УДК 004.942

М.С. ЧЕСАНОВСЬКИЙ*

ФОРМАЛЬНО-АЛГОРИТМІЧНА ТА ПРОГРАМНА ІМПЛЕМЕНТАЦІЯ МОДЕЛЕЙ КЕЙСІВ ДАНИХ ПРО ПРОЦЕС БУРІННЯ

*Івано-Франківський національний технічний університет нафти і газу, м. Івано-Франківськ, Україна

Анотація. *Проаналізовано доцільність та перспективи гібридизації кейс-базованих міркувань та специфікацій інформаційно-пошукових задач на основі обмежень. Узагальнено результати формально-алгоритмічної імплементації моделей кейсів даних про процес буріння шляхом моделювання відповідних типових кейсів. Показано, що кейс є певного роду екземпляром для процесу рішення технологічної проблеми, що виникає у процесі буріння. Виділено два основних компоненти кейсу: власне опис проблеми та рішення поточної технологічної проблеми як такої. В результаті пропонована методологія максимально опирається на попередній досвід інтелектуальної системи у формі вже вирішених успішних кейсів, з яких формується можливий набір релевантних контрольованих параметрів технологічного процесу відповідно. В той же час, весь процес модифікаційної адаптації кейсів пропонується розглядати як процес рішення інформаційно-пошукових задач із накладеними обмеженнями, що дає можливість розвинути досить ефективну методологію, яка може бути поширена на цілий ряд предметних областей. За допомогою релевантності кейсу в заданий момент часу можливо встановлювати межі, при яких необхідно повідомляти операторам про відповідність отриманих значень до очікуваних у кейсі. Це має сенс у режимі реального часу при накладанні кейсів, які отримали повідомлення про небезпеку або некоректні налаштування бурової установки, що може призвести до надмірного зношення, зупинки чи виведення з ладу бурової. Проведене тестування системи підтвердило успішність функціонування спроектованого інформаційного інтелектуального рішення на множині фактичних даних про процес буріння нафтових та газових свердловин.*

Ключові слова: *кейс-базовані міркування, інформаційно-пошукові задачі на основі обмежень, технологічні проблеми, релевантність, рішення кейсу, буріння свердловин, інтелектуальна система.*

Аннотация. *Проанализированы целесообразность и перспективы гибридизации кейс-базированных размышлений и спецификаций информационно-поисковых задач на основе ограничений. Обобщены результаты формально-алгоритмической имплементации моделей кейсов данных о процессе бурения путем моделирования соответствующих типовых кейсов. Показано, что кейс является своего рода экземпляром для процесса решения технологической проблемы, возникающей в процессе бурения. Выделены два основных компонента кейса: собственно описание проблемы и решения текущей технологической проблемы как таковой. В результате предлагаемая методология максимально опирается на предыдущий опыт интеллектуальной системы в форме уже решенных успешных кейсов, из которых формируется возможный набор релевантных контролируемых параметров технологического процесса соответственно. В то же время, весь процесс модификационной адаптации кейсов предлагается рассматривать как процесс решения информационно-поисковых задач с наложенными ограничениями, что дает возможность развить достаточно эффективную методологию, которая может быть распространена на целый ряд предметных. С помощью релевантности кейса в заданный момент времени возможно устанавливать границы, при которых необходимо сообщать операторам о соответствии полученных значений, ожидаемых в кейсе. Это имеет смысл в режиме реального времени при наложении кейсов, которые получают сообщение об опасности, или некорректные настройки буровой установки, что может привести к чрезмерному износу, остановке или выводу из строя буровой. Проведенное тестирование системы подтвердило успешность функционирования спроектированного информационного интеллектуального решения на множестве фактических данных о процессе бурения нефтяных и газовых скважин.*

Ключевые слова: *кейс-базированные рассуждения, информационно-поисковые задачи на основе ограничений, технологические проблемы, релевантность, решение кейса, бурение скважин, интеллектуальная система.*

Abstract. *The expediency and prospects of hybridization of case-based considerations and specifications of information-search constraints based problems are analyzed. The results of the formal-algorithmic implementation of case data models on the drilling process by simulating the corresponding typical cases are generalized. It is shown that the case is a kind of instance for the process of solving a technological problem that occurs during the drilling process. There are two main components of the case – the actual description of the problem and the solution of the current technological problem as such. As a result, the proposed methodology is based as much as possible on previous experience of the intellectual system in the form of already solved successful cases from which a possible set of relevant controllable parameters of the technological process is formed, respectively. At the same time, the whole process of modifying adaptation of cases is proposed to be regarded as a process of solving information retrieval problems with imposed constraints, which makes it possible to develop a very effective methodology that can be extended to a number of subject areas. With the help of relevance of the case at a given time point it is possible to set limits on which it is necessary to inform operators about the correspondence of the received values to the expected in the case. It makes sense in real-time when applying cases that have reports of danger or incorrect drilling rig settings, which can lead to excessive wear, stopping, or malfunctioning of the drilling rig. The conducted testing of the system has confirmed the success of the operation of the designed information intelligent decision on the set of actual data on the process of drilling of oil and gas wells.*
Keywords: *case-based considerations, constraints based problems, technological problems, relevance, case handling, well drilling, intellectual system.*

1. Вступ

Кейс-базовані міркування (міркування на основі прецедентів відповідно) [1–6] є однією із найбільш просунутих технік у штучному інтелекті, оскільки дозволяє ефективно поєднати типи базових концепцій у штучному інтелекті (методи міркування і методи машинного навчання для вирішення проблем), базуючись на попередньому досвіді в формі кейсів-прецедентів-випадків. Виходячи із заданої проблеми (технологічної проблеми в області буріння нафтових і газових свердловин [7–8]), методологію міркування (розмірковування) слід відповідно застосовувати для того, щоб видобути минулі схожі кейси з метою повторного або модифікованого застосування вирішення проблем, що в них закладено. Як тільки рішення буде знайдено, то можна застосувати відповідні методи машинного навчання для покращання знань системи, базованих на прецедентно-кейсовому експертному досвіді систем: з програмно-прагматичної точки зору слід приєднати до баз-кейсів новий кейс, а до баз знань новий експертний досвід.

Розвиток технологій за принципом швидкого розроблення застосувань накладає свій відбиток, безумовно, і на досліджувану проблему, що вимагає створення методологій для ефективного дослідження простору рішень та відповідного вибору рішення з найменшою вартістю та найбільшою ефективністю, покращеною функціональністю, і якістю застосування як такого. Поставлена задача має значний простір вирішення. Проте, безумовно, комплектність проблеми є суттєвим бар'єром для її вирішення. Поява саме класу певних систем, систем на основі знань дає можливість використання саме минулого досвіду для побудови нових ефективних рішень. Тому серед підходів менеджменту знань слід виділяти саме ефективність підходу кейс-базованих міркувань (КБМ) та інформаційно-пошукових задач на основі обмежень (ІПО) (Constraints satisfaction problem – CSP, відповідно) [9–10], їх можливому поєднанні та спільній гібридній імплементації, що визначає загальний фреймворк представленого дослідження, *метою* якого є формально-алгоритмічна та програмна імплементація моделей кейсів даних про процес буріння, що дасть можливість надавати рекомендації операторам технологічного процесу у часі, наближеному до реального, без необхідності ручного введення значень контрольованих параметрів.

2. Основна частина

Кожен кейс представляє відповідно опис деякої минулої проблеми TP^{src} і відповідного асоційованого рішення минулої проблеми $TP^{src}.Sol \to Sol^{src}$. На основі введених характеристик надійності та метричної відстані найбільш схожі кейси можуть бути видобуті з необхідною адаптацією при відповідній потребі, що буде розглядатися як відповідне рішення для нової проблеми $TP^{src}.Sol \to Sol^{src}$. Суть нової проблеми полягає у формуванні моделі знань через визначену кількість змінних (параметрів, технологічного процесу): $Md(\text{КВ.ТР}) = \{\{V\},\{V\},...,\{V_n\}\}_{i,j,m}$, тобто виражатимемо модель знань через вид представлення для бази знань. Базуючись на такій основі (змінні – параметри), можна також легко переходити від рівня даних до рівня знань, що вже є задачею концептуального рівня, яка проте початково може мати просте вирішення, якщо виходити з того, що база даних – це сукупність ініціалізованих параметрів, що спостерігаються (або контролюються відповідно) на вході і виході технологічного процесу TP^{in}, TP^{out}. Відповідно, можемо розглядати представлення виду $\Delta B = \{p_1^{in} = v_i^1, \cdots, p_{n1}^{in} = v_{n1}^1, p_1^{out} = v_i^2, \cdots, p_{n2}^{out} = v_{n2}^2\}$ для бази даних технологічного процесу $TP^{in} \cdot \Delta B$ та відповідно представлення через стрілку для бази знань $TP.KB = \{\{p_1^{in} = v_i^1, \cdots, p_{n1}^{in}\} \to \{p_1^{out} = v_i^2, \cdots, p_{n2}^{out} = v_{n2}^2\}_j\}_{i,j,m}$.

Тобто, в найбільш загальному випадку кейс представлятиметься деяким кортежем виду $\langle TP, Sol, Out.Tp \rangle$, де TP, власне, технологія проблеми, Sol – її очікуване рішення, $Out.Tp$ – очікувані результати. Проте таке представлення не може претендувати на повну вичерпність щодо складових компонент кейсу. Наприклад, за аналогією з експертними системами можна також включати константу пояснень, що дасть змогу підсилити складову зворотного зв'язку з клієнтом.

У загальному випадку сутність пропонованої методології така:

1. Видобування початкового первинного набору обмежень (обов'язкового), що задовольняє кейс:

$$C_{set}^{init} \big| = TP.Case^{new}.$$

2. Виділення початкового набору параметрів, підстановки яких покриватиме набір цільових обмежень:

$$TP_{set}^{init} = (\mathrm{p}_1^{init}, ..., \mathrm{p}_n^{init}) \big| - CS^{trg}.$$

3. Вирішити накладені обмеження за допомогою солвера:

$$Solve(CS^{trg}) \to \{(\mathrm{p}_1^{sol}, ..., \mathrm{p}_k^{sol}) \big| = CS^{trg}\}.$$

4. Визначити істинність для стану надобмеженості (недообмеженості) системи (випадок, коли накладено надто багато (надто мало) обмежень і, відповідно, через це немає рішення (існує надто багато рішень)):

$$\{Sol.\#(CS^{trg}) << \#[CS^{trg}]\} \to \text{Relax}[CS^{trg}],$$
$$\{Sol.\#(CS^{trg}) >> \#[CS^{trg}]\} \to \text{Restrict}[CS^{trg}].$$

5. Перевірити істинність стану, для якого існує рівно одне рішення:

$$\#\, p[TP.State].Sol = \min_{\#}\{Solve[CS^{trg}]\}.$$

6. Якщо система є надообмеженою (недообмеженою) відповідно, то слід повторити реструктуризацію початкового набору параметрів:

$$if_{under}^{Over}\{(CS^{trg})\} \to \text{Reshuffle}^{ou}\{p_1^{init}CSet, \cdots, p_n^{init}CSet\}.$$

7. Вивести рішення для експерта. Якщо рішення задовольнятиме експерта та переходитиме до завершення зі збереженням вибраного методу адаптації та отриманого рішення відповідно:

$$if\{p_1^{sol}CSet^{sol}, \cdots, p_k^{sol}CSet^{sol}\}.\{CSet_1^{sol}, \cdots, CSet_k^{sol}\}\Big|=^{sd}\{CS^{trg}\},$$

де $sd \cong CF$, де CF – ступінь впевненості експерта в виділеному стані $\{TP.State.CS^{trs}\}$.

8. Якщо рішення не задовольнятиме експерта, то тоді слід змінити метод адаптації шляхом модифікації підстановки і перейти до кроку 3.

$$if\ sd <> CF \to \text{Reshuffle}^{sol}(\{p_1^{sol}.CSet_1^{sol}, \cdots, p_l^{sol}CSet^{sol}\}),\ \text{де}\ l \le k.$$

9. Зберегти нове рішення та використаний метод адаптації у випадку їх достатності релевантності (задаючи ступінь релевантності як такої).

$$rd(\{p_1^{sol}.CSet_1^{sol}, \cdots, p_k^{sol}CSet_k^{sol}\}, TP.\text{State.CS}^{trg}),$$

$$if\ \ rd \to 1, CBase.\{\{p_1^{sol}.CSet_1^{sol}, \cdots, p_k^{sol}CSet_k^{sol}\}, TP.\text{State.CS}^{trg}, \text{Reshuffle}^{ou,sd}(\ )\}.$$

Такий підхід, на відміну від стандартної монолітної архітектури, дозволить розробляти, впроваджувати та масштабувати підсистеми незалежно одна від одної.

Переваги такого підходу:

1. Дає можливість розвивати кожну частину рішення паралельно, невеликими командами.

2. Кожен такий сервіс можна розгортати незалежно. Якщо одна частина системи вже готова до впровадження, то немає потреби затримувати її, якщо інші сервіси ще на стадії розробки. Це дає можливість доставки рішення швидше.

3. Масштабованість. Кожен із сервісів можна запускати в багатьох примірниках. Якщо є навантаження тільки на деякі сервіси, то можна саме цей розмножити і горизонтально масштабувати.

4. Сервіси, які відповідають тільки за одну частину, є більш простим рішенням, ніж загальна монолітна система. Якщо конкретний сервіс перестає задовольняти конкретним вимогам, його можна викинути чи переписати заново. Це не буде настільки складним і дорогим, якщо б зміни потрібно було вносити в моноліт, особливо, якщо необхідно змінити цілий підхід.

5. Можливість застосовувати різні технології. Стандартизація повинна бути, але якщо є достатньо підстав поміняти технологію, то можна це зробити як частково, так і повністю. При цьому не потрібно робити такий перехід відразу, а можна розділити це на декілька ітерацій.

Таким чином, розроблене рішення було розділено на ряд підсистем (сервісів).

1. AdapterService – кожен екземпляр даного сервісу підлаштовується під конкретне джерело(давач, панель управління тощо) для отримання значень параметрів.

2. ParameterService – відповідає за управління параметрами та збереження значень параметрів.

3. CaseService – відповідає за управління кейсами.

4. CaseResolverService – виконує резолвінг кейсів за заданими критеріями, а також зберігає результати виконання резолвінгів.

5. NotificationService – відповідає за доставку повідомлень на сторонні ресурси для інформування персоналу про виникнення позаштатних ситуацій.

6. CaseManagementPortal – веб-орієнтований портал для менеджменту параметрів та кейсів, а також, резолвінгу кейсів у ручному режимі, що може використовуватись операторами.

7. IdentityService – сервіс для авторизації та аутентифікації, побудований за технологією, при використанні якої користувач переходить від одного сервісу до іншого.

Крім того, IdentityServer є реалізацією технології Single SignOn для проектів на базі ASP.NET Core. За допомогою цієї технології було реалізовано можливість взаємодії між різними частинами системи без потреби повторної авторизації, фактично, один AccessToken використовується для всіх підсистем. Загальна архітектура розробленого рішення має вигляд:

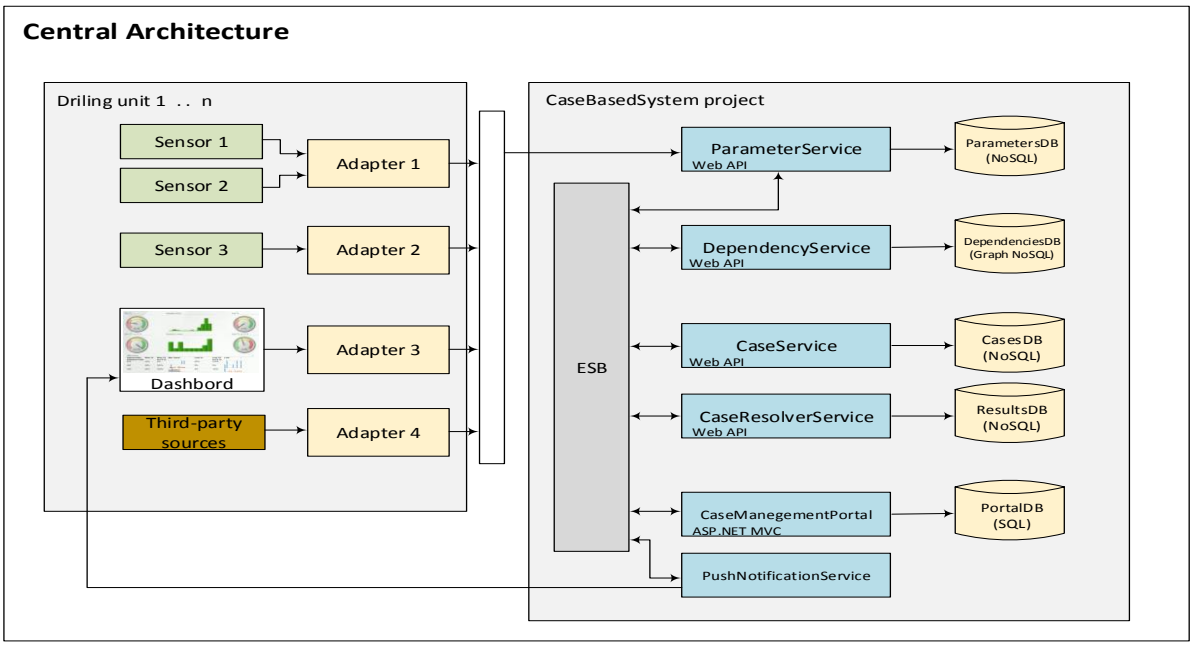


Рисунок 1 – Загальна архітектура розробленого рішення

AdapterService

Процес збору даних вимагає конкретної стандартизації вхідних значень параметрів, які будуть зберігатися в базі даних. Втім, дані з давачів є різнотипними, що вимагає адаптації цих потоків значень параметрів перед доставкою їх у систему. Саме тому було розроблено концептуальну архітектуру для рішення проблеми адаптації даних. Загальна архітектура цього рішення має вигляд:

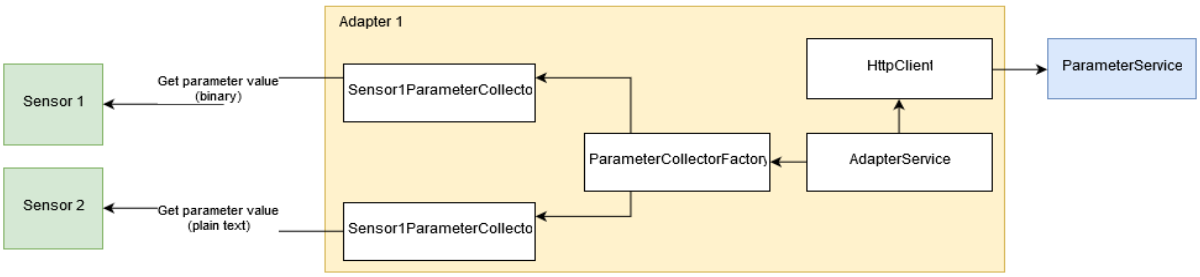


Рисунок 2 – Загальна архітектура сервісу «адаптер»

Кожен давач повертає свій формат даних, саме тому необхідно, щоб для кожного виду давача повинна бути розроблена своя реалізація загального абстрактного класу ParameterCollector з перевизначеним методом GetValue(). Тоді це дасть можливість зчитувати та конвертувати значення параметрів в єдиний формат перед відправкою їх на ParameterService.

Архітектура даного рішення представлена абстрактною, оскільки конкретна реалізація кожного адаптера може бути виконана в будь-якій формі програмного рішення. Таким чином, дане рішення може бути розроблене на різних мовах програмування і бути представлене як консольне, веб- або мобільне рішення залежно від потреб і можливостей середовища, в якому йому необхідно буде виконуватися. На базі цієї концепції було розроблено консольне рішення для адаптації даних, отриманих із давачів, та веб-рішення для адаптації даних, отриманих з пульта управління буровим процесом.

ParameterCollectorFactory розроблений за шаблоном проектування «абстрактна фабрика», що забезпечує інкапсуляцію окремих давачів під єдиним інтерфейсом, упускаючи їх реалізацію.

Кожна реалізація цього інтерфейсу матиме власний тип та реалізацію методу GetValue(). Вхідний параметр dataDictionary може бути використаний для передачі необхідних вхідних даних, наприклад, ключів доступу.

Клас AdapterContext містить необхідний набір даних для збору та відправки даних.

Однією з важливих складових роботи системи є збір та накопичення миттєвих значень параметрів та збереження їх у базі даних. Враховуючи великий об’єм даних, який постійно накопичуватиметься, постало питання збереження даних у NoSql типі баз даних, щоб забезпечити швидке збереження, а також, що дуже важливо, високу доступність цих даних за відповідними ключами.

ParameterService було розроблено як ASP.NET Web API рішення, що забезпечуватиме отримання даних із різних джерел по HTTP-протоколу. Пізніше було додано реалізацію для взаємодії між сервісами через інтеграційну шину даних (ESB), реалізовану на базі RabbitMQ. Це дозволяє забезпечити неявну взаємодію між сервісами. Таким чином, зберігається можливість масштабування сервісу незалежно від інших сервісів.

Обов’язками даного сервісу є:

– збереження миттєвих значень параметрів;

– витягування значень для заданих параметрів з додатковими часовими фільтрами;

– створення, редагування, видалення та витягування параметрів.

Створення кейсів є важливою складовою системи, але при цьому це не є постійно навантажуваною частиною системи. Таким чином, винесення цієї частини в окремий сервіс дозволить відділити і використовувати проект відповідно до потреб. Для збереження кейсів вибрано SQL-базу даних, оскільки на даний момент це повністю покриває вимоги швидкодії за умови дотримання основних принципів нормалізації та індексування SQL-баз даних.

Сервіс розроблено як ASP.NET Web API з застосуванням архітектурного стилю REST.

Обов’язками даного сервісу є створення, редагування, видалення та витягування кейсів.

Зважаючи на те, що рекомендаційні повідомлення повинні відправлятись на панель управління як тільки вони створюються, то необхідним було створення сервісу, який відповідав би за відправку повідомлень на різні пристрої за вимогою. Для даного рішення було вибрано server push технологію, реалізовану за допомогою Notification Hub service, розробленим компанією Microsoft, який дозволяє відправляти повідомлення з сервера на інші платформи.

Процес відправки повідомлень складається з двох етапів:

1. Пристрій реєструється на отримування push-повідомлень.

2. Відправка повідомлень на підписані пристрої.

Крім того, сервіс має можливість відправляти електронні листи на вибірку адрес як альтернативу push-повідомленням.

Однією із найзавантаженіших частин системи є CaseResolverService, яка відповідає за виконання кейсів на вимогу оператора або в автономному режимі за наявності відповідних конфігурацій системи. Причина високого навантаження є та, що для адекватної роботи кейсів необхідно опрацьовувати великий об'єм миттєвих значень.

Сам процес обробки одного кейса складається з трьох базових дій:

1. Попередня обробка даних (pre-actions).

2. Вирішення основної задачі кейсу для отримання вихідного результату (action).

3. Обробка вихідного результату та формування рекомендаційного висновку (post-actions).

Програмна реалізація CaseResolver-а відбувається з застосуванням CaseContext класу, що описує одиничний процес резолвінгу.

Попередня обробка даних – це не обов'язковий крок, але необхідний, якщо є потреба в підготовці значень перед вирішенням основної задачі, наприклад, зміна одиниць вимірювання значень параметра.

Обробка вихідного результату необхідна для формування зрозумілого повідомлення на основі вихідних параметрів.

Вхідний параметр CaseArgs має необхідний набір параметрів, щоб запустити процес вирішення кейсу. Цей аргумент використовується для створення CaseContext класу, який передаватиметься.

Такий підхід відкриває можливість написання невеликих стандартизованих програмних рішень, які виконуються безпосередньо під час роботи системи, без потреби рекомпіляції всього рішення.

В основі розробленої системи стоїть обробка параметрів, які описують певну властивість чи стан об'єкта або явища. Враховуючи, що параметр може описувати різні стани, властивості чи явища, було вирішено розробити окрему сутність Parameter, яка б описувала будь-який параметр із заданим діапазоном значень.

Було виділено два типи параметрів: числовий та текстовий.

Числовий тип параметрів має заданий діапазон значень у вигляді мінімального(min) і максимального(max) значень, а також додаткового параметра крок(step), який по замовчуванню приймає значення одиниці (1). Таким чином, встановлені межі описують можливі діапазони значень для кожного числового параметра.

Текстові параметри передбачають можливість перелічування можливих значень для заданого параметра, що дозволяє оператору вибирати один або декілька значень у процесі створення відповідного кейсу.

Кожен параметр повинен належати до певної категорії параметрів. Загалом виділяються такі категорії: динамічно керовані, статично керовані, збурюючі та вихідні.

Згідно з розробленим підходом, кожному параметру присвоюється свій унікальний ідентифікатор, який використовується в системі адаптерів для збору миттєвих значень у процесі буріння.

В окремому вікні можна переглянути всі створені параметри з описом.

База даних кейсів представляє собою колекцію, кожен елемент якої описує певний випадок за допомогою очікуваних вхідних та вихідних параметрів. При цьому для створення кейсу важливим є тільки опис параметрів, а не їх конкретних значень.

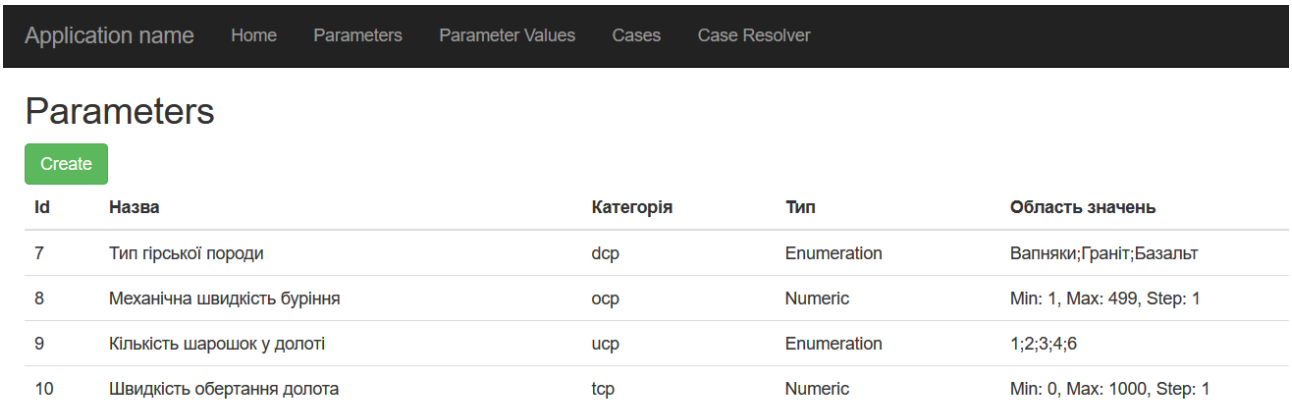


Рисунок 3 – Вікно перегляду параметрів

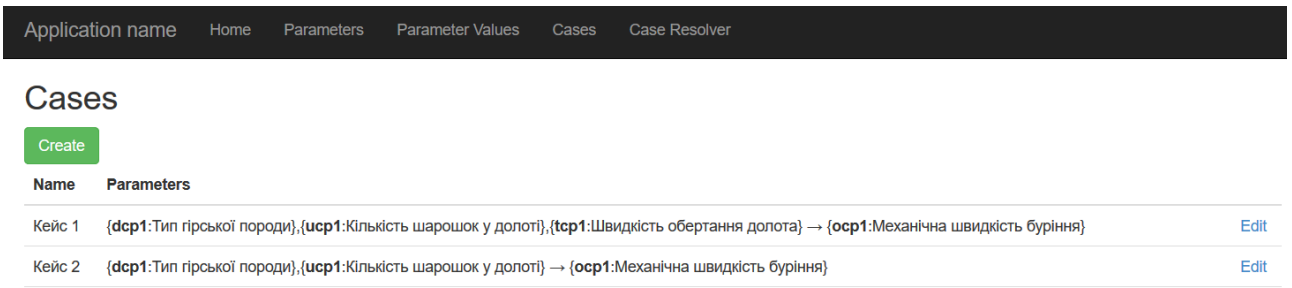


Рисунок 4 – Вікно перегляду кейсів

На рисунку представлено два випадки, кожен з яких описує два набори даних: вхідні (dcp, ucp, tcp) та вихідні (ocp). При цьому для кожного кейсу кожен параметр може приймати звужений діапазон значень у межах свого глобального діапазону, описаного при створенні параметра.

На рисунку представлено вікно створення нового кейсу з двома вхідними та одним вихідним параметром. Кожен параметр має заданий піддіапазон значень у межах глобального діапазону.

Створені кейси можуть бути використані в вікні Case Resolver для знаходження співпадінь по заданому діапазону.

Релевантність кейсів є важливою складовою системи, що забезпечується шляхом накладання на кейси великих об'ємів отриманих значень у процесі буріння з різних бурових комплексів.

Накладання відбувається шляхом порівняння отриманих значень із вхідними та вихідними значеннями кожного кейсу для встановлення рівня відповідності отриманих значень у процесі буріння з очікуваними значеннями, описаними в кожному кейсі. Результатом накладання є отримане відсоткове значення релевантності даного кейсу для поточної бурової установки в певний момент часу.

При цьому релевантність кейсу відносно бурової установки описується як середнє відсоткове значення всіх накладань кейсу.

Таким чином, кожен кейс має набір релевантностей для кожної установки, що дає можливість оцінювати якість кейсу, а також проводити пошук кейсів за релевантністю в базі знань.

На рисунку кожен рядок представляє набір значень різних параметрів, отриманих за один день.

Таким чином, після накладання даного кейсу до кожного часового періоду отримано результат, зображений на рис. 5.

Дата	Релевантність, %	Опис
2018-06-06 12:00:00	100	
2018-06-07 12:00:00	95	Оскільки швидкість обертання перевищує очікувану
2018-06-08 12:00:00	15	Порода інша, швидкість нижча і результат нижчий очікуваного
2018-06-09 12:00:00	78	Оскільки швидкість обертання суттєво перевищує очікуване значення. Результат вищий ніж очікуваний
Середнє значення	72	

Рисунок 5 – Релевантність кейсу для різних часових періодів

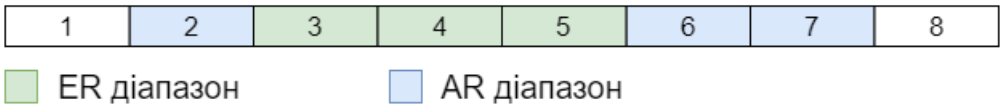
Отже, для вибраної бурової релевантність даного кейсу становить 72%, що може бути використано під час пошуку кейсів із виведенням за релевантністю.

Також, за допомогою релевантності кейсу в заданий момент часу можливо встановлювати межі, за яких необхідно повідомляти операторам про відповідність отриманих значень до очікуваних у кейсі. Це має сенс у режимі реального часу при накладанні кейсів, які мають повідомлення про небезпеку або некоректні налаштування бурової установки, що може призвести до надмірного зношення, зупинки чи виведення з ладу бурової.

Враховуючи той факт, що ідеальні співпадіння під час пошуку зустрічаються надто рідко, було розглянуто пошук наближення шуканого результату(ER, expected result) до фактичних діапазонів значень (AR, actual result), описаних у кейсах.

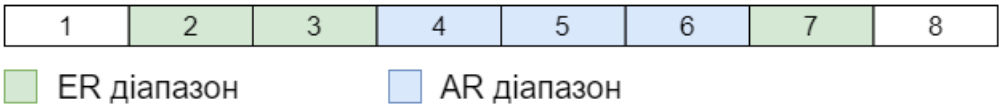
Пошук співпадінь та визначення наближеності шуканого результату до фактичного для числових типів параметрів описується за одним з 6 сценаріїв поведінки:

1. Діапазон значень ER входить у діапазон значень AR. Це відбувається, якщо діапазон значень ER повністю входить у діапазон AR. У такому випадку можна говорити про повне співпадіння шуканого результату для вибраного кейсу.



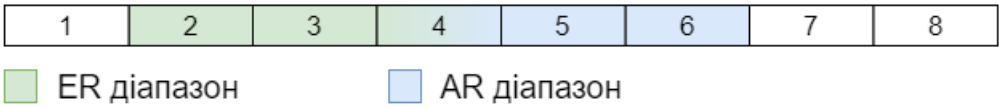
Сценарій, коли діапазон значень ER входить у діапазон значень AR

2. Діапазон значень AR входить у діапазон значень ER. У такому випадку можна говорити про частковий результат, де необхідно вирахувати, який відсоток AR входить у діапазон значень ER.

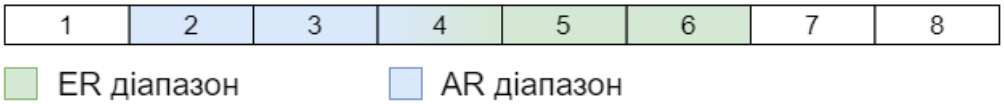


Сценарій, коли діапазон значень AR входить у діапазон значень ER

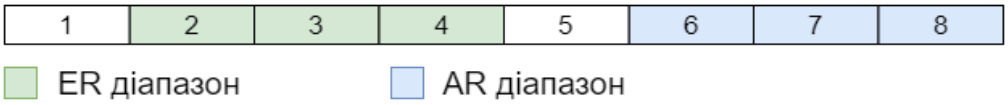
3. Часткове співпадіння, де ER зліва, описується як часткове перетинання крайніх правих значень ER відносно крайніх лівих значень AR.



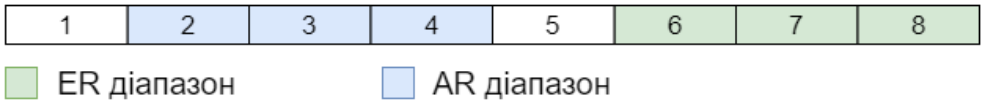
4. Часткове співпадіння, де ER справа, описується як часткове перетинання крайніх лівих значень ER відносно крайніх правих значень AR.



5. Немає співпадінь, де ER зліва. У даному сценарії релевантність параметра визначається як відсоткове наближення крайніх правих значень ER відносно крайніх лівих значень AR.



6. Немає співпадінь, де ER справа. Визначається як відсоткове наближення крайніх лівих значень ER відносно крайніх правих значень AR.



Для текстових діапазонів значень повним співпадіння вважається тільки тоді, коли всі значення ER входять у діапазон значень AR, а частковим – якщо тільки частина значень діапазону ER входить у діапазон значень AR. При цьому відсоткове значення вираховується пропорційно відносно кількості співпадінь до загального розміру діапазону значень ER.

Програмна реалізація кожного сценарію представлена реалізацією інтерфейсу IRelevantResolver, лістинг якого представлено нижче:

```
    public interface IRelevantResolver
    {
        /// <summary>
        /// Used to specify the ParameterType of the scenario
        /// </summary>
        ParameterType ParameterType { get; }

        /// <summary>
        /// Used to describe scenario
        /// </summary>
        /// <param name="actualParameter">AR</param>
        /// <param name="expectedParamater">ER</param>
        bool IsValid(ResolveParamater actualParameter, ResolveParamater
expectedParamater);

        /// <summary>
        /// Use to find relevation of the expected parameter to actual
parameter
        /// </summary>
        /// <param name="actualParameter">AR</param>
        /// <param name="expectedParameter">ER</param>
        /// <returns></returns>
        double Resolve(ResolveParamater actualParameter,
          ResolveParamater expectedParameter);
    }
```

Поле ParameterType використовується для встановлення типу параметра для поточного сценарію, що дозволяє легко виділити всі можливі сценарії з загальної колекції по типу параметра для діапазону значень ER, що дозволяє уникнути зайвих перевірок.

Метод IsValid() використовується для знаходження необхідного сценарію. Таким чином, якщо результат виконання методу є True, то необхідний сценарії знайдено і відбувається виконання методу Resolve(), в іншому випадку відбувається перехід до наступного сценарію.

Метод Resolve() порівнює два діапазони значення й визначає рівень відповідності між ними у відсотковому відношенні, який повертається як цілочисельне значення.

Псевдокод пошуку та резолвінгу для діапазону значень виглядає таким чином:

Ініціалізація параметрів відбувається через IoC-контейнер, що дозволяє динамічно створювати екземпляри.

Результат резолвінгу представлено у вигляді 100 кейсів із найвищими показниками релевантності. При потребі оператор може змінити кількість результуючих значень, але при цьому резолвінг відбуватиметься повторно.

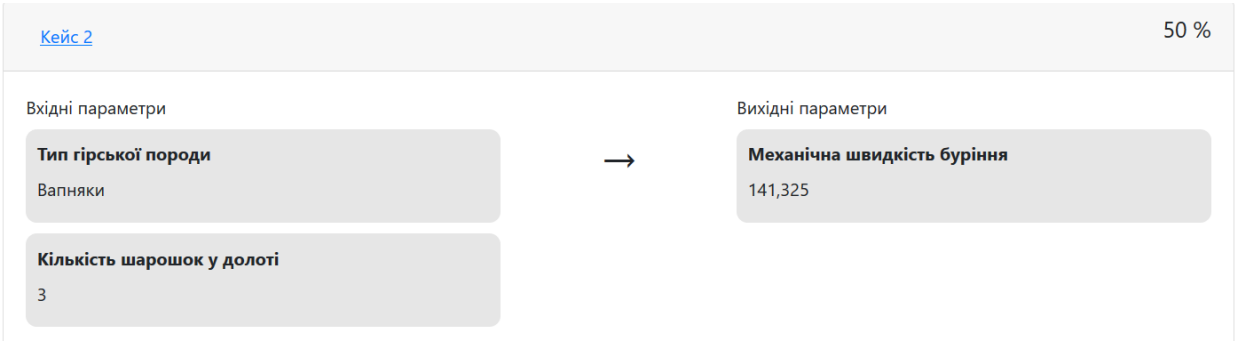


Рисунок 6 – Виведення результатів резолвінгу

Також, оператор може відразу переглянути кожен результуючий кейс у списку результатів.

Важливим пунктом роботи системи є підтримка миттєвих повідомлень для кейсів, які необхідно відправляти, якщо релевантність кейсу досягла наперед визначеної відмітки.

В автономному режимі це дає можливість надавати рекомендації операторам у часі, наближеному до реального, без необхідності мануального введення значень для резолвінгу.

Кожен кейс може мати одне чи більше повідомлень з різними рівнями релевантності.

Крім того, при створенні повідомлення можна зазначати тип повідомлення, що допомагатиме встановлювати додаткові кольорові чи звукові сигнали, щоб оператор міг швидше реагувати на важливі повідомлення.

Загалом виділяються 3 типи повідомлень:

• інформативне – встановлюється, якщо повідомлення несе суто рекомендаційний характер;

• попереджувальне – встановлюється, якщо повідомлення несе попередження про можливі критичні ситуації;

• критичне – встановлюється для кейсів, вихідними результатами яких є опис критичних ситуацій.

IdentityServer

Захист даних та методів роботи системи є важливою складовою в нафтогазовій галузі, оскільки вразливість безпеки може призвести до витоку даних про нафтогазові об’єкти або до виведення з ладу чи зупинки роботи бурової установки. Для захисту даних і забезпечення зручного способу авторизації та аутентифікації було вирішено використати технологію IdentityServer 4, яка є реалізацією стандарту OpenID, що дає можливість створювати

єдиний обліковий запис для аутентифікації на багатьох інтернет-ресурсах, що можуть бути не зв'язані між собою.

3. Висновки

Таким чином, центральним елементом створеної системи є підтримка миттєвих повідомлень для кейсів, які необхідно відправляти, якщо релевантність кейсу досягла наперед визначеної відмітки. Це особливо актуально для резолвінгу кейсів, де вихідні значення яких несуть попереджувальний чи аварійний характер. Кожен кейс може мати одне чи більше повідомлень з різними рівнями релевантності. Створені кейси можуть бути використані для знаходження співпадінь за заданим діапазоном. Релевантність кейсів є важливою складовою системи, що забезпечується шляхом накладання на кейси великих об'ємів отриманих значень у процесі буріння із різних бурових комплексів. Накладання відбувається шляхом порівняння отриманих значень із вхідними та вихідними значеннями кожного кейсу для встановлення рівня відповідності отриманих значень у процесі буріння з очікуваними значеннями, описаними в кожному кейсі. Результатом накладання є отримане відсоткове значення релевантності даного кейсу для поточної бурової установки в певний момент часу. При цьому релевантність кейсу відносно бурової установки описується як середнє відсоткове значення всіх накладань кейсу. Таким чином, кожен кейс має набір релевантностей для кожної установки, що дає можливість оцінювати якість кейсу, а також проводити пошук кейсів за релевантністю в базі знань.

СПИСОК ДЖЕРЕЛ

1. Abdrabou E. A. M. L., Salem A.-B. M. Case-based reasoning tools from shells to object-oriented frameworks. *InICCOMP*. 2008. Vol. 67. P. 781–786.
2. Bergmann R., Schaaf M. Structural case-based reasoning and ontology-based knowledge management: A perfect match. *Journal of Universal Computer Science*. 2003. Vol. 9, N 7. P. 608–626.
3. Bierer A., Hofmann M. Dimensions of case-based reasoner quality management / ed. L. McGinty, D. Wilson. *Case-Based Reasoning Research and Development*: Lecture Notes in Computer Science. Springer Berlin Heidelberg. 2009. Vol. 5650. P. 90–104. DOI: 10.1007/978-3-642-02998-1 53.
4. Gronau N., Laskowski F. Using case-based reasoning to improve information retrieval in knowledge management systems / ed. E. Menasalvas, J. Segovia, P. Szczepaniak. *Advances in Web Intelligence* : Lecture Notes in Computer Science. Springer Berlin Heidelberg. 2003. Vol. 2663. P. 94–102. DOI: 10.1007/3-540-44831-49.
5. Hüllermeier E. Case-Based Approximate Reasoning. Berlin: Springer, 2007. – 63 p.
6. Hüllermeier E. Schlegel P. Preference-based CBR: first steps toward a methodological framework. *In Proceedings of the 19th International Conference on Case-Based Reasoning Research and Development:* ICCBR'11: Springer-Verlag, 2011. P. 77–91. DOI: 10.1007/978-3-642-23291-6_8 38,63.
7. Sheketa V.I., Melnyk V.D., Romanyshyn Y.L., Chesanovskyy M.S. The Construction of Technological Problems Cases for the Purpose of Intelligible Control. *Perspective technologies and methods in MEMS design (MemsTech'2016)*: XIIth International Conference (Lviv-Polyana, 20–24 th April 2016). Lviv, 2016. P. 96–99.
8. Sheketa V.I., Pikh V.Y., Styslo T.R., Chesanovskyy M.S. Modeling methodology for knowledge-based systems of wells drilling control. *Perspective technologies and methods in MEMS design (MemsTech'2017).* XIIIth International Conference (Lviv–Polyana, 20–23 th April 2017). Lviv, 2017. P. 56–58.
9. Barták R., Dechter R. Constraint Processing. New-York: Morgan Kaufmann Publisher, 2003. 210 p.
10. Tsang E. Foundations of Constraint Satisfaction. London and San Diego: Academic Press, 1993. 421 p.

Стаття надійшла до редакції 26.10.2018