УДК 004.4'236:004.75:621.382:681.3/.5
Г.І. Воробець, О.І. Воробець, кандидати фіз.-мат. наук,
Р.В. Рогов,

СТРУКТУРНО–ЛОГІЧНА ОРГАНІЗАЦІЯ СЕРВЕРНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ УПРАВЛІННЯ ТЕХНОЛОГІЧНИМИ МЕРЕЖАМИ

Анотація. *Розглянуто внутрішню структурну реалізацію мережевої системи контролю та управління технологічними процесами, побудовану за принципами SCADA-системи. Описано програмний транслятор скриптового коду. Запропоновано алгоритм підключення модулів розширення для забезпечення гнучкості впровадження системи у виробництво.*

Ключові слова: *технологічна мережа, транслятор скриптового коду, система управління.*

G.I. Vorobets, PhD, **A.I. Vorobets,** PhD
R.V. Rogov,

THE STRUCTURE AND LOGIC ORGANIZATION OF THE SERVER SOFTWARE FOR MANAGEMENT OF TECHNOLOGICAL NETWORKS

Abstract. *It is presented internal structural realization of the network monitoring system and management of the technological processes, constructed on SCADA-system principles. It is described realization of the program compiler of a script code and the algorithm of realization connections of modules of expansion for maintenance of flexibility of introduction of system in manufacture is offered.*

Keywords*: a technological network, the compiler of a script code, a control system.*

Г.И. Воробец, А.И. Воробец, кандидаты физ.-мат. наук
Р.В. Рогов,

СТРУКТУРНО–ЛОГИЧЕСКАЯ ОРГАНИЗАЦИЯ СЕРВЕРНОГО ПРОГРАМНОГО ОБЕСПЕЧЕНИЯ ДЛЯ УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМИ СЕТЯМИ

Аннотация. *Рассмотрена внутренняя структурная реализация сетевой системы контроля и управления технологическими процессами, построенная по принципам SCADA-системы. Описан программный транслятор скриптового кода. Предложен алгоритм подключения модулей расширения для обеспечения гибкости внедрения системы в производство.*

Ключевые слова*: технологическая сеть, транслятор скриптового кода, система управления.*

Вступ. Технологічна мережа – це об'єднання декількох технологічних вузлів у єдину систему для виконання конкретного вдання, під управлінням центрального комп'ютера-сервера. Більш узагальнено – це клієнт-серверний варіант реалізації SCADA-системи (SCADA – Supervisory Control and Data Acquisition). Принципи роботи такої мережі детально описано в [3].

Метою даної роботи є створення програмного забезпечення серверної частини для контролю та управління технологічними процесами за допомогою засобів локальних мереж.

Результати дослідження. Згідно з вимогами до SCADA-систем [1], технологічна мережа вирішує наступні задачі (рис.1) [3]:

обмін даними з технологічними модулями, які реалізують на основі локальних ЕОМ (ЛЕОМ) або спеціалізованих мікрокон тролерів і призначені для безпосереднього управління технологічним об'єктом (ТО) через відповідні пристрої спряження (ПС);

обробку технологічної інформації в режимі реального часу;

відображення інформації на екрані у зручній для користувача формі;

підтримку бази даних із технологічною інформацією в реальному часі;

аварійну сигналізацію при виникненні нештатних ситуацій в ході контрольованого процесу;

генерування звітів про хід технологічного процесу;

забезпечення зв'язку із зовнішніми програмами обробки даних (СКБД, електронні таблиці, текстові процесори, тощо).

Крім того, програмне забезпечення (ПЗ) серверної частини (рис. 1) має містити модуль «журнал подій», призначений для ведення протоколів досліджень та детального

опису всіх подій і, зокрема, нештатних ситуацій, що відбуваються в системі.

Таким чином, технологічну мережу можна розглядати як класичну клієнт-серверну систему, функціональні особливості якої визначаються специфікою процесів, що обслуговуються [3]. Технологічний модуль у такій системі відіграє роль, як правило, клієнтської (активної) сторони взаємодії, тоді як серверна (пасивна) сторона призначена для обробки запитів від клієнтів.

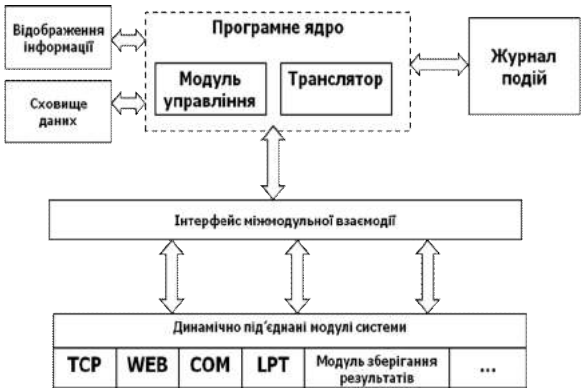

Рис. 1. Загальна структура серверного ПЗ

Для абстрагування користувачів ПЗ від особливостей технологічного процесу та принципів фізичної реалізації взаємодії між модулями ПЗ і ТО в даному програмному забезпеченні введено поняття «параметр». Параметр – це певна приведена характеристика p_i фізичної величини x_i деякого технологічного процесу, яка описується набором значень у певні моменти часу t_i. При цьому існує однозначна відповідність між множинами P↔X (P=$\{p_i\}$, X=$\{x_i\}$) в будь-який момент часу. Такий підхід забезпечує певну універсальність створюваного ПЗ і спрощує його застосування у разі зміни конфігурації технологічної мережі.

Програмне ядро (рис. 1) являє собою поєднання модуля управління системою та програмного транслятора. Модуль управління призначений для забезпечення базової функціональності керування системи. Програмний транслятор застосовуєтьмя для підтримки реалізації скриптів користувача на спеціалізованій і вбудованій в програмне ядро скриптовій мові.

Транслятор функціонально відповідає за взаємодію трьох компонентів:

1) IDE, призначеного для синтезу інтегрованого середовища розробки (Integrated Development Envirentment).

2) Engine як головного ядра транслятора;

3) Debugger як засобу налагоджування скрипта (рис. 2).

Сам транслятор побудований на основі компонента *JvInterpreter* із набору компонентів *JEDI*. Він реалізує простий інтерпретатор скрипта. Синтаксис скриптової мови є Pascal–подібним [2] з підтримкою засобів об'єктно-орієнтованого програмування (реалізовано підтримку класів, а також принципів поліморфізму та наслідування).

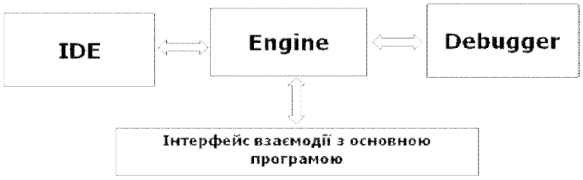

Рис. 2. Структура транслятора

Загалом алгоритм роботи інтерпретатора можна записати так:

1) інтерпретатор зчитує наступний у списку програми оператор;

2) даний оператор перевіряється на наявність помилок. У випадку наявності помилки, або невідомого оператора генерується виняткова ситуація (Exception) та припиняється виконання скрипта;

3) якщо оператор вірний – інтерпретатор намагається виконати його. Якщо ж в процесі виконання виникає логічна помилка (ділення на нуль, знаходження логарифма від'ємного числа тощо), то генерується виняткова ситуація і виконання скрипта переривається. В інших випадках зчитується наступний оператор;

4) якщо в скрипті немає більше операторів, то, зрозуміло, що інтерпретатор закінчує свою роботу.

Загальна структура транслятора (рис. 3) реалізована у вигляді набору окремих модулів.

JvInterpreterProgram – компонент з бібліотеки *JEDI*, який безпосередньо відповідає за роботу транслятора. Він реалізує лише виконання скрипта, проте не містить засобів для налагоджування коду, а також не має власного середовища розробки (текст скрипта передається як набір рядків типу *TStrings*).

FormDesigner – компонент системи, який забезпечує функціональність дизайнера форм (рис. 4). Подібно до дизайнера Delphi, забезпечує стандартні графічні компоненти (поля вводу, написи, кнопки), а також специфічні компоненти, які потрібні для управління та контролю технологічними процесами (різного роду аналогові вольтметри, лічильники, самописці, тощо).

Debugger – компонент, який забезпечує можливість налагоджування коду. Він забезпечує можливість встановлення точок зупину виконання скрипта (BreakPoint), які дають змогу відслідковувати стан змінних у процесі виконання, а також реалізує можливість покрокового виконання коду. Також даний компонент уможливлює отрима доступ до всіх змінних, які були описані в скрипті користувача під час його виконання.

CodeInside – компонент, призначений для реалізації випливаючих підказок, що випливають під час написання тексту скрипта користувачем (рис. 5). Дає змогу уникнути багатьох синтаксичних помилок, а також забезпечує прискорити процес написання коду.

VCL реалізовано у вигляді бібліотеки візуальних компонентів, розроблено на основі аналогічної бібліотеки середовища Delphi. В даному компоненті програмно реалізовано всі візуальні та невізуальні компоненти даної системи, які синтезовано на вбудованій скриптовій мові. Структурно *VCL* є складною ієрархією більш ніж 100 класів, в основі яких лежить класс *TObject*.

Використання Pascal-подібної скриптової мови дозволило без значних втрат використати бібліотеку VCL для середовища Delphi (дана бібліотека розповсюджується безплатно і з відкритим вихідним кодом). Користувач сам може створювати власні компоненти і додавати їх до цієї бібліотеки. Також вона містить функції для забезпечення доступу до самої системи, зокрема для реалізації параметрів, отримання їх значень, опису їх взаємодії тощо. Модулі динамічного підключення (plug-in – див. нижче) також можуть додавати свої власні функції, модулі, класи до даної бібліотеки. Наприклад, модуль підключення та роботи з COM-портом реалізує функції для обміну даними з послідовним портом вводу-виводу.

Основне завдання при розробці даного програмного забезпечення – реалізація його універсальності, забезпечення можливості максимального розширення областей впровадження даної системи.

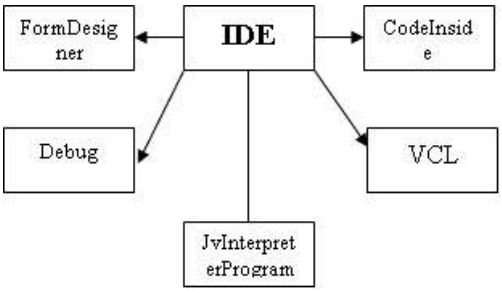

Рис. 3. Загальна структура транслятора

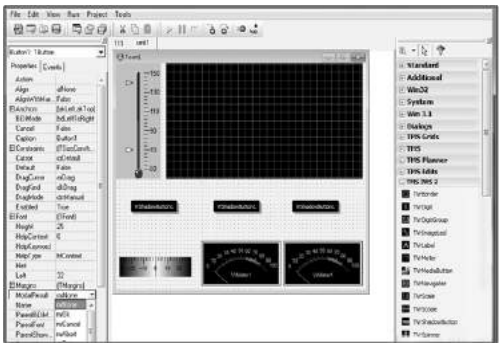

Рис. 4. Приклад використання дизайнера форм *FormDesigner*.

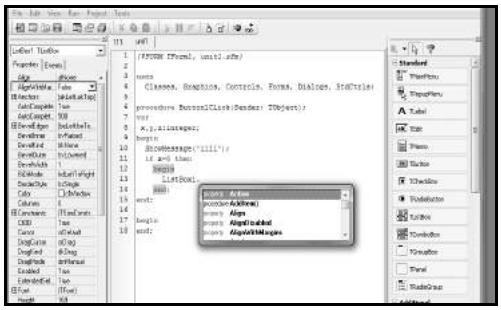

Рис. 5. Приклад використання *CodeInside* – компонента

Для забезпечення гнучкості системи впроваджено підтримку модулів динамічного підключення (*plug-in*). Особливістю даних модулів є можливість підключення їх функціональності в режимі реального часу без перекомпіляції системи в цілому. Кожний динамічний модуль – це бібліотека динамічного підключення DLL (*Dynamic Load Library*), що виконує заданий набір функцій. Оскільки програмне забезпечення побудоване на принципах об'єктно-орієнтованого

програмування [2], то обов'язковою умовою взаємодії динамічного модуля та серверного програмного забезпечення є реалізація певного набору інтерфейсів мережевого доступу до ТО (рис. 6).

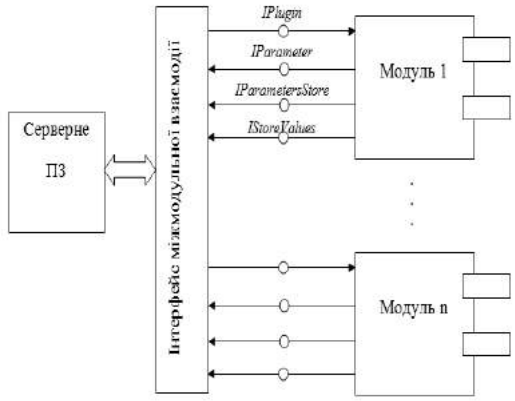

Рис. 6. Підключення динамічних модулів

Застосований модульний підхід до реалізації системи надає такі переваги:

гнучкість системи. При впроваджені системи потрібно розробити лише необхідний модуль, що реалізує певну унікальну особливість нового заданого технологічного процесу;

зменшення термінів впровадження завдяки повторному використанню вже розроблених модулів;

економія апаратних ресурсів (модулі, функціональність яких не потрібна в даний момент, не завантажуються в ОЗУ).

Уже розроблено необхідний набір динамічних модулів, які реалізують взаємодію з портами вводу–виводу (COM, LPT), обмін даними між сервером та клієнтами в локальній комп'ютерній мережі за протоколом TCP, модуль збереження результатів у файл, а також модуль, що забезпечує доступ до сервера шляхом використання WEB-технологій (HTTP-протокол).

Висновки. Описана в роботі структура серверного програмного забезпечення для контролю та управління технологічними процесами забезпечує високі показники гнучкості в процесі встановлення, надійності, а також універсальності завдяки використанню модульної структури, застосування вбудованої скриптової мови та можливості використання існуючих локальних мереж як засобів комунікації.

Список використаної літератури

1. Андреев Е.Б. SCADA–системы: взгляд изнутри / Е.Б.Андреев, Н.А.Куцевич, О.В.Синенко – М.: Изд-во «РТСофт», 2004. – 176 с.

2. Буч Г. Объектно–ориентированный анализ и проектирование с примерами приложений на С++ / Гради Буч. – СПб.: Бином, 2010. – 252 с.

3. Рогов Р.В. Сетевая система контроля технологического процесса выращивания полупроводниковых кристаллов и тонких пленок / Р.В. Рогов, С. В. Мельничук, Г.И. Воробец // Технология и конструирование в электронной аппаратуре. – 2005. – №. 5. – С.52–54.

Отримано 02.02.2012

References

1. Andreev E.B., Kutsevich N.A., Sinenko O.V. SCADA–systems: a sight from within – http: /www.booksgid.com/ information [in Russian].

2. Buch G. Object – focused analysis and designing with examples of appendices on C++ / Gradi Buch. – SPb.: Binom, 2010. – 252 p. [in Russian].

3. Rogov R.V., Melnichuk G.I., Vorobets S.V. The network system of monitoring of technological process of cultivation of semiconductor crystals and thin films / Technology and designing in electronic equipment. – 2005. – № 5. – P.52–54 [in Russian].

Рогов Роман Вікторович, ас. каф. комп. систем та мереж Чернівецького нац. ун-ту ім. Ю. Федьковича,
rman_rogov@mail.ru ,тел. (0372)-52-64-46.

Воробець Георгій Іванович, канд. фіз,-мат. наук, доц. каф. комп. систем та мереж Чернівецького нац. ун-ту ім. Ю. Федьковича,vgeorge@chnu.cv.ua
тел. (0372)-52-64-46.

Воробець Олександр Іванович, канд. фіз,-мат. наук, доц. каф. комп. систем та мереж Чернівецького нац. ун-ту ім. Ю. Федьковича, o.vorobets@chnu.edu.ua
тел. (0372)-52-64-46.