УДК 681.518.2:622.026

РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КОМП'ЮТЕРНОЇ СИСТЕМИ ОПТИМАЛЬНОГО КЕРУВАННЯ ПРОЦЕСОМ БУРІННЯ

В.Б.Кропивницька, Б.В.Клим, Т.В.Гуменюк

ІФНТУНГ, 76019, Івано-Франківськ, вул. Карпатська, 15, тел. (03422)
e-mail: p u b l i c @ n u n g . e d u . u a

Решаются вопросы построения программного обеспечения для компьютерной системы оптимального управления процессом бурения нефтяных и газовых скважин, оформленного в виде отдельного оптимизационного модуля. Разработана структуная схема взаимодействия оптимизационного модуля с комплексом СКУБ-М2, что упрощает процесс его интеграции в существующее программное обеспечение.

The questions of software development for computer collection of process control of drilling of oil and gaseous access boreholes, what is designed as a separate optimization module, are being decided in the article. The flow diagram of co-operation of the optimization module is designed with a complex SKUB-M2, which simplifies the process of its integration in existent software.

В період переходу до ринкової економіки зросли вимоги до ефективності проведення бурових робіт. Збільшення вартості обладнання, інструменту поставило нові вимоги до технології буріння, що значно підвищило перспективу застосування оптимальних автоматизованих систем контролю та керування цим процесом в умовах апріорної невизначеності. Цій проблемі присвячена ціла низка робіт як вітчизняних, так і зарубіжних дослідників, які, в основному, вирішували проблему оптимального керування процесом буріння з використанням методів на основі детерміновано-статистичних моделей. Значно меншу увагу дослідники приділяли можливості технічної реалізації оптимальних алгоритмів, що в окремих випадках призводить до неможливості їх застосування. В роботі [3] була поставлена і розв'язана задача оптимального керування з дискретно-неперервною зміною керувальних дій, яка дає можливість спростити технічну реалізацію оптимальних керувальних дій, завдяки чому досягається здешевлення і прискорення будівництва свердловині.

Дана стаття присвячена впровадженню оптимальної системи керування процесом буріння свердловин з дискретно-неперервними керувальними діями в комплекс засобів наземного контролю і керування процесом буріння СКУБ-М2, який являє собою апаратні засоби інформаційного забезпечення персоналу бурових установок експлуатаційного і глибокого розвідувального буріння нафтових і газових свердловин [1].

Програмне забезпечення, розроблене для розв'язання задач оптимального керування [2, 3], оформлено у вигляді окремого оптимізаційного модуля, що дає змогу легко інтегрувати його в існуючий пакет прикладних програм.

Взаємодія комплексу СКУБ-М2 та оптимізаційного модуля пояснюється структурною схемою, наведеною на рис. 1.

Контролер ПЗОД формує масив даних про процес буріння, який передається інтерфейсом RS-485 на пристрій реєстрації інформації. Отримана інформація реєструється і архівується в базі даних (БД) для збереження виміряних та розрахованих технологічних параметрів процесу буріння. Для розв'язання задач ідентифікації та оптимального керування з БД вибираються такі вхідні параметри: набір значень осьового навантаження на долото, швидкість обертання долота, час та проходка, які передаються до оптимізаційного модуля.

Для розроблення програмного забезпечення збору і обробки даних контролером застосовано систему UltraLogic [5], основна мова програмування якої це – мова функціональних блокових діаграм Function Block Diagram. Вона вважається потужною і в той же час простою в користуванні інструментальною системою, яка не тільки скорочує час проектування програми для контролера відповідно до замовлення споживача, але і забезпечує доробку програми в умовах експлуатації її розробником, а за необхідністю – споживачем. Система надає користувачеві можливість виконувати програмні модулі іншими мовами програмування (Сі, Паскаль, Асемблер): до системи входять функціональні блоки, які забезпечують роботу з масивами, що дозволяє організувати реєстрацію і аналіз швидкоплинних процесів. Контролери та комп'ютери, що програмуються на UltraLogic, можуть об'єднуватися в мережі передачі даних на базі інтерфейсу RS-485 і Ethernet. Важливою особливістю системи є те, що пакет не має обмежень щодо кількості змінних, каналів вводу-виводу і розроблених програм. До складу системи входить DDE сервер, призначений для зв'язку з будь-якими Windows додатками [4, 5].

З метою інтеграції оптимізаційного модуля в інформаційно вимірювальний комплекс, на інструментальній панелі останнього було додано піктограму '*Консультація*'. Після натискання на цю піктограму програмного комплексу IVK відкривається діалогове вікно '*Вибір таблиць бази даних*', у полях якого необхідно вка-

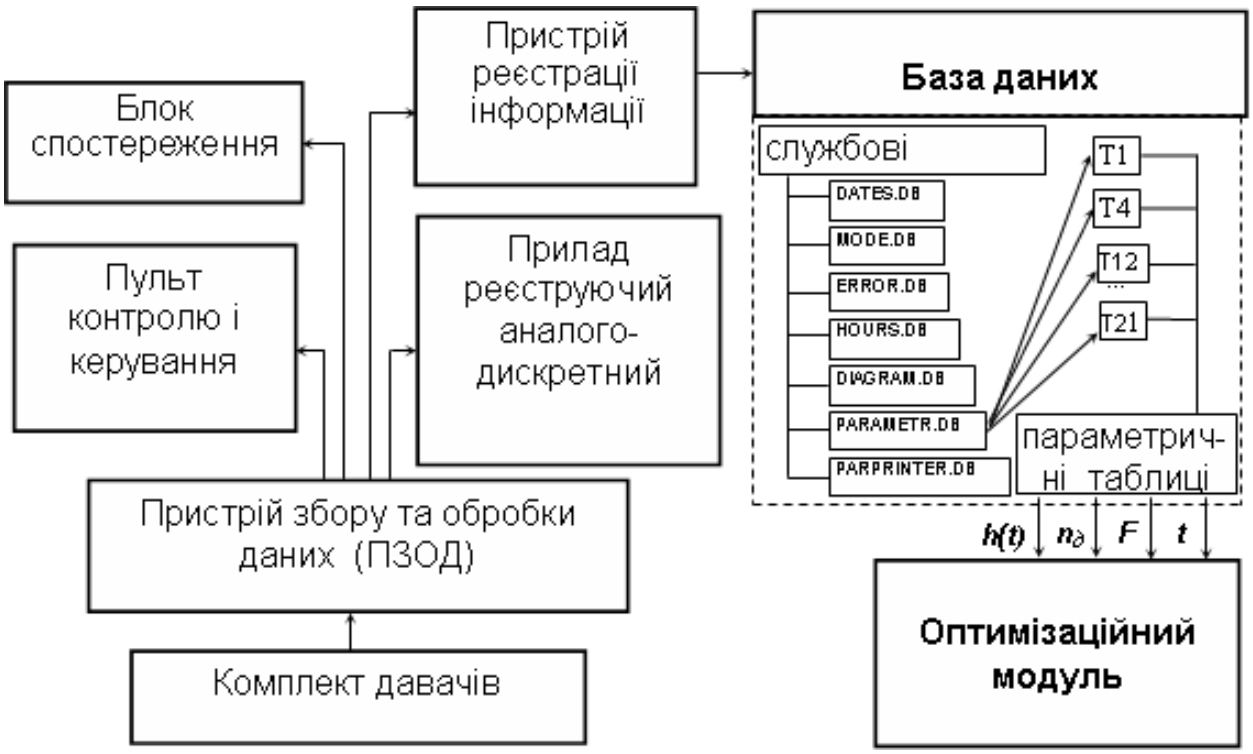

Рисунок 1 — Структурна схема взаємодії оптимізаційного модуля з комплексом СКУБ–М2

зати шляхи до таблиць із наборами значень відповідних параметрів.

Описаний етап роботи оптимізаційного модуля реалізується так званим інтерфейсним блоком, завдання якого полягає у веденні діалогу з користувачем та одержанні від нього необхідної інформації (набору шляхів до таблиць баз даних із значенням параметрів) і подальшого збереження її у внутрішніх структурах даних програми.

Для створення діалогових вікон використовується стандартний клас MFC (Microsoft Foundation Class) *CDialog*, базовий для наслідування. Створення, власне, графічного представлення діалогового вікна здійснюється в редакторі ресурсів шляхом розміщення на готовий шаблон необхідних елементів керування.

Кожен елемент управління зв'язується із відповідним членом класу похідним від *CDialog*, тобто значення кожного створеного елемента управління зберігається в елементах даних створеного класу діалогового вікна. Поновлення відповідних структур даних здійснюється автоматично у момент закривання вікна шляхом натискання кнопки *'OK'*, або у випадку виконання команди UpdateData(TRUE).

Для забезпечення можливості відображення останніх введених користувачем шляхів, використовується зворотня передача даних – від членів класу до елементів управління, за допомогою команди UpdateData(FALSE).

Обробка даних після натискання на кнопку '*Огляд*' діалогового вікна '*Вибір таблиць бази даних*' реалізовується за допомогою стандартного механізму обробки подій графічними засобами самого середовища розробки програми.

Діалогове вікно '*Вибір таблиці*' створюється на основі стандартного MFC класу діалогового вікна відкриття/збереження файлів *CFileDialog*. Шлях до вибраного користувачем файлу в даному діалоговому вікні можна одержати викликавши метод GetFileName() для об'єкта класу *CFileDialog*. Після цього одержане значення присвоюється відповідному елементові класу діалогового вікна '*Вибір таблиць бази даних*' і реалізовується поновлення значення елемента управління *EditBox* шляхом виконання команди UpdateData(FALSE).

На наступному етапі роботи програми запускається блок доступу до баз даних (БДБД), який зчитує значення введених користувачем полів, під'єднується до зазначених таблиць архівних даних програмного комплексу СКУБ та вибирає із них набори значень відповідних параметрів (навантаження на гак, частота обертання та положення талевого блоку відповідно), записуючи їх у тимчасовий бінарний файл такого формату:

```
число значень параметра F
значення_1
значення_2
   ...
значення_Fk
число значень параметра N
значення_1
значення_2
   ...
значення_Nk
число значень параметра H
значення_1
значення_2
   ...
значення_Hk
```

Варто зазначити, що під час запису даних у бінарний файл програма здійснює узгодження розмірностей, зокрема перетворює частоту обертання із обертів за хвилину в оберти за секунду, а також реалізує обчислення величини проходки на основі положення талевого блоку.

Наведемо приклад під'єднання до таблиці бази даних із наборами значень осьового навантаження (F), яке здійснює БДБД:

```
try {
 CDatabase db_F;
 CString connect;
 connect+="DRIVER={Driver do
     Microsoft Paradox(*.db)}";
 connect+="DBQ=";
 connect+=path_to_F; // шлях
       від інтерфейсного модуля
 db.OpenEx(connect);
}
catch (CDBException *xcp) {
 cerr << T("Database
                  exception: ")
      << (const TCHAR *)
       xcp->m_strError << endl;
 xcp->Delete();
}
```

Як бачимо із наведеного фрагменту програми, для обробки помилок, що можуть виникати під час під'єднання до бази даних, використовується механізм виключень (exception). Обробка виключення полягає у виведенні деякого діагностичного повідомлення, що пояснює характер помилок і дає можливість користувачеві виконати правильні дії.

Для інкапсуляції записів таблиці бази даних в MFC використовується клас CRecordset, від якого породжується похідний клас із заданням джерела даних ODBC. Після виконання такого наслідування похідний клас включатиме структури даних, що відображатимуть вміст відповідних стовпчиків заданої таблиці.

Таким чином, завдання БДБД полягає в переборі записів таблиць за допомогою класу похідного від CRecordset і запису наборів значень параметрів у бінарний файл.

Переміщення записами таблиці реалізується за допомогою методу MoveNext() класу CRecordset, при цьому у випадку досягнення кінця таблиці метод IsEOF() цього ж класу повертає значення TRUE. Альтернативою IsEOF() є метод GetRecordCount(), що повертає число записів в таблиці, з якою зв'язаний об'єкт класу CRecordset.

Наведемо фрагмент програмного коду БДБД, що заносить до бінарного файлу input.bin набір значень параметрів осьового навантаження, який зчитується із відповідної таблиці:

```
ofstream obf ("input.bin",
       ios::out | ios::binary |
                  ios::trunc);
int count_F = m_table_
          F->GetRecordCount();
obf.write((char *) &count_F,
                  sizeof(count));
for (int i=0; i<count_F; i++) {
    obf.write((char *)
             &m_table_F->value,
              sizeof(float));
    m_table_F->MoveNext();
}
```

Описані дії повторюються і для інших об'єктів CRecordset, зв'язаних із таблицями баз даних з наборами значень частоти обертання та положення талевого блоку. Кінцеві результати кожного разу дописуються в кінець бінарного файлу input.bin.

Сформувавши необхідний бінарний файл, БДБД більше не потребує доступу до баз даних і закриває їх за допомогою методу Close(), що викликається для кожного відкритого об'єкта класу CDatabase.

Завершальним етапом роботи БДБД є передача управління математичному блоку, що запускається як окремий процес за допомогою API-функції CreateProcess. Основними параметрами даної функції є:

– ім'я виконавчого файлу, що запускається як окремий процес;

– структура типу STARTUPINFO, що містить інформацію про параметри запуску процесу;

– структура типу PROCESS_INFORMATION, що заповнюється інформацією про процес після його запусканні.

Основними елементами структури STARTUPINFO, на які варто звернути увагу, є:

– dwX – розташування вікна, в якому запускатиметься програма, по горизонталі відносно верхнього лівого кута екрана;

– dwY – розташування вікна, в якому запускатиметься програма, по вертикалі відносно верхнього лівого кута екрана;

– dwXSize – ширина вікна програми;

– dwYSize – висота вікна програми;

– dwFlags – прапорець, що визначає параметри початкового відображення вікна програми.

Після виконання зазначених дій запускається математичний блок оптимізаційного модуля програми, який є окремою самостійною процедурою із графічним інтерфейсом. Він створений засобами математичного пакету MatLab та перекладеним на мову C++ вбудованим компілятором системи mcc (MatLab C Compiler).

Оскільки алгоритми, що реалізують розрахунок оптимальних параметрів процесу буріння, реалізовані у вигляді окремих файлів-сценаріїв системи MatLab, що повинні запускатися в строго визначеній послідовності, то для їх консолідації в єдине ціле на базі існуючого графічного інтерфейсу використовуються наявні вбудовані засоби даного математичного пакету.

Як головний сценарій, навколо якого об'єднуються всі інші модулі, вибрано процедуру початкової ініціалізації `IndDril`. На по-

чатковому етапі роботи описаний сценарій зчитує із бінарного файлу `input.bin`, сформованого БДБД, набори значень параметрів осьового навантаження, проходки та швидкості обертання та поміщає їх у власні внутрішні структури даних – масиви.

Враховуючи наведений раніше формат бінарного файлу, що генерується БДБД, зчитування значень відповідних параметрів здійснюється такою послідовністю команд сценарію `IndDril` середовища Matlab:

```
fid = fopen('input.bin', 'rb');
[N, count] =
      fread(fid, 1, 'float64');
[F, count] =
 fread(fid, [N, 1], 'float64');
[N, count] =
      fread(fid, 1, 'float64');
[n, count] =
 fread(fid, [N, 1], 'float64');
[N, count] =
      fread(fid, 1, 'float64');
[h, count] =
 fread(fid, [N, 1], 'float64');
fclose(fid);
```

Створення графічного інтерфейсу для відображення результатів обчислень та забезпечення можливості діалогової взаємодії з користувачем здійснюється за допомогою типових елементів управління, що підтримуються будь-якою високорівневою мовою програмування.

Для створення вікна головної програми в середовищі Matlab використовується функція `figure`, параметрами якого є координати початкового розташування вікна, тип границь, заголовок вікна тощо. Як результат своєї роботи функція повертає дескриптор вікна, який надалі використовуватиметься під час створення елементів управління.

Безпосереднє створення елементів управління в середовищі MatLab здійснюється за допомогою команди `uicontrol`, яка має такі параметри:

`Style` – визначає тип елемента управління;

`text` – статичний текст;

`pushbutton` – кнопка з реакцією на натискання;

`listbox` – список;

`checkbox` – незалежний прапорець;

`edit` – поле вводу та редагування;

`frame` – створення рамки;

`radiobutton` – група залежних прапорців.

`BackgroundColor` – задає колір фону елемента управління;

`Position` – визначає координати розташування елемента управління в межах батьківського вікна;

`String` – задає надпис на елементі управління;

`FontSize` – визначає розмір шрифта;

`Visible` – прапорець відображення/не відображення елемента;

`Enable` – прапорець блокування елемента управління;

`Callback` – задає функцію, що викликатиметься при взаємодії користувача із елементом управління.

Таким чином, описані параметри функції `uicontrol` дають можливість створити елемент управління необхідного класу і налаштувати його властивості відповідно до потреб користувача. Графічний інтерфейс математичного модуля включає елементи управління всіх описаних вище типів.

Для динамічної зміни графічного інтерфейсу в сценарії IndDril використовується функція `set`, що дає змогу змінювати властивості елементів управління в процесі роботи. Таким чином, характер інтерфейсу математичного модуля базується на частковому відображенні множини всіх елементів управління в кожен конкретний момент часу в залежності від дій користувача. Це досягається за допомогою динамічної зміни властивості `Visible` для кожного елемента управління. Так, наприклад, для того, щоб тимчасово "сховати" список, кнопку та поле для редагування, в сценарії IndDril використовується така послідовність команд:

```
set(hList(i),'Visible','off');
set(hButton(i),'Visible','off');
set(hEdit(i),'Visible','off');
```

Очевидно, що розвинений графічний інтерфейс передбачає наявність величезної кількості елементів управління. Для полегшення роботи із множинами таких елементів у головному сценарії математичного модуля використовується методика, за якою дескриптори однотипних елементів управління зберігаються у вигляді масивів, що дає можливість використовувати типові циклічні структури для доступу до великої групи елементів. Прикладом використання такої методики є "приховування" нижньої навігаційної панелі головного вікна математичного модуля, що складається із групи залежних прапорців, в допоміжній функції `clearall`:

```
for i=1:11
set(hRadio(i),'Visible','off');
end;
```

Результати роботи математичного модуля відображаються не лише у вигляді текстової інформації (за допомогою списків), а й у вигляді графіків. Для забезпечення такої можливості в сценарії IndDril використовуються спеціальні об'єкти `axes`, що являють собою форму із розміткою для відображення графіків. З кожним із таких об'єктів асоціюється окремий об'єкт-графік, дескриптор якого повертається функцією `plot`. Варто зазначити, що вивід будь-якого графіка завжди здійснюється в поточний об'єкт `axes`.

Таким чином, процес відображення графічної інформації умовно можна розбити на такі кроки:

1) вибір необхідної форми (об'єкта `axes`) для відображення графіка;

2) безпосередня побудова графіка за допомогою функції `plot` та збереження його дескриптора;

3) відображення у вікні відповідної форми та графіка за допомогою встановлення властивості `Visible` в `on` функцією `set`.

Реакція елементів управління головного вікна математичного модуля на дії користувача здійснюється на основі типового механізму обробки подій. Властивість `Callback` кожного елемента управління задає функцію, що використовується для обробки подій від даного елемента.

Для створення виконавчого файлу на основі сценаріїв математичного пакету MatLab, використовується вбудований компілятор mcc, параметрами якого є імена усіх складових файлів проекту, при цьому основний сценарій вказується першим.

Таким чином, створення exe-файлу на основі сценаріїв математичного блоку здійснюється за допомогою командного рядка виду:

```
mcc -m -B sglcpp -L Cpp IndDril
ident1 ident2 check optimdril
clearall dril_subcont_1 spid_orto
onexit rad radoptim fun_sum
fun_ind fun_VV fun_VK fun_HP
fun_start fun_s fun_fs fun_pr
fun_pr1 fun_SubCon_1 fun_g fun_V0
fun_Ke fun_Kq hamfun fun_tb
fun_Ke1 fun_V01 fun_Kq1
```

Отже, завдання БДБД передачі управління математичному блоку фактично полягає в запусканні одержаного exe-файлу. Сценарії математичного модуля зчитують набори значень параметрів із відповідного бінарного файлу, сформованого на початковому етапі БДБД, і реалізують подальшу їх обробку, відображаючи результати у вигляді графіків, значень визначених параметрів та рекомендацій.

Після завершення роботи користувача із математичним модулем, керування передається блоку повернення, який реалізує коректне завершення роботи всього оптимізаційного модуля, закриваючи відповідні файли-потоки та з'єднання з базами даних, видаляючи створений БДБД бінарний файл для передавання і інформації в сценарії MatLab, і передає керування програмному інформаційно-вимірювальному комплексові.

Література

1 Вощинский В.С., Ролик В.А. Модернізований комплекс засобів наземного контролю і керування процесом буріння нафтових і газових свердловин СКУБ-М2 // Нафтова і газова промисловість. – 2004. – №3. – С. 24-29.

2 Горбійчук М.І., Кропивницька В.Б. Оптимальне керування процесом механічного буріння // Нафтова і газова промисловість. – 2005. – №3. – С. 20-22.

3 Горбійчук М.І., Кропивницька В.Б. Субоптимальне керування процесом заглиблення свердловин // Нафтова і газова промисловість. – 2003. – №1. – С. 24-25.

4 Камелин А. Автоматизированная система управления стендами тестирования погружного електрооборудования // Нефтегазовая промышленность. – 2004. – № 3. – С. 16-23.

5 Патрахин В.А. Средства программирования PC-совместимых контроллеров // Программное обеспечение. – 2003. – №1-2. – С.34-40 .