найкращу серед них.

Крім того, окрему локальну віртуальну організацію заснували вчені, які займаються ядерним синтезом і працюють із ресурсами російського Грід для великих обсягів даних (Russian Data Intensive Grid (RDIG)). Наразі ця організація розробляє свої власні методики застосування Грід-технологій, формулює вимоги до ППЗ й розміщає в Грід-інфраструктурі свої перші застосування.

8.1.6 Грід для обчислювальної хімії

У Грід-інфраструктурі успішно розміщені й використовуються в режи-

мі нормальної експлуатації кілька застосувань, які вирішують наступні завдання: розрахунок спостережуваних величин у хімічних реакціях; моделювання молекулярної динаміки складних систем; розрахунок електронної структури молекул, молекулярних агрегатів, рідин і твердих тел.

Головним користувачем Грід-технологій в галузі обчислювальної хімії є апріорний молекулярний симулятор GEMS (Grid Enabled Molecular Simulator – «Молекулярний симулятор на основі Грід-технологій»). Для експлуатації застосування GEMS заснована ВО обчислювальної хімії CompChem. У Грід-інфраструктурі розміщені й працюють у режимі нормальної експлуатації ще кілька застосувань для обчислювальної хімії. Готується розміщення в інфраструктурі EGEE нових застосувань; вживаються заходи до розширення співробітництва між науковими групами, які працюють в галузі обчислювальної хімії.

Застосування GEMS використовується для моделювання динаміки реакцій у складних хімічних системах.

ABCtraj розраховує спостережувані величини атом-діатомної взаємодії в газоподібній фазі. Події генеруються методом Монте-Карло. Програма підключена до середовища молекулярної віртуальної реальності, де на монітори виводяться результати моделювання.

Venus розраховує перетини і коефіцієнти швидкості елементарних хімічних реакцій, моделюючи зіткнення атомів і молекул при початкових умовах, отриманих методом Монте-Карло. Для кожного зіткнення, від вихідних елементів до продуктів реакції, вирішуються рівняння Хемілтона, які описують рух атомів.

Застосування DI-Poly моделює молекулярну динаміку складних систем. Воно де-факто є стандартом в співтовариствах обчислювальної хімії і обчислювальної біології.

Застосування RWAVEP виконує квантово-механічні розрахунки ймовірності різних хімічних реакцій, застосовуючи підхід, оснований на понятті хвильового пакета. Для різних наборів вихідних станів генеруються різні події.

У найближчому майбутньому віртуальна організація CompChem планує розгорнути нові застосування, наприклад:

– COLUMBUS – комплект програм для складних розрахунків «з першооснов» електронної структури молекул. Програми призначені, головним чином, для розширених багатопосилальних розрахунків основного і збудженого станів електронної оболонки атомів і молекул.
– GAMESS – програма для виконання «з першооснов» розрахунків в галузі квантової молекулярної хімії, вона дозволяє обчислювати хвильові функції у вигляді антисиметрізованого добутку молекулярних орбіталей. Можливі наступні кореляційні виправлення до цих хвильових функцій: урахування залежності взаємодії від конфігурації; виправлення на основі теорії збурювань другого порядку; виправлення на основі уявлень про спарені кластери; можливо також наближення в рамках функціональної теорії щільності.

Крім того, віртуальна організація CompChem, з метою створення інтерфейсу з Грід, який настроюється користувачем, буде експериментувати із системою CHARON, яка відповідає специфічним вимогам співтовариства обчислювальної хімії.

8.1.7 Грід для e-science і інші застосування

В EGEE вітаються заявки на розміщення нових застосувань. Так останнім часом в системі з’явилась велика кількість нетрадиційних застосувань про які раніш навіть й мови не йшло.

8.1.7.1 Грід і цифрові бібліотеки

Проект GRACE (GRid enabled seArch and Categorization Engine) – Грід-система пошуку й категоризації, завершений у лютому 2005. GRACE дав проекту EGEE конкретний і швидкий зворотний зв’язок з використання Грід, а також сприяв поширенню знань про EGEE.

8.1.7.2 Фінансові застосування в середовищі Грід

Робота над фінансовими застосуваннями включає співробітництво з Міжнародним центром теоретичної фізики ім. Абдуса Салама (The Abdus Salam International Centre for Theoretical Physics – ICTP), який розгорнув італійську національну Грід-інфраструктуру для фінансово-економічних досліджень у рамках проекту Egrid, фінансованого міністерством освіти і науки Італії.

8.1.7.3 Створення мультимедіа засобами Грід

В інфраструктурі EGEE ця галузь застосувань – одна з найновітніших. Вона тут тільки починає свою діяльність. Мультимедійні застосування тестувались на системах налагодження у віртуальній Грід-лабораторії GILDA проекту EGEE.

8.1.7.4 Поширення Грід на промислові застосування

В EGEE успішно працюють кілька промислових застосувань: від обробки сейсмічних даних до застосувань на основі SME у біотехнології і виробництві пластиків. Вони відкривають доступ до колосальних ресурсів Грід-інфраструктури EGEE.

Необхідно відзначити, що завданню залучення виробничих застосувань у рамках проекту EGEE приділяється велика увага. Досягненню мети цього завдання сприяють Виробничий форум EGEE (Industry Forum), Група виробничих завдань (Industry Task Force) і програма EGEE з роботи з бізнес-партнерами (EGEE Business Associate), які мусять зробити відкриту для виробництва інфраструктуру і ноу-хау Грід EGEE. Організації співробітничають із технічними радами і напрямками EGEE нарівні із представниками наукових застосувань.

Один зі способів прогнозу виверження вулканів ґрунтується на перетворенні геофізичної інформації про поведінку вулканів на звукові хвилі. По-

тім картина звукових хвиль – своєрідна «музика» вулканів – аналізується, і це дозволяє зробити прогноз поведінки вулканів у близькому майбутньому, у тому числі прогнозувати виверження. У співробітництві із проектом EELA (Європейсько-Латиноамериканська інфраструктура), EGEE надає мережні і обчислювальні ресурси для такого аналізу поведінки вулканів Етна (на Сицилії) і Тунгурахуа (у Еквадорі).

8.1.7.5 Грід-застосування в галузі археології

Завданням застосування ArchaeoGRID – за допомогою Грід-інфраструктури EGEE і розподілених систем ряду інших споріднених проектів (зокрема, проекту створення розподілених електронних сховищ даних DILIGENT) здійснювати комплексний аналіз складних і різнорідних археологічних даних, отриманих при розкопках і польових дослідженнях. Ці дані можуть відноситися до фізики, хімії, науки про Землю, біології, географії, антропології, соціології… З розвитком археологічних методів і теорій розвиваються також способи одержання даних. Крім того, археологічні дані втрачають частину свого значення, коли розглядаються поза початковим просторово-часовим контекстом, у якому вони були отримані. Цей просторовий і часовий контекст сучасних археологічних досліджень простирається від місця розкопок до величезних регіонів і охоплює тривалі періоди часу. Заключна стадія археологічної роботи також дуже складна, вона включає візуалізацію результатів (цифрові карти, віртуальні навколишні середовища, двовимірні й тривимірні зображення і так далі) і текстовий опис, оснований на аналізі результатів і теоретичних знань.

Структура ArchaeoGRID поєднує ці різні підходи і дані в єдину систему, зручну для аналізу (який також планується проводити з використанням Грід-ресурсів).

8.2 Контрольні запитання і завдання для самостійної роботи

1. Поясніть такі нові терміни, як e-Science, e-Health, e-Commerce, охарактеризуйте застосування, які виконуються в Грід, назвіть основні напрямки використання Грід.
2. Охарактеризуйте світову Грід-систему Enabling Grids for E-sciencE (EGEE). Хто є основними користувачами цієї Грід-інфраструктури?
3. Розкажіть про експерименти на Великому Адронному Колайдері в сфері фізики високих енергій. Які Грід-застосування використовуються в цих експериментах?
4. Проаналізуйте Грід-застосування для фізики високих енергій окрім експериментів на Великому Адронному Колайдері.
5. Розкажіть про Грід-застосування в галузі біомедицини.
6. Які Грід-застосування в Грід-системі EGEE підтримують роботи в галузі астрофізики?
7. Які Грід-застосування задіяні в Грід-системі EGEE для обслуговування експериментів в галузі наук про Землю і в геофізиці?
8. Розкажіть про вирішення питань ядерного синтезу засобами Грід.
9. Що Ви можете сказати про Грід-застосування в галузі обчислювальної хімії?
10. Назвіть і стисло опишіть використання Грід-застосуваннь для e-science і для інших напрямків в науці і техніці.

Навчальне електронне видання

РОЛЬЩИКОВ ВАДИМ БОРИСОВИЧ

ТЕХНОЛОГІЇ РОЗПОДІЛЕНИХ СИСТЕМ ТА ПАРАЛЕЛЬНИХ ОБЧИСЛЕНЬ ГРІД-ТЕХНОЛОГІЇ. ЗМІСТОВНИЙ МОДУЛЬ №3

Конспект лекцій

Видавець і виготовлювач
Одеський державний екологічний університет
вул. Львівська, 15, м. Одеса, 65016
тел./факс: (0482) 32-67-35
E-mail: info@odeku.edu.ua
Свідоцтво суб'єкта видавничої справи
ДК № 5242 від 08.11.2016