УДК 004.45:007.51

ПАТЕРНИ ВЗАЄМОДІЇ СППР

Ю.В. Бондарчук
Київський національний університет імені Тараса Шевченка
e-mail: byv@univ.kiev.ua

Осмислення СППР, на думку автора, має дослідити наступні елементи абстракції СППР (рис. 1):

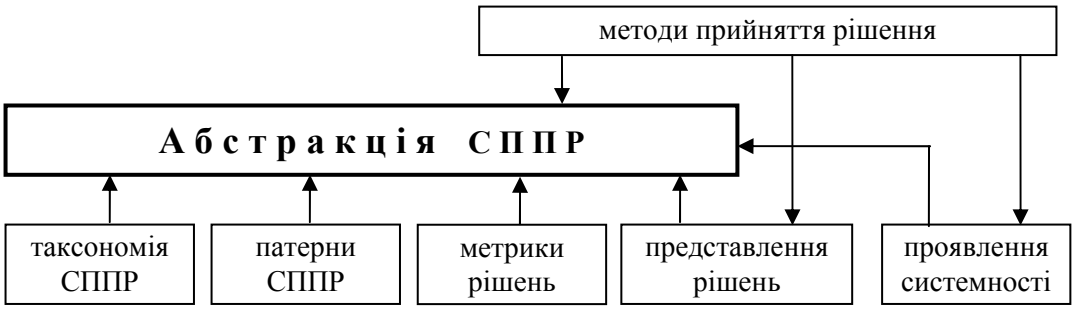

Рис. 1

1) *проявлення системності* СППР, тобто що їх робить окремими системами поруч з іншими системами (дещо є в [1]).
2) *таксономія* (класифікація) завершиться (формалізується) після наповнення інших елементів абстракції (деякі пропозиції є в [2]);
3) *методи прийняття рішень* (МПР), як на думку автора, взагалі розвиваються не як прикладна наука – цікаві для МПР задачі є непідйомними для оперативного прийняття рішень, бо рекомендації потрібні не через 100 років обчислень;
4) *метрики рішень* логічно шукати для опису:
   - складності рішення, але не алгоритмічної складності, а пошуку серед рішень в множині СППР (є деякі ідеї в [3]);
   - порівняння рішень як формальної процедури;
   - представленні місця нового рішення серед вже існуючих в множині СППР;
5) *представлення рішень*, кодування рішень є дуже відсталим від потреб часу елементом теорії СППР. Тут потрібний "вибух", подібний до HTML для WWW, коли Інтернетом змогли користуватися всі;
6) *патерни* (шаблони) СППР, як програмних систем, є особливими способами діяльності з конструювання/пошуку рішення для ОПР.

Для чого наводяться 6 елементів, якщо розглядатиметься лише один? Щоб було зрозуміле місце патернів серед всіх абстракцій. Чітко окреслені патерни, зокрема, служитимуть наповненню таких елементів абстракції СППР, як системність, таксономія та метрики.

Склад патернів СППР пропонується такий:

1) *логічні* патерни представляють уявлення ОПР про СППР і роботою над рішенням, або політики для ОПР, зокрема, що ОПР може/не може в СППР;
2) *архітектурні* патерни стосуються структури СППР;
3) патерни *рішень* задають спосіб отримання рішення, особливо коли вже є певна множина варіантів (наприклад, результат пошукового запиту), можливі варіанти суперпозиції рішень, редукції тощо;
4) патерни *доступу до даних*, з якими працює ОПР;
5) патерни *взаємодії* СППР описують можливості СППР зв'язуватися з іншими СППР, бути розподіленими чи бути СППР-хмарами.

Патерни взаємодії стосуються можливості однієї СППР мати доступ до бази рішень (моделей, даних) інших СППР та, в свою чергу, бути відкритою для інших СППР. Тобто мова йде про можливу степінь віртуалізації СППР. Наприклад, звернення ОПР до іншої СППР за рішенням може породити цілий ланцюжок проходження СППР. За аналогією, як аудіо/відео/web контент не залежить від ОС, хоча є різні платформи ОС. Також практика WWW не пішла шляхом декількох суперуніверсальних web-серверів для всіх, але навпаки – реально працювати з мільйонами web-серверів. Якщо зробити дещо фантастичне зараз припущення, що СППР "відкриваються" для обміну контентом чи генерації рішень на запити інших СППР (представляючи інтерфейс для ОПР), то які патерни (шаблони) їх взаємодії можливі? Пропонуються такі патерни взаємодії (рис. 2):

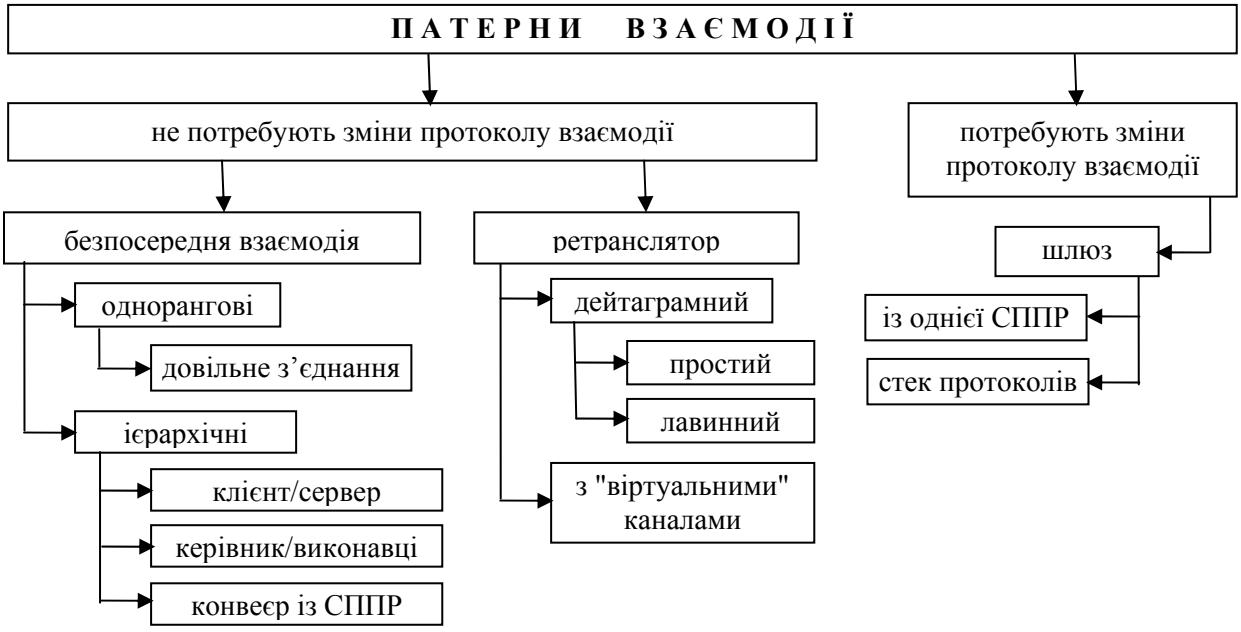

Рис. 2

Самі патерни відносяться до трьох рівнів абстракції.

1-й рівень стосується відношення до "родини" СППР із спільним протоколом та різними протоколами взаємодії.

2-й рівень абстракцій уточнює роль СППР-ініціатора взаємодії у сценарії дій з іншими СППР та ролі інших СППР у формуванні рішення. Отже тут є три абстракції: *безпосередня взаємодія*, *ретранслятор* та *шлюз*.

3-й рівень деталізує ролі СППР у взаємодії. Так ролі, наприклад, відрізняють взаємодію за патернами *конвеєр із СППР* чи *з "віртуальними" каналами*, чи *стек протоколів*, але ззовні вони схожі на конвеєр. Фактично 3-й рівень – рівень реалізації.

Інтерпретація 9-ти патернів 3-го рівня абстракції наступна.

1. **Довільне з'єднання**. СППР звертається до будь-якої відомої їй СППР. Формально адресат якби виконує роль сервера, проте це не є виділеною роллю сервера у такій взаємодії – це однорангова взаємодія.

2. **Клієнт/сервер**. В цій взаємодії СППР-адресат наділена функцією сервера рішень. Як і в класичному патерні паралельних обчислень, СППР-сервер може бути *виділеним*, *розподіленим* або *реплікованим* сервером рішень/даних. Найвищим вираженням такого патерну, мабуть, буде хмарна СППР (розподілені обчислення в мережі серверів з балансуванням навантаження та кешуванням запитів). Певна степінь зрілості СППР потребуватиме віртуалізації взаємодії у вигляді хмар СППР.

3. **Керівник/виконавці**. СППР-ініціатор запитів звертається до множини СППР, "знаючи" що кожна з них може добавити щодо генеруємого рішення (на відміну від патерна довільного з'єднання). Але рішення, моделі чи дані ініціатор запитів із відповідей

конструює сам. Цей патерн для випадку, коли потрібне рішення є декомпозицією (обгорткою) окремих підзадач, які знаходяться в різних СППР. Також це може бути "опитування" експертних систем, коли їх відповіді залежать від інкапсульованих в них рішеннях, моделях та даних.

4. **Конвеєр із СППР**. Цей патерн передбачає взаємодію СППР, коли результат щодо запиту якоїсь СППР передається на вхід іншої, передбаченої сценарієм взаємодії СППР. Результат роботи такого конвеєра повертається ініціатору взаємодії. Кожний клас рішень може мати свій конвеєр. Отже, рішення чи необхідні дані з'являються після проходження конвеєру. Природним застосуванням може бути при роботі з експертними системами, в ситуативних центрах. Важливим є те, що СППР конвеєру обов'язково генерують свою частину рішення/даних. Формально цей патерн може бути реалізованим патерном керівник/виконавці, але тоді всі проміжні дані мають поступати до СППР-ініціатора і ще раз передаватися до СППР наступного етапу генерації рішення. А от конвеєр максимально мінімізує дублювання переміщення даних.

5. **Простий дейтаграмний ретранслятор**. Робота СППР як ретранслятора означає бути посередником у передачі запитів до цільової СППР (яка може задовільнити запит), але наперед невідома. Тобто якась СППР має необхідне рішення/дані, але на момент запиту це невідомо СППР-ініціатору (ситуація, дані, експертні оцінки тощо постійно міняються). Інакше б ініціатор звернувся до такої СППР напряму. Дейтаграмність у роботі ретранслятора (за аналогією із пакетними мережами з однокроковою маршрутизацією) означає довільну маршрутизацію до цільової СППР. СППР-ініціатор звертається до будь-якої відомої їй СППР. Якщо та не може виконати запит (не має рішення/даних), вона також звертається до будь-якої відомої їй СППР, крім вже пройдених, тобто виступає ретранслятором запиту. Конструювання маршруту проходження СППР має завершитися досягненням цільової СППР. Зрозуміло, що сценарій конструювання рішення може передбачати обмеження на кількість ретрансляцій (хопів), бо, врешті, жодна із СППР, до яких можливий доступ, може нічим і не "допомогти" ініціатору. ***Інтерпретація 2***. Цей патерн також може бути використаним і у випадку, коли рішення конструюється по маршруту проходження СППР, а не обов'язково лише все вирішується у цільовій СППР (але це і не патерн конвеєра СППР, бо маршрут не фіксується).

6. **Лавинний дейтаграмний ретранслятор**. На відміну від простого ретранслятора, коли конструювання маршруту до цільової СППР зводилося до звернення до однієї із відомих СППР, лавинна дейтаграмність означає, що СППР-ініціатор зразу звертається до кількох або всіх відомих СППР. Кожна з приймаючих запит СППР стане цільовою, або виступить як ретранслятор. Лавинний патерн дозволяє знайти кілька цільових СППР і отримати вже варіанти рішень/даних. Однак не лише неправильно працюючий ініціатор може створити аналог DDoS атаки, бо для цього буде достатньою добра "обізнаність" СППР одна про одну.

Можливий різновид лавинного звернення до СППР, коли ініціатор звертається до всіх своїх сусідів (в смислі метрики рішень [3], а не просторового розміщення) є, знову ж за аналогією із мережевою маршрутизацією, mesh-з'єднанням СППР.

7. **Ретранслятор з "віртуальним" каналом** працює із вже заданим переліком СППР. Цей патерн використовується коли для СППР-ініціатора вже є знання про множину СППР, які можуть мати потрібні рішення/дані, але ініціатору постійно про це невідомо. Таким чином, ініціатор пропонує фіксований маршрут через ряд СППР. Якщо якась СППР із маршруту може задовольнити запит, то вона стає цільовою та проходження маршруту завершується. Інакше така СППР стає лише ретранслятором запиту до наступної у маршруті СППР. І така наступна СППР вже прописана у маршруті. Зрозуміло, що проходження всього маршруту може дати негативний результат. Робота за цим

патерном має "підступну" особливість – вибір маршруту за певним стереотипом попереднього досвіду.

8. **Шлюз із однієї СППР**. Всі попередні патерни були розраховані на спільний протокол взаємодії всіх СППР. Так СППР-ретранслятор протокол не змінювала, а лише була учасником конструювання маршруту. Для взаємодії СППР-ініціатора із цільовою СППР, коли вони з різними протоколами (рішень, моделей, даних), потрібний посередник у трансляції, який за аналогією із мережевими сервісами пропонується називати *шлюзом*. Якщо транслятор є у СППР-ініціатора, то це був би той же патерн довільного з'єднання. Отже мова йде про те, що шлюзом є окрема СППР. Важливо зауважити, що якщо такий шлюз є просто формальним транслятором, то термін СППР-шлюз був би зайвим (транслятор він і є транслятор). Було б доречно використати термін, наприклад, *серверний шлюз*. ***Інтерпретація 1***. СППР-шлюз, крім автоматичної трансляції, може залучати ОПР, експертів, нейромережі тощо для правильної трансляції. ***Інтерпретація 2***. У шлюзі може бути реалізована і задача класифікації (типу *що це*), після якої йде звернення до спеціалізованої СППР. Такий шлюз є типовим брокером рішень у мережі СППР.

9. **Стек протоколів**. На відміну від патерна *шлюз із однієї СППР*, який виходить із того, що дві взаємодіючі СППР працюють із різними протоколами взаємодії. Може бути так, що дві СППР працюють за спільним протоколом, але безпосередньо вони між собою взаємодіяти не можуть, бо, наприклад, знаходяться в різних хмарних СППР із своїми політиками. І в цих політиках не передбачені СППР-ретранслятори.

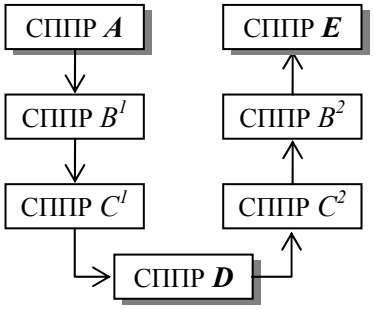

Рис. 3

Інтерпретація 1. Наприклад, *СППР A* має задіяти *СППР E* (див. рис. 3) і це можна зробити лише через *СППР D*, яка працює за відмінним для *A* та *E* протоколом. Тобто *A* та *E* не мають транслятора для *D*. Наприклад, B^1 є шлюзом даних (моделей), C^1 є лише шлюзом протоколу представлення рішень. Отже, лише через B^1 та C^1 СППР *A* може взаємодіяти з *D*. В свою чергу, *D* для взаємодії з *E* повинна пройти шлюзи C^2 та B^2 у зворотному порядку. Ланцюжок A-B^1-C^1-D-C^2-B^2-E представляє стек протоколів: чим *A* починала взаємодію із *E*, тим *E* завершує і чим *A* завершувала взаємодію із *E*, з того *E* починає. Пари СППР B^1 та B^2 (як і C^1 та C^2) є різними системами – їх об'єднує лише роль бути відповідними шлюзами.

Інтерпретація 2. *A* передає рішення *E*, яке має пройти три рівня, наприклад, експертних СППР, де B^1 та B^2 (як і C^1 та C^2) є різними СППР (експертами), але з однакових груп питань. І лише *D* спільна експертна СППР для *A* та *E*. СППР B^1 та B^2 (як і C^1 та C^2) працюють незалежно, ніяк не узгоджуючи свої рішення. Отже, рішення від *A* проходить стек експертних систем для використання у *E*.

Також є можливою композиція основних 9-ти патернів. Власне в хмарних СППР це так і буде, бо для користувача це буде виглядати патерном довільного з'єднання.

Список літератури

1. Бондарчук Ю.В. Щодо модифікації парадигми СППР //Системи підтримки прийняття рішень. Теорія і практика (СППР-2013): Збірник доповідей наук.-практ. конф. – Київ: ІПММС НАНУ, 2013. - С. 9-11.

2. Бондарчук Ю.В. Варіант таксономії СППР //International conference "Problems of decision making under uncertainties" (PDMU-2014).). 12-16.5.2014, Мукачево. - С. 68-69.

3. Бондарчук Ю.В. Метричні характеристики розподілених СППР //International Workshop "Problems of decision making under uncertainties" (PDMU-2013). 13-17.5.2013, Східниця. - С. 93-94.