

СИСТЕМА КЕРУВАННЯ ПОТОКАМИ ОБЧИСЛЕНЬ НА ОСНОВІ СТАНДАРТУ WS-BPEL ДЛЯ РОЗВ'ЯЗАННЯ ІНЖЕНЕРНИХ ЗАДАЧ

© Петренко А., Булах Б., 2012

Запропоновано підхід до створення систем керування обчислювальними сценаріями з використанням веб-сервісів та застосування стандартної мови опису їх взаємодії WS-BPEL для організації автоматизованого виконання інженерних та наукових обчислень. Досліджено окремі архітектурні рішення запропонованої BPEL-орієнтованої системи.

Ключові слова: веб-сервіси, керування потоками робіт, грід-обчислення.

The approach for computing workflow management systems development using web services and WS-BPEL orchestration description standard for automated execution of engineering and scientific computations is proposed. Particular architecture solutions of the proposed BPEL-oriented system are studied.

Key words: web services, workflow management, grid computing.

Вступ

Показовим прикладом сучасного масштабу наукових обчислень можна вважати аналіз потоків експериментальних даних з Великого адронного колайдера, де для розподіленої обробки десятків петабайтів даних на рік за допомогою грід-технологій залучаються сотні високопродуктивних обчислювальних ресурсів [1]. Можна із впевненістю стверджувати, що сучасні числові експерименти відрізняє складність процесів, що моделюються, надвеликі об'єми обчислень, інтеграція апаратних засобів збирання даних, міжорганізаційний та міждисциплінарний масштаб. Внаслідок цього істотно зростає складність систем наукових та інженерних обчислень, які повинні володіти достатніми можливостями з адаптації до динамічних умов: варіювання мети та сценарію, змін у підходах, апаратних та програмних засобах виконання обчислень тощо.

Адаптаційні можливості таких систем можуть бути поліпшені за умови використання workflow-концепції (буквально – потоків робіт) [2], ідея якої полягає у декомпозиції загальної обчислювальної задачі у послідовність окремих обчислювальних етапів. Таке представлення обчислювального сценарію, доступне для редагування користувачем-дослідником або ж автоматизованими засобами, надасть програмній системі гнучкості та розширить спектр її можливостей, перетворивши її на систему управління обчислювальними сценаріями. Розвиток веб-технологій робить можливими розроблення та впровадження систем проектування та виконання потоків обчислень, які є важливою частиною сучасних наукових та інженерних досліджень, з високим ступенем автоматизації.

Постановка задачі

Метою статті є аналіз підходів до побудови сучасних систем управління обчислювальними сценаріями, які орієнтуються на сучасні стандарти веб- та грід-технологій. Головну увагу приділено розгляду BPEL-стандарту та сумісного інструментарію як ключових компонентів для організації потоків обчислень з веб-сервісів. Актуальною лишається необхідність адекватно оцінити придатність цього підходу до виконання інженерних та наукових обчислень, проаналізувати сильні та слабкі сторони запропонованого рішення в контексті специфіки задач моделювання та аналізу даних, визначити шляхи розв'язання виявлених неузгодженостей.

Аналіз існуючих рішень

Системи управління сценаріями обчислень мають певні переваги порівняно із спеціалізованими системами із фіксованим обчислювальним сценарієм. Деякі з переваг від оперування потоками робіт як абстракцією для гнучкого обчислювального сценарію наведені нижче:

- можливість динамічного коригування обчислювальним процесом (зокрема – безпосередньо під час його виконання) залежно від рішень дослідника або у відповідь на появу нових даних;
- широкі можливості з повторного використання готових розробок (компонентів потоку робіт, самих потоків як основи для інших потоків або їх складових) для розв’язання нових задач;
- можливість залучення компонентів від сторонніх розробників завдяки стандартизованому інтерфейсу компонентів у потоці робіт, що спрощує колективну підтримку системи та вказує простий шлях розширення її функціональності;
- можливість абстрагування від спеціалізації окремих компонентів побудови сценаріїв, що дає змогу створювати системи для міждисциплінарних досліджень;
- можливість попереднього моделювання та аналізу довільного спроектованого сценарію перед його виконанням завдяки визначеній семантиці потоку робіт;
- можливість автоматичного компонування сценарію за абстрактним описом мети експерименту за допомогою семантичного опису складових компонентів потоку робіт.

Слід зазначити, що ці перспективні можливості в існуючих сьогодні системах реалізовані із різним ступенем успішності (наприклад, підтримка автоматизованого складання сценаріїв все ще не реалізована на належному рівні). Платою за гнучкість та адаптованість сценаріїв є відносно гірша продуктивність подібних розподілених систем порівняно з жорстко зв’язаними монолітними спеціалізованими рішеннями, оскільки забезпечення універсальності системи та слабкої зв’язаності компонентів потоку часто призводить до надлишкових передавань даних між компонентами.

З часу появи перших успішних спроб реалізації концепції workflow-систем для наукових задач пройшло близько десяти років. У витоків нинішніх розробок стояли такі системи, як Discovery Net [2]. Цей проект мав на меті насамперед вирішення проблем «наук про життя», моніторингу геонебезпек, моделювання навколишнього середовища, пошуку відновлювальних джерел енергії. З часом, завдяки вдалим архітектурним рішенням, що не залежали від профілізації системи, коло застосувань розширилося до біоінформатики, хімії, і навіть фінансів та бізнесу. Насамперед це міждисциплінарне програмне рішення мало слугувати інструментом аналізу даних (в т.ч. т.зв. «інтелектуального» – data mining), зібраних від численних постачальників – пристроїв та ресурсів, об’єднаних мережею Інтернет або глід.

Архітектура системи – багатоланкова, верхній рівень якої становить клієнтське програмне забезпечення для швидкого розроблення потоків робіт із графічним редактором сценарію. Проміжна ланка представлена сервером потоків, відповідальним за виконання робочих потоків, авторизацію та узгодження потоків із залученими гетерогенними ресурсами, які становлять останню ланку архітектури. Визначено внутрішні формати для робочих потоків та їх компонентів, що дає змогу розгортати цілий потік як окремих компонентів, готовий до використання. Залучення метаданих для кожного компонента, що описують типи його вхідних та вихідних даних, дає змогу верифікувати потоки на обов’язковий збіг типів з’єднаних входів та виходів (жорстка типізація).

Ці архітектурні рішення виявились вдалими та часто запозичувались проектантими-послідовниками. Серед успішних нині діючих міждисциплінарних workflow-середовищ слід назвати Triana Workflows [3] та Kepler [4]. Обидві системи пропонують:

- зручний графічний інтерфейс користувача із редактором графів потоків робіт;
- багату бібліотеку стандартних компонентів для різних дисциплін та чималу кількість додаткових компонентів, розроблених у межах інших проектів;
- власний диспетчер виконання сценаріїв та засоби моніторингу.

Обидві системи розроблені у вигляді вільно поширюваної Java-програми, яку користувач має завантажити, встановити на своєму комп’ютері та періодично оновлювати. Рішення відрізняються за підходами до організації координованого виконання сценаріїв (Triana використовує модель

обчислень, де послідовність виконання неявно визначається маршрутами передавання даних, які проектує користувач, а у Kepler взагалі доступні на вибір кілька різних моделей обчислень для аналізу даних та моделювання). Але, найголовніше, подібні системи не сумісні ні на рівні диспетчера виконання, ні на рівні опису потоків, ні навіть на рівні опису компонентів.

Водночас одним із відомих підходів до створення сумісних розподілених систем є сервісно-орієнтована архітектура (COA), серед головних принципів якої можна назвати декомпозицію на слабо зв'язані функціональні одиниці (сервіси), забезпечення здатності сервісів до поєднання (композиції), функціональна сумісність завдяки стандартному опису (контракту) сервісів. Якщо базові компоненти вищезгаданих систем управління потоками робіт не можна вважати повноцінними сервісами, то веб-сервіси є незалежною від платформи реалізацією принципів COA, що ґрунтується на стандартному описі інтерфейсу (WSDL), забезпеченні взаємодії через протокол обміну XML-повідомленнями стандартної структури (SOAP) та стандартним механізмом реєстрації та пошуку сервісів (UDDI).

Заради справедливості слід зазначити, що Triana та Kepler мають компоненти-адаптори для роботи з веб-сервісами, однак більш вигідним може виявитись принцип прямого використання веб-сервісів як компонентів потоку робіт. Прикладом повноцінної workflow-системи, яка ґрунтується на стандартах веб-сервісів, є середовище Taverna [5] – складова проекту розвитку е-науки myGrid, виконавцями якого є ряд британських наукових закладів. Первинною метою розроблення системи було задоволення потреб спільноти біоінформатиків в інструменті для побудови робочих потоків з численних віддалених веб-сервісів.

Отже, ця система прямо орієнтована на використання веб-сервісів як стандартного відкритого інтерфейсу компонентів, не нав'язуючи власний внутрішній опис, завдяки чому коло застосувань системи було швидко розширене. Поряд із можливістю імпорту та використання віддалених веб-сервісів, бібліотека компонентів також містить і локальні Java-компоненти для виконання допоміжних задач. Можливе віддалене керування виконанням потоків робіт через сервер.

Перераховані вище системи часто називають системами управління «науковими потоками робіт» на протизагу системам управління «бізнес-процесами». Бізнес-процес у загальному розумінні є послідовністю дій (виконуваних як людьми, так й інформаційними системами) для досягнення певної мети (виробництва «продукту» або надання «послуги») для певної групи споживачів. Бізнес-процес може бути реалізований програмно, зокрема у межах парадигми COA та технології веб-сервісів. Причому, останній підхід успішно використовується у розподілених системах промислового масштабу (таких, як системи управління виробництвом чи управління продажами рівня підприємства). Були розроблені і набули поширення відкриті стандарти на опис сценаріїв композиції сервісів (такі, як BPEL – Business Process Execution Language, тобто мова виконання бізнес-процесів) та відповідний цим стандартам інструментарій (диспетчери виконання BPEL-процесів, або BPEL-engines). Застосування цих напрацювань у системі управління інженерними та науковими обчислювальними сценаріями може виявитись вдалим вибором при створенні інтегрованої системи, що максимально використовує переваги від використання відкритих стандартів. Нижче розглянемо окремі аспекти цього підходу: загальну архітектуру системи, WS-BPEL як мову опису обчислювальних сценаріїв, аналіз потоків робіт, специфіка виконання BPEL-процесів.

Загальна архітектура системи

Очевидно, що задачі управління підприємством та проведення числових експериментів мають свою специфіку, хоча обидві можуть описуватись багатокроковими сценаріями [6].

Основна задача моделювання за допомогою бізнес-процесів – виробити узагальнене бачення процесу, що має справу з багатьма різними учасниками та інформаційними системами. Будучи ретельно розробленою та погодженою, модель бізнес-процесу може бути реалізована на програмному рівні та включена на постійній основі в систему управління. Для потоків робіт, скажімо, математичного моделювання більш характерним є акцент на разовому виконанні: замість послідовності життєвого циклу «проектування, впровадження, використання» маємо «проектування, запуск, аналіз результатів». При цьому слід зазначити, що тривалість виконання одного запуску наукового потоку може легко сягати тижнів, навіть за умови залучення високопродуктивних ресурсів.

Бізнес-процеси виконують дії, призначені для досягнення конкретного результату (кінцева мета процесів типу «замовлення товару» очевидна і конкретна), тоді як наукові потоки робіт є сценаріями експериментів, часто із наперед не відомим результатом. Це означає, що останні модифікуються набагато інтенсивніше, причому часто є потреба скоригувати хід експерименту просто під час його проведення. Найбільшу цінність наукових потоків становить результат числового експерименту (врховуючи супутню статистичну інформацію щодо процесу обчислень), тоді як у бізнес-процесах цінується надійність виконання поставленої задачі.

Відрізняються у цих підвидів робочих потоків і групи користувачів. Бізнес-процеси на абстрактному рівні розробляються менеджерами та бізнес-аналітиками, однак їх конкретизована програмна реалізація – справа їх колег, IT-спеціалістів та програмістів. Наукові робочі потоки призначені для того, щоб спростити процес організації обчислень для інженерів та науковців, які часто не є спеціалістами у програмуванні та адмініструванні та не мають залежати від таких спеціалістів. Тож системи управління такими робочими потоками мають бути достатньо дружніми для користувачів, приховуючи усі низькорівневі деталі роботи із програмними інтерфейсами та протоколами.

При моделюванні бізнес-процесів семантика переходів від кроку А до кроку Б визначається так [6]: Б може бути запущений лише після того, як буде виконано А. Тобто бізнес-процеси зосереджені на потоці управління, а потік даних є вторинним (питання передавання даних описуються окремо). Для науково-інженерних сценаріїв зазвичай саме маршрути передавання даних визначають послідовність виконання кроків, тобто первинним є саме потік даних: Б споживає дані від А – отже, має бути запущений після А. Такі сценарії є «конвеєрами» з обробки даних (часто – надвеликих об'ємів) із відносно простою конфігурацією потоку управління (як правило – ациклічний напрямлений граф), на відміну від бізнес-процесів, які зосереджені на складній поведінці та взаємодії акторів і мають підтримувати складні конструкції потоку управління для опису дій у певних ситуаціях (наприклад, транзакції). З іншого боку, інженерним та науковим сценаріям властиві свої специфічні шаблони, такі як одночасний паралельний запуск великого числа екземплярів сценарію з різними вхідними параметрами.

Отже, завдяки своїй специфіці, засоби управління бізнес-процесами надають можливість створювати достатньо складні сценарії виконання обчислень та розгортати їх як сервіси для спільного та повторного використання. З іншого боку, для успішної адаптації цих засобів для виконання інженерних та наукових потоків робіт слід додатково продумати організацію маршрутів передавання даних, підтримку зручного механізму редагування і повторного виконання сценаріїв, підтримку надійного виконання надтривалих операцій, можливість залучення грид-обчислень.

Перш ніж переходити до розгляду конкретних архітектурних рішень, слід визначити узагальнені на основі вищевикладеного вимоги до системи управління інженерними та науковими обчислювальними сценаріями:

- вимоги до *інтерфейсу користувача*: графічний редактор потоку, що приховує внутрішній опис моделі потоку; дружній, звичний, інтуїтивно зрозумілий пересічному користувачу інтерфейс із достатньою виразною потужністю;
- вимоги до *програмних інтерфейсів*: підтримка ряду стандартних відкритих форматів опису потоків, інтерфейсів складових компонентів та форматів наукових даних, сумісність із поширеним програмним забезпеченням; можливість залучення компонентів від сторонніх розробників; підтримка виконання розподілених та грид-обчислень;
- вимоги до процедури *виконання* сценаріїв: підтримка довготривалих розрахунків; можливість призупиняти сеанс роботи, не перериваючи обчислень; забезпечення надійності виконання (належної обробки виключних ситуацій); ведення журналу та накопичення супутньої статистики;
- вимоги до супутнього *інструментарію*: відкриті бібліотеки складових компонентів для сценаріїв, з однієї або кількох предметних областей (в ідеалі – незалежність системи від конкретної області застосування); модулі візуалізації результатів виконання потоків; засоби діагностики, відлагодження, перевірки (верифікації) створених потоків; засоби, що сприяють колективній роботі.

Запропоновану загальну архітектуру системи ВРЕЛ-орієнтованої системи управління інженерними та науковими обчислювальними сценаріями наведено на рис. 1. Їй притаманні усі основні складові workflow-систем: редактор, набір компонентів потоку, диспетчер виконання.

Однак архітектурні деталі визначаються використанням мови WS-BPEL для внутрішнього опису сценаріїв та BPEL-engine як диспетчера виконання.

Сервер виконання сценаріїв

Більшість систем управління сценаріями обчислень (приклади розглянуті вище) є або локальними програмами, або ж клієнт-серверними рішеннями. Вади локальної версії відомі: вимкнення машини означатиме переривання/призупинку експерименту; користувачі мають власноруч встановлювати програмне забезпечення та слідкувати за необхідністю його оновлення; розробники ж мають турбуватися про підтримку користувачів попередніх версій. Перший з цих недоліків усувається за допомогою клієнт-серверної архітектури: контроль за безперервним виконанням потоку переноситься на сервер. Там же розгортаються підсистеми безпеки та обліку користувачів.

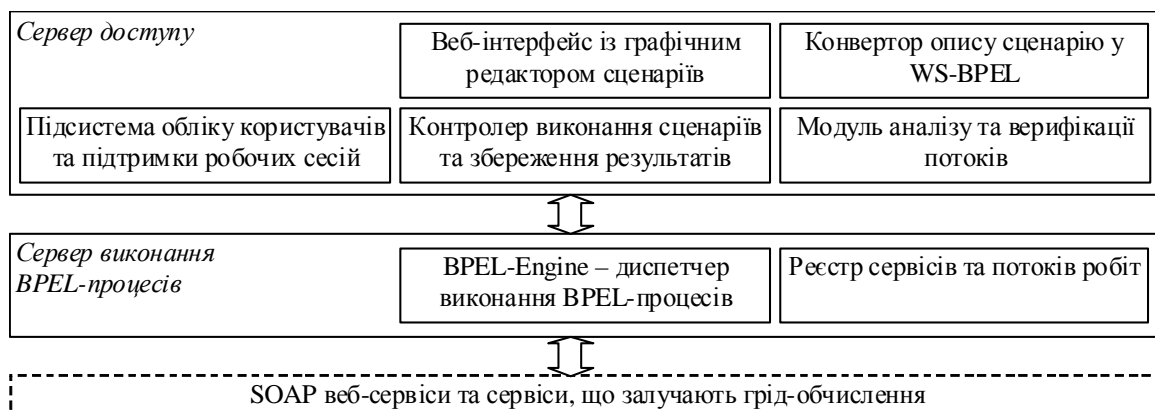


Рис. 1. Основні складові архітектури BPEL-орієнтованої системи управління обчислювальними сценаріями із залученням грид-обчислень

Інтерфейс користувача

Решту з названих щойно проблем усуває веб-інтерфейс. Сучасний рівень розвитку засобів для побудови «розвинутих Інтернет-додатків» (RIA – Rich Internet Applications) дає змогу успішно реалізовувати веб-інтерфейс і для складних графічних редакторів. За цим підходом можна організувати доступ користувачів до функціональності системи через єдиний веб-портал, що усуває проблеми із підтримкою версій. З іншого боку, такий підхід вимагатиме наявності доступу до мережі Інтернет не лише під час виконання потоків, а й на фазі їх розроблення, що не завжди є доцільним та зручним, проте є виправданим з огляду на те, що ця система управління сценаріями обчислень орієнтована на використання віддалених веб-сервісів як складових потоку робіт.

Диспетчер виконання сценаріїв

Головним компонентом сервера виконання є диспетчер, орієнтований на роботу з BPEL-описами обчислювальних сценаріїв. Використання стандартної мови, якою є WS-BPEL, зводить до мінімуму залежність системи від конкретної реалізації BPEL-диспетчера (яких є чимало, зокрема відкритих). Складовою архітектури системи є конвертор графічного опису потоку, створеного у редакторі, у WS-BPEL-опис взаємодії веб-сервісів. Бажаним також є аналізатор опису на наявність логічних вад та оцінки ефективності спроектованого сценарію.

Реєстр сервісів

Оскільки компонентами робочого потоку є веб-сервіси, то і роль бібліотеки компонентів відіграє реєстр веб-сервісів. Стандартним рішенням є UDDI-реєстр, проте йому бракує стандартних можливостей для семантичного опису веб-сервісів, які б доповнювали синтаксичний WSDL-опис інтерфейсу для забезпечення розширених можливостей із автоматизованого пошуку сервісів та їх компонування.

Хоча структури даних UDDI все ж можуть слугувати для штучного додання семантики в реєстр. Іншими словами, можна «прив'язати» елементи онтології веб-сервісів до структур UDDI. Тобто, існує можливість створення модулів семантичного пошуку сервісів. У [7] пропонується підхід, який полягає у розширенні структури даних UDDI businessService додатковим елементом serviceProfile, який вказуватиме на збережену онтологію сервісу. Можливо подолати обмеженість синтаксичного пошуку UDDI за рахунок «обгортання» інтерфейсу UDDI у компонент-брокер, що підтримує семантичний пошук, та, паралельно із внутрішньою переадресацією запитів до UDDI, опитує базу онтологій [8].

WS-BPEL як мова опису обчислювальних сценаріїв

Оскільки сьогодні не існує загальноприйнятих стандартів на мови опису інженерних/наукових обчислювальних сценаріїв, то як мову опису потоків робіт, складених з веб-сервісів, доцільно прийняти один із стандартів опису взаємодії веб-сервісів взагалі. Найпопулярнішою сучасною мовою цього класу є WS-BPEL версії 2.0 [9] – стандарт OASIS на оркестровку веб-сервісів (з англ. service orchestration – централізоване кероване виконання вервісів на противагу децентралізованому «choreography»-підходу). Цей стандарт має чималу передісторію та розроблений з урахуванням накопиченого досвіду використання своїх попередників: WSFL, XLANG, BPML, BPEL4WS [10].

Оскільки цей стандарт орієнтований на опис бізнес-процесів, складених з веб-сервісів, то технічним деталям виклику сервісів, обробки помилок, передач повідомлень, роботі з WSDL-описами приділяється значна увага, тоді як питань визначеності результату описаного процесу, аналізу його логічної коректності стандарт майже не торкається. Тому аналізатори BPEL-опису можуть легко перевіряти лише синтаксичну коректність, але не логічну.

Потік керування

XML-мова WS-BPEL більшою мірою орієнтована на опис потоку керування, що звично для мов опису бізнес-процесів. Сценарій у термінах мови так і називається – процесом (process). Перелік видів визначених базових «дій» (у термінах мови – activity) невеликий: прийом зовнішнього запиту (receive) та відповідь на зовнішній запит (reply), виклик веб-сервісу (invoke), присвоювання значень змінних (assign) та дії з генерації виключень, очікування, виходу. Receive/reply використовуються для організації взаємодії з процесом (наприклад, процес завжди розпочинається з операції receive та повертає результати роботи через операцію reply). Конструкція invoke позначає у сценарії місце виклику операції віддаленого веб-сервісу.

Ці базові дії можуть бути впорядковані у потік кількома способами. Перший спосіб, придатний для добре структурованих сценаріїв, полягає у використанні конструкцій sequence (послідовне виконання), if (вибір за умовою), while/repeatUntil (цикл за умовою), forEach (ітерації). Конструкції можуть «вкладатися» одна в одну. В цьому випадку весь сценарій являє собою сукупність послідовних гілок, умов та циклів. Для забезпечення можливості організації паралельних гілок виконання та побудови повноцінного графу потоку управління визначена ще одна конструкція – flow (потік). За її допомогою усі дії можуть бути впорядковані на основі їх взаємозалежностей (links): кожна дія всередині потокової конструкції має набір вхідних та вихідних залежностей, що пов'язують її з іншими діями у причинно-наслідкову послідовність.

Тобто, конструкція flow дає змогу відобразити граф сценарію, створений у графічному редакторі, мовою BPEL: вершини графу (кроки сценарію) позначаються «діями» (activities), а напрямлені дуги графу позначаються «залежностями» (links). Однак є одне суттєве обмеження мови WS-BPEL: залежності можуть бути активовані лише один раз, унеможливаючи циклічні графи. Це створює певні проблеми з перекладу графу мовою BPEL, оскільки змушує запроваджувати перевірку на замкнуті шляхи у графі опису сценарію, та використання конструкцій while/repeatUntil поруч із flow для уможливлення циклічного виконання дій.

Потік даних

Веб-сервіси взаємодіють через обмін XML-повідомленнями, що є їх перевагою і недоліком одночасно. З одного боку, веб-сервіси не прив'язані до конкретних апаратно-програмних платформ, а типи даних легко перевіряються та зіставляються завдяки механізму XML-schema. З іншого боку, використання XML негативно позначається на продуктивності взаємодії через істотні накладні витрати на парсинг повідомлень, щільно «запакованих» у XML-елементи (що також збільшує об'єм самих повідомлень, які пересилаються мережею, порівняно з корисними даними).

Мова WS-BPEL пропонує єдиний спосіб організації потоку даних між сервісами – використання змінних для збереження та передавання даних. Змінні із вхідними XML-повідомленнями передаються конструкції виклику сервісу, результат виклику повертається також у вигляді змінної із вмістом вихідного повідомлення. Допускається редагування змінних (копіювання даних) за допомогою конструкції assign.

Такий стан речей зумовлює цілу низку незручностей. По-перше, за умови використання лише синтаксичного WSDL-опису інтерфейсу сервісів користувач змушений власноруч поєднувати елементи вхідних та вихідних повідомлень, які можуть бути достатньо складними за структурою.

Виходом може бути запровадження додаткових семантичних метаданих та максимальне спрощення структур даних, що передаються.

По-друге, постає проблема організації ефективного передавання файлів великих об'ємів, що не є рідкістю для інженерних чи наукових застосувань. Накладні витрати на передачу даних через XML-повідомлення та централізований характер організації обчислень (обмін даними через сервер-диспетчер) змушують шукати інших шляхів для організації потоків файлових даних. Як варіант рішення – використання окремих сервісів організації передач файлів між віддаленими хостами, на яких розгорнуті обчислювальні веб-сервіси (аналогічних до сервісу Reliable File Transfer з пакета проміжного програмного забезпечення грид Globus Toolkit 4, що здатен управляти передаванням файлів між двома віддаленими машинами за ефективним протоколом передачі файлів gridFtp).

Автоматизовані засоби розробки

Мова WS-BPEL занадто складна для її безпосереднього використання дослідниками в середовищі проектування обчислювальних сценаріїв. Чимало комерційних програмних платформ управління бізнес-процесами пропонують графічні редактори для створення WS-BPEL-описів процесів. Такий редактор доступний і для відкритого середовища розробки Eclipse. Однак подібні рішення все ж не придатні для використання науковцями чи інженерами, оскільки зосереджуються на графічному представленні усіх елементів WS-BPEL, що робить схему сценарію занадто перевантаженою технічними деталями та інтуїтивно незрозумілою для неспеціалістів.

Замість використання редактора мови BPEL пропонується альтернативний підхід, який полягає у розробленні: а) редактора потоків робіт, який би дав змогу користувачам швидко будувати інтуїтивно зрозумілий граф потоку з невеликої кількості базових примітивів, б) перекладача структури графу мовою WS-BPEL.

Запропонований підхід був втілений у прототипі робочого середовища, розробленого згідно з проектом «Міждисциплінарний комплекс оптимального математичного моделювання в грид-середовищі з автоматичним формуванням і вирішенням рівнянь відповідних математичних моделей». Редактор потоків, орієнтований на специфіку задач проектування, реалізовано на JavaScript, що дає змогу використовувати як клієнт простий веб-браузер без необхідності встановлення додаткових модулів. Спроектвані та перекладені мовою WS-BPEL 2.0 сценарії запускалися на виконання за допомогою відкритого BPEL-engine OW2 Orchestra 4.8. Приклад розробленого графічного інтерфейсу наведено на рис.2. Замість роботи із сервісами, користувач оперує етапами та процедурами маршруту проектування. Перелік застосованих правил перекладу графу потоку робіт на елементи мови WS-BPEL наведено в таблиці.

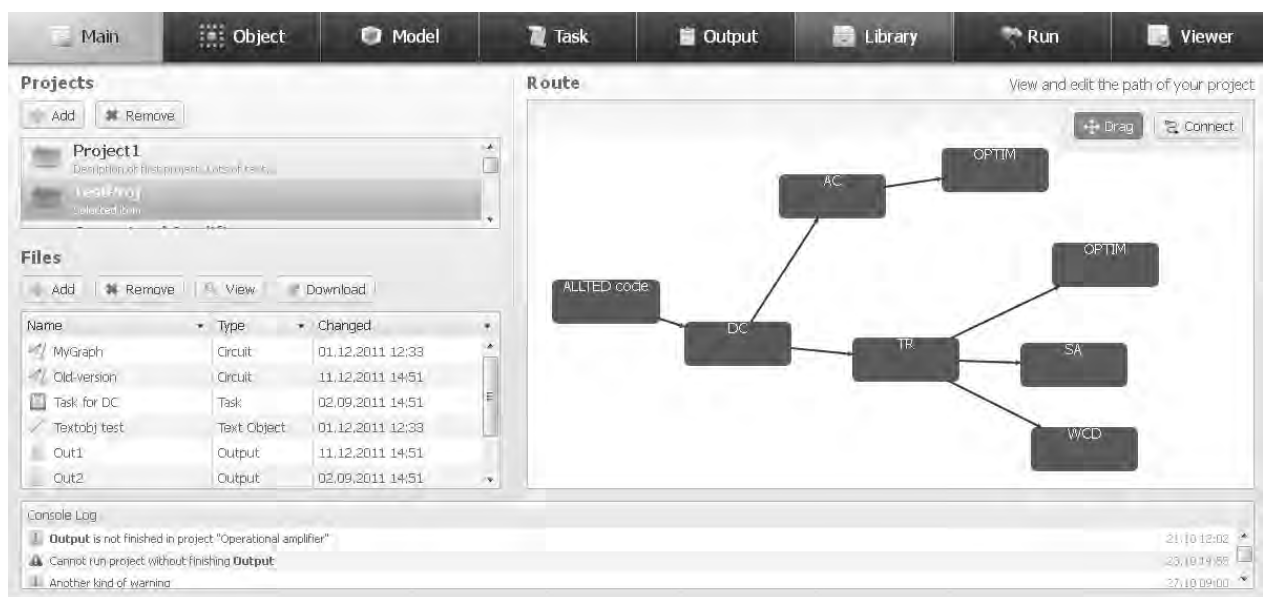


Рис. 2. Фрагмент інтерфейсу редактора маршрутів системи проектування

Примітиви опису обчислювального сценарію та їх переклад на мову WS-BPEL

Елементи графу потоку робіт	Відображення на WS-BPEL 2.0
Повний граф потоку	Конструкція flow
Виклик операції	Конструкція invoke із вхідними та вихідними змінними. Для асинхронного виклику – конструкція receive
Виклик довготривалої операції	Для реалізації довготривалих операцій використовується під-потік flow з ініціюючим викликом операції, періодичним опитуванням стану та викликом операції, що повертає результати
Перехід	Елемент link
Передача даних	Конструкція assign. Також застосовується для ініціалізації змінних

Аналіз потоків робіт

Засоби для попереднього аналізу спроектованих обчислювальних сценаріїв є важливою умовою для забезпечення ефективної та комфортної роботи проектувальника. Як вже зазначалося, засоби виконання BPEL-сценаріїв зобов'язані перевіряти їх лише на синтаксичні помилки та деякі структурні вади (недопустимі поєднання базових конструкцій). Зважаючи на це, виконання непроаналізованого сценарію може не дати адекватних результатів, необгрунтовано затягнути або достроково завершити виконання потоку робіт внаслідок логічних помилок проектувальника (вічні цикли, тупикові гілки тощо). Іншою стороною аналізу є передбачення часу виконання потоку та визначення критичних місць (англ. bottlenecks) у сценарії.

Незручністю під час аналізу потоку робіт стає те, що мова BPEL малоприсадибна для автоматизованого дослідження логічних вад та інших властивостей сценарію. Натомість існує досвід використання численних математичних абстракцій для аналізу потоків робіт.

Теорія масового обслуговування. Суть підходу полягає у представленні кроків робочого потоку у вигляді моделей СМО. Вхідні черги моделюють механізм передавання повідомлень між сервісами робочих потоків, а час затримки моделює час виконання одного кроку.

Теорія автоматів. Цей підхід ґрунтується на представленні усіх сервісів потоку, що обмінюються повідомленнями, як кінцевих автоматів. Множина таких автоматів разом із чергами повідомлень моделюватиме робочий потік, який можна дослідити на наявність блокувань, перегонів, невизначеностей та ін., тобто – верифікувати опис конкретного потоку на відповідність очікуваній поведінці. Описані приклади використання подібної моделі для аналізу робочого потоку, описаного мовою BPEL, та перекладу отриманого аналітичного опису мовою Promela (Process Meta Language) для автоматизованого моделювання за допомогою верифікатора SPIN (Simple Promela Interpreter) [11].

Теорія графів. Звернення до елементів теорії графів не лише для опису, а й подальшого аналізу робочих потоків є достатньо природним рішенням: вершини та ребра графів слугують абстракцією для кроків потоку та переходів між ними.

Добре розвинена теорія мереж Петрі, що базується на класичній теорії графів і є розширенням теорії кінцевих автоматів, є доволі популярним засобом аналізу робочих потоків, як абстракція дискретних розподілених систем. У межах цієї теорії чітко визначені поняття стану, переходу та ін., вона добре досліджена математично. Позиції мережі Петрі відповідають крокам потоку, переходи мережі – переходам у потоці, послідовність виконання потоку задається дугами мережі Петрі. Шлях до аналізу реального робочого потоку – переклад його опису на модель мережі Петрі та її подальше дослідження. Так, можливий повний переклад елементів описової мови BPEL примітивами мереж Петрі [12]. Однак, зважаючи на запропонований багатокроковий підхід до створення опису обчислювального сценарію (редактор – граф – перекладач – BPEL-опис), доцільніше може виявитись аналізувати первинну модель на основі графу, як ближчу до мережі Петрі, ніж вторинний BPEL-опис. Мережа Петрі може використовуватись, наприклад, для такої

практичної задачі, як виявлення циклів та тупикових гілок в описі потоку. Автоматизований аналіз мережі Петрі, як правило, виконується за допомогою побудови дерева досяжності.

Алгебри процесів. Числення процесів (або алгебра процесів) – ще один альтернативний підхід для формального моделювання конкурентних систем. Це аналітичний інструмент для високорівневого опису взаємодії та синхронізації між незалежними процесами, із своїми алгебраїчними законами, що дають змогу аналізувати описи процесів та судити про їх еквівалентність. Існує чимало різновидів таких числень (CSP, CSS, π -числення), але всі вони мають ряд спільних властивостей: взаємодія між процесами, як правило, описується через концепцію передавання повідомлень, а не спільної пам'яті; використання для опису складних процесів невеликого набору примітивів та операцій на них; визначення алгебраїчних законів для цих операцій, що дають змогу виконувати еквівалентні перетворення тощо. Одним з відносно нових та популярних формалізмів з цієї родини є π -числення (π -calculus), що є розвитком роботи над численням процесів CCS (англ. Calculus of Communicating Systems). π -числення надає можливість опису конкурентних процесів, конфігурація яких може змінюватись під час роботи. Чимало робіт присвячено аналізу потоків робіт (в т.ч. № 686 BPEL) за допомогою π -числення [13].

Високопродуктивні обчислення у BPEL-процесах

На початку статті вже зазначалося, що характерною рисою сучасних інженерних та наукових обчислень є залучення високопродуктивних обчислювальних ресурсів, які дають змогу проводити масштабні експерименти за адекватний проміжок часу. Окреме місце посідає технологія грід як дієвий механізм організації скоординованого використання неоднорідних, географічно рознесених, високопродуктивних обчислювальних ресурсів, наданих у спільне використання.

Нині грід-користувач, як правило, взаємодіє із грід-ресурсами через командну консоль, оскільки таким є стандартний підхід до надання інтерфейсу користувача розробниками програмного забезпечення грід проміжного шару (ПЗПШ). Поруч із консоллю команд, більшість ПЗПШ надає програмні інтерфейси та бібліотеки для розроблення більш розвинутих графічних клієнтів та грід-порталів. Зважаючи на це, постає проблема включення засобів управління грід-задачами до BPEL-орієнтованої системи управління науковими та інженерними обчислювальними сценаріями. Нижче наведені можливі шляхи її вирішення.

Грід-сервіси

Спробу адаптації веб-сервісів до умов динамічного грід-середовища відображено у концепції відкритої архітектури грід-сервісів OGSA та її останній реалізації – WSRF. Кілька ПЗПШ (Globus Toolkit 4, UNICORE 6) були розроблені саме за стандартами WSRF, що робить потенційно можливим використання грід-сервісів WSRF як складових сервісів обчислювального сценарію. Однак їх специфіка та неповна сумісність зі стандартним WS-інструментарієм не дає змоги легко застосувати цей підхід. Потрібні додаткові зусилля для адаптації BPEL-засобів до WSRF-сервісів [14].

Веб-сервіси доступу

Альтернативним підходом є розроблення стандартного SOAP-веб-сервісу (відповідно № 686 сумісного з BPEL-інструментарієм) для запуску задач на грід-ресурсах, використовуючи інтерфейси та бібліотеки конкретних реалізацій ПЗПШ. Це дасть змогу зберегти одну з важливих переваг запропонованої архітектури – орієнтацію на стандарти веб-сервісів.

Висновки

Серед підходів до побудови сучасних систем управління обчислювальними сценаріями, які орієнтуються на нинішні стандарти веб-технологій, на окрему увагу заслуговує ідея поєднання workflow-концепції для представлення сценаріїв та стандартного BPEL-інструментарію компонування веб-сервісів для забезпечення їх виконання. Розглянуто окремі аспекти такого підходу з погляду його придатності для розв'язання інженерних та наукових обчислювальних задач, та висунуто пропозиції щодо загальної архітектури системи та реалізації її окремих складових. Орієнтація системи на існуючі

стандарти, модульність її архітектури дадуть змогу розвивати її функціональність синхронно із подальшим розвитком веб-технологій та стандартів. Перспективні напрямки для вдосконалення запропонованої системи – залучення семантичних технологій для автоматичної композиції сценаріїв та впровадження розвинутих процедур аналізу спроектованих рішень.

1. *Worldwide LHC Computing Grid (WLCG) project № 686* <http://lcg.web.cern.ch/lcg/public/>.
2. V. Curcin, M. Ghanem. *Scientific workflow systems – can one size fit all?* // *Proceedings of Biomedical Engineering Conference CIBEC 2008. – Cairo International. – 2008. – P. 1–9.*
3. *Triana – Open Source Problem Solving Software* – <http://www.trianacode.org/>
4. *The Kepler Project* – <https://kepler-project.org/>
5. *Taverna Workflow Management System* – <http://www.taverna.org.uk/>
6. B. Ludäscher, M. Weske, T.M. McPhillips, S. Bowers. *Scientific Workflows: Business as Usual?* // *Proceedings of the 7th Intl. Conf. on Business Process Management BPM Ulm, Germany. – 2009. – P. 31–47.*
7. J. Jang, B. Jeong, H. Cho, J. Lee. *Capability and Extension of UDDI Framework for Semantic Enterprise Integration.* // *Proceedings of International Conference on Advances in Production Management Systems, Washington D.C., September 18–21, 2005.*
8. C. Goodwin, D.J. Russomanno, J. Qualls. *Survey of Semantic Extensions to UDDI: Implications for Sensor Services.* // *Proceedings of the International Conference on Semantic Web and Web Services, CSREA Press, Las Vegas, Nevada. – 2007. – P. 16–22.*
9. *Web Services Business Process Execution Language* – docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf.
10. I.J. Taylor, E. Deelman, D.B. Gannon, M. Shields (Eds.). *Workflows for e-Science. Scientific Workflows for Grids.* – Springer. – 2007. – 530 p.
11. X.Fu, T.Bultan, J.Su. *Analysis of interacting BPEL Web Services* // *Proceedings of the 13th international conference on World Wide Web. – 2004. – P. 621–630.*
12. C. Ouyang, E. Verbeek, W. van der Aalst et al. *Formal semantics and analysis of control flow in WS-BPEL* // *Science of Computer Programming. – Vol.67, No.2–3. – July, 2007. – P. 162–198.*
13. M. Weidlich, G. Decker, M. Weske. *Efficient Analysis of BPEL 2.0 Processes using pi-Calculus.* // *Proceedings of the IEEE Asia-Pacific Services Computing Conference (APSCC'07), Japan. – 2007. – P. 266–274.*
14. T. Dörnemann, T. Friese, S. Herdt et al. *Grid Workflow Modelling Using Grid-Specific BPEL Extensions.* // *Proceedings of German e-Science Conference, Baden-Baden. – 2007.*