4 ПРОГРАМУВАННЯ ЗАСОБІВ КОМП‘ЮТЕРНО-ІНТЕГРОВАНИХ СИСТЕМ УПРАВЛІННЯ

4.1 Програмні засоби нижнього рівня КІСУ

4.1.1 Види програмного забезпечення

Програмне забезпечення розділяється на системне і прикладне. Системне програмне забезпечення складають операційні системи контролерів і автоматизованих робочих місць оператора.

Операційна система (ОС) – організована сукупність програмних засобів, призначена для керування ресурсами обчислювальної системи. Операційні системи діють як інтерфейс між апаратурою і користувачами і займають найважливіше місце в сукупності сучасних системних програмних засобів. Вони є основою організації обчислювального процесу і визначають ефективність як використання апаратних компонентів системи, так і вирішення поставлених задач.

Прикладне програмне забезпечення (ППЗ) підрозділяється на:

• ППЗ контролерів: непроцедурні технологічні мови, що дозволяють легко реалізовувати логічні операції; конфігуратор і бібліотека програмних модулів (модулі математичних функцій, первинної обробки інформації, регулювання). Особливостями ППЗ контролерів є: простота використання технологічних мов; наявність у бібліотеці модулів сучасних досконалих алгоритмів (алгоритми самонастроювання регуляторів, адаптивного керування, нечіткого регулятора й ін.). Деякі контролери можуть виконувати програми, написані мовамих високого рівня;

• ППЗ пультів операторів.

В ППЗ включаються пакети прикладних програм як загального призначення (статистична обробка інформації, експертна система підтримки прийняття управлінських рішень і т.п.), так і об'єктного (раціональне, а іноді оптимальне керування типовими процесами).

Розробка прикладного програмного забезпечення пультів оператора може здійснюватися двома шляхами: з використанням традиційних мов програмування (C++, Java, Python та ін.) або з використанням існуючих готових інструментальних проблемно-орієнтованих засобів.

Процес створення ППЗ з нуля з використанням традиційних мов програмування для КІСУ є неприпустимо тривалим і складним.

4.1.2 Операційні системи реального часу

На теперішній час найбільш поширеними ОС є Microsoft Windows і UNIX. Під їх керуванням працюють більшість серверів і робочих станцій. Але для використання в керуючих обчислювальних пристроях нижніх рівнів на перший план виходять вимоги можливості роботи в режимі реального часу.

Системою реального часу називається будь-яка система, в якій суттєву роль грає час генерації вихідного сигналу. Часова затримка від одержання вхідного сигналу до видачі вихідного повинна бути малою, щоб забезпечити прийнятний час реакції на незаплановані події.

Для побудови систем реального часу використовуються мови програмування реального часу (МП РЧ) та операційні системи реального часу (ОС РЧ).

ОС РЧ (англ. Real-Time Operating System, RTOS) – це система, яка обслуговує зовнішні процеси, які розвиваються у пристроях, що мають суворі обмеження на час відповіді. Діями ОС РЧ керують переривання від зовнішніх процесів; якщо вони не будуть швидко оброблені (у залежності від процесу протягом мікро-, мілісекунд чи секунд), то хід зовнішнього процесу може спотворитися. Ці системи часто проектуються для окремого застосування, наприклад, для керування конкретним технологічним процесом.

Проте поняття «система реального часу» не треба ототожнювати з поняттям «швидка система». Це не завжди правильно. Час затримки реакції СРЧ на подію буває не так і важливий (він може досягати декількох секунд). Головне, щоб цього часу було досить для конкретної ситуації і гарантовано.

Операційна система, яка може забезпечити необхідний час виконання завдання реального часу навіть в найгірших випадках, називається операційною системою жорсткого реального часу. Система, яка може забезпечити необхідний час виконання завдання реального часу в середньому, називається операційною системою м'якого реального часу.

Порівняння ОСРЧ і звичайних операційних систем показане у табл.4.1.

Таблиця 4.1

	ОС реального часу	**ОС загального призначення**
Основна задача	Встигнути зреагувати на події, що відбуваються на устаткуванні	Оптимально розподілити ресурси комп'ютера між користувачами та задачами
На що орієнтована	Обробка зовнішніх подій	Обробка дій користувача
Як позиціонується	Інструмент для створення конкретного апаратно-програмного комплексу реального часу	Сприймається користувачем як набір застосунків, готових до використання
Кому призначена	Кваліфікований розробник	Користувач середньої кваліфікації

Загальні вимоги, що пред’являються до систем реального часу:

а) своєчасна реакція – після того як відбулася подія, реакція має бути не пізніше, ніж через необхідний час, перевищення цього часу розглядається як серйозна помилка;

б) одночасна обробка інформації, яка характеризує зміну процесу декількох подій. Навіть якщо одночасно відбувається кілька подій, реакція на жодне з них не повинно запізнюватися. Це означає, що система реального часу повинна мати вбудований паралелізм. Паралелізм досягається використанням декількох процесорів в системі і/або багатозадачним підходом.

Основу втілення системи реального часу складає ОС РЧ чи спеціалізовані керуючі програми, об'єднані в спеціалізовану ОС. Використання універсальних ОС РЧ у повному обсязі для випадку різних конфігурацій систем керування як на основі багатопроцесорної системи, так і у формі автоматизованого робочого місця з контролерами є недоцільним. Причиною є ієрархічна надмірність універсальних ОС, яка забезпечує можливість підключення засобів налагодження програм і інших засобів полегшення праці програміста. Тому використовуються спеціалізовані ОС РЧ, розраховані на вирішення вузького кола задач.

Кожна спеціалізована ОС проектується з урахуванням доступних ресурсів і повного обсягу пам'яті, необхідного для виконання оперативних задач.

Особливі вимоги до ОС РВ *керуючих систем*:

а) оперативна взаємодія (обмін) із блоком керування на базі зворотного зв'язку за результатами керування;

б) підтримка стратегії керування з мінімально можливим простоєм основного устаткування системи керування;

в) можливість реконфігурації структури і перерозподіл задач між процесорами.

У режимі оперативної роботи повинні розв’язуватись наступні спеціальні задачі:

а) видача керуючих сигналів на контролери чутливих елементів і виконавчих пристроїв;

б) приймання запитів на обслуговування віддалених користувачів.

в) взаємодія з іншими комп'ютерами мережі, що забезпечують технологічну підготовку й інформування про стан керованого технологічного процесу.

г) тестування стану обчислювального комплексу і пристроїв системи керування;

д) реініціалізація (тобто перезапуск) ОС при виявленні особливих ситуацій.

На сьогодні найбільш широко розповсюдженими ОС РЧ є QNX Neutrino, RTOS, RTEMS, ChorusOS, RTX для Windows NT, INtime, TinyOS, OSEK / VDX, Contiki, pSOS, INTEGRITY, LynxOS, Microware OS-9, GRACE-OS, C EXECUTIVE, CMX-RTX, Inferno.

4.1.2 Програмування ПЛК

Однією з важливих особливостей сучасного програмного забезпечення ПЛК є те, що можна використовувати різні технологічні мови програмування. Стандартом МЕК1131-3 визнані 5 мов програмування:

а) мова крокових діаграм LD;

б) мова функціональних блокових діаграм FBD;

в) мова послідовних функціональних схем SFC;

г) мова структурованого тексту ST;

д) мова інструкцій IL.

Кожна з них має свої особливості і пристосована для розв'язання тих чи інших задач. Наприклад, мова крокових діаграм має переваги при реалізації алгоритму керування, який раніше був реалізований на релейно-контактних схемах. Мова функціональних блоків найбільш зрозуміла для фахівців, які мають великий досвід створення систем з окремих функціональних блоків – таймерів, лічильників, блоків арифметичних операцій і т.д. Якщо у програмі користувача необхідно виконувати велику кількість операцій обробки інформації, краще використовувати мову структурованого тексту.

Раніше основним технічним засобом для програмування ПЛК були спеціалізовані пульти програмування, Вони використовувались як у вигляді вбудованих в ПЛК блоків, так і у вигляді окремих технічних засобів, які приєднувались до ПЛК в процесі програмування. Ці пульти використовувались також при відлагоджуванні програм, виконанні процедур діагностики і тестування ПЛК.

Сучасні термінали програмування на базі персональних комп'ютерів мають суттєві переваги над пультами програмування. Для програмування контролерів використовується спеціальне програмне забезпечення, основним завданням якого є створення максимальних зручностей для програмування і відлагодження програм. Процедури програмування замінюються на більш прості процедури конфігурування, що дає можливість користувачу, не входячи в тонкощі програмування, забезпечувати виконувати досить складних процедур.

Програмні термінали дають можливість використовувати для різних за алгоритмами задач різні технологічні мови. А вже при записі програми у ПЛК різні фрагменти програми поєднуються в один машинний код.

4.2 Програмні засоби верхнього рівня КІСУ

4.2.1 Концепція інтеграції програмного забезпечення

Інтеграція має на увазі не єдину глобальну систему як таку, а перш за все взаємодія всіх рівнів програмного забезпечення між собою. Різні програмні системи, створені за допомогою різних засобів, встановлених на різних платформах, які працюють на різних комп'ютерах, вміють "домовлятися". Тобто вони знають, як запитати один в одного дані і як послати один одному "вказівки".

У компанії Microsoft є глобальна інтеграційна тенденція, що реалізувалася у створенні COM-технології.

COM – Component Object Model (модель складових об'єктів) – і її мережеве розширення DCOM – Distributed COM (розподілена COM) – це технологія, введена спочатку Microsoft для інтеграції різних офісних додатків. Модель COM оперує об'єктами, дуже схожими на об'єкти в об'єктно-орієнтованих мовах програмування типу C++. Але сама технологія COM не є мовою програмування. Вона тільки регламентує поведінку своїх об'єктів. Об'єкт після створення надає свою функціональність процесу, який його викликав, а після використання – знищується.

Об'єкти COM передають свою функціональність через інтерфейси. Інтерфейс в COM (не плутати з тим, що зазвичай розуміється під словом інтерфейс) об'єднує групу взаємопов'язаних функцій, що надаються об'єктом. Саме інтерфейс, вірніше, покажчик на нього є тим, з чим працює процес, що викликає даний об'єкт.

Об'єкт COM – сторона пасивна. Він лише передає через інтерфейси свої функції. Для програми, що його реалізує, вживається термін COM-сервер. Запитуюча програма, відповідно, називається COM-клієнт. Але це не виключає того, що обидві програми одночасно можуть бути і COM-серверами, і COM-клієнтами.

Щоб створити об'єкт, потрібно знати, де він знаходиться. У Windows для цього використовується реєстрація об'єктів в системному реєстрі. У реєстрі реєструються також інтерфейси. При цьому кожен COM-предмет реєстрації має унікальний ідентифікатор, званий GUID (Globally Unique Identifier – глобальний унікальний ідентифікатор). Реєстрація робить доступною інформацію про розташування об'єктів всіх програм, що є на комп’ютері.

Для мережевих рішень існує DCOM – розширення COM, що дозволяє добиратися до об'єктів на інших комп'ютерах. Важливим є те, що з точки зору програмування нічого не змінюється: DCOM – це системний сервіс, який робить COM прозорим в локальних мережах.

4.2.2 Технологія OPC

У системах керування технологічними процесами сьогодні широко застосовуються численні програмні рішення (наприклад, SCADA) різних виробників, причому робота цих програмних систем базується на постійному обміні даними з компонентами системи автоматизації (контролерами, модулями ПЗО і т.д.). Можливість такої взаємодії забезпечується виробниками цих програмних рішень шляхом самостійної розробки ними драйверів, що інтегруються у вищезгадані програмні пакети. Такий підхід, як правило, веде до наступних проблем:

• збільшення витрат: повинні розроблятися окремі драйвери для кожного підтримуваного пристрою (драйвер – програма операційної системи, що обслуговує окремі периферійні пристрої обчислювальної системи; драйвер повинний враховувати всі особливості конструкції пристрою і роботи його в реальному часі);

• обмежена функціональність драйверів: розроблювачем драйверів часто підтримуються не усі функції відповідного пристрою;

• обмежені можливості розширення і зміни складу компонентів системи автоматизації: унаслідок модернізації апаратної платформи драйвер або взагалі не може більше використовуватися, або може працювати нестабільно;

• конфлікти доступу: різні програми не можуть одночасно здійснювати доступ до тих самих компонентів системи автоматизації, тому що звертання до даних здійснюється через власні драйвери, робота одного з яких у кожен момент часу блокує можливість роботи всіх інших.

Вирішити ці проблеми можуть виробники апаратних компонентів КІСУ, розробивши власні драйвери, оснастивши їх спеціальними стандартизованими інтерфейсами, щоб програми різних виробників програмного забезпечення могли без проблем їх використовувати.

OPC – це набір повсюдно прийнятих специфікацій, що надають універсальний механізм обміну даними в системах контролю і управління. Абревіатура OPC традиційно розшифровується як OLE for Process Control. OLE (англ. Object Linking and Embedding, вимовляється як oh-lay [олей]) – технологія зв'язування та впровадження об'єктів в інші документи та об'єкти, зокрема, таблицю, створену в Word можна передати в Excel і навпаки

Суть OPC проста: надати розробникам промислових програм універсальний фіксований інтерфейс (тобто набір функцій) обміну даними з будь-якими пристроями. При цьому використовується технологія клієнт-сервер.

Кожне підприємство, яке виробляє будь-який пристрій, наприклад, датчик, розробляє для нього OPC-сервер – програму, яка отримує дані у внутрішньому форматі пристрою або системи і перетворює ці дані в формат OPC. За своєю суттю OPC-сервер – це такий собі універсальний драйвер фізичного обладнання, що забезпечує взаємодію з будь-яким OPC-клієнтом.

Розробники програмного забезпечення більш високого рівня, наприклад, SCADA-пакетів включають в нього OPC-клієнт – програму, яка приймає від OPC-серверів дані в форматі OPC. Таким чином, OPC-сервер створює свого роду абстракцію апаратури, дозволяючи будь-якому OPC-клієнту записувати і зчитувати дані з пристрою.

Через інтерфейси OPC одні програмні системи можуть читати чи записувати дані в інші програмні системи, обмінюватися подіями, оповіщати одна одну про позаштатні ситуації (тривоги), здійснювати доступ до даних, зареєстрованих в архівах («історичні» дані). Ці програмні системи можуть розташовуватися як на одному комп'ютері, так і бути розподіленими по мережі.

Існує три основних способи отримання OPC-клієнтом даних від OPC-сервера: синхронне читання, асинхронне читання і підписка. При синхронному читанні клієнт посилає серверу запит зі списком цікавих йому змінних і чекає, коли сервер його виконає. При асинхронному читанні клієнт посилає серверу запит, а сам продовжує працювати. Коли сервер виконав запит, клієнт отримує відповідь. І, нарешті, в разі передплати клієнт передає серверу список параметрів, які його цікавлять, а сервер потім регулярно надсилає клієнту інформацію про змінені параметри. Ці списки в термінології OPC називаються групами. Кожен клієнт може підтримувати одночасно багато груп з різною швидкістю оновлення.

Тепер у розроблювачів програмного забезпечення відпадає необхідність написання нових драйверів, якщо внаслідок модернізації деякого апаратного компонента змінюється набір функцій доступу до її даних. Користувачі одержують велику свободу вибору при конфігуруванні і підборі апаратних засобів вирішення задач автоматизації.

4.2.3 Застосування OPC у КІСУ

На рис. 4.1 представлена схема, що ілюструє можливі області застосування OPC-серверів в КІСУ.

Розрізняються кілька рівнів управління:

а) нижній рівень – польові шини (fieldbus) і окремі контролери;

б) середній рівень – цехові мережі;

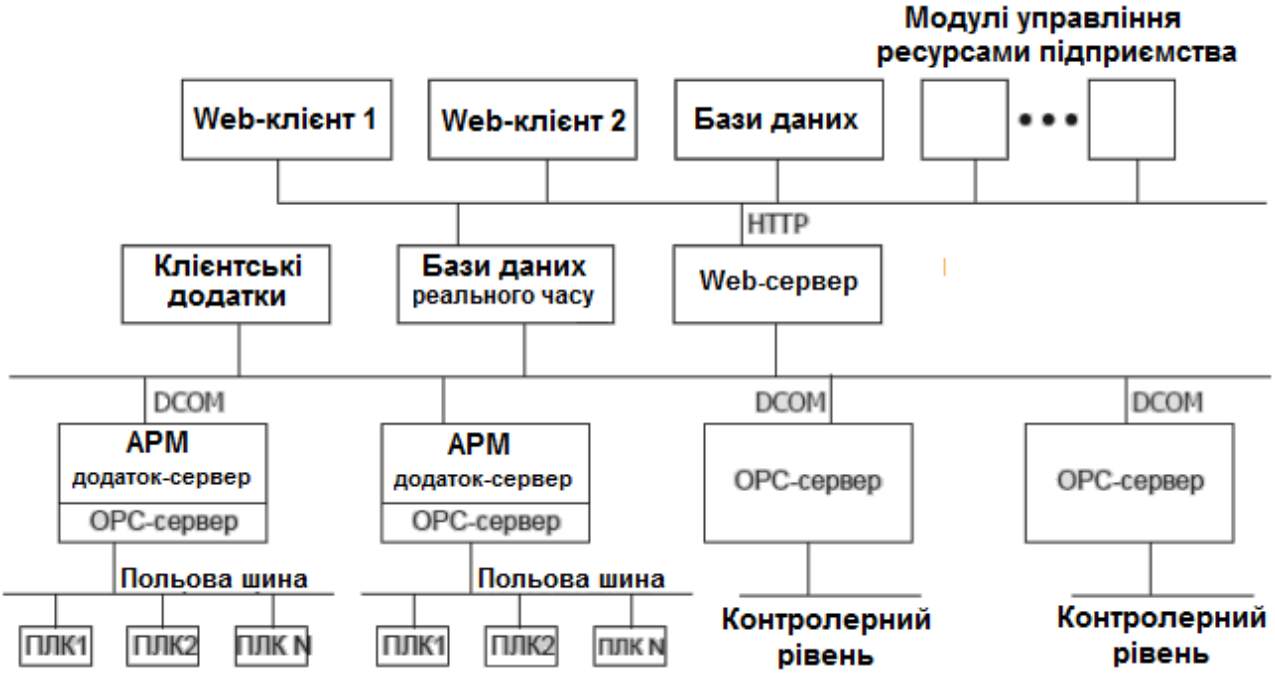

Рис. 4.1 – Можливі області застосування OPC-серверів у КІСУ

в) рівень АСК ТП – рівень роботи систем типу SCADA;

г) рівень АСУП – рівень додатків управління ресурсами підприємства.

Кожен з цих рівнів може обслуговуватися OPC-сервером, поставляючи дані OPC-клієнту на більш високому рівні або навіть "сусідові".

Розглянемо варіанти використання OPC-сервера.

Якщо є обладнання, наприклад, плата АЦП, керована за допомогою драйвера на комп'ютері з Windows або іншої ОС, що підтримує COM / DCOM, то безпосередньо поверх драйвера реалізується OPC-сервер. Заміна пристрою не потребує зміни інших додатків: драйвер змінюється, але OPC-інтерфейс поверх нього залишається колишнім.

При наявності пристрою під керуванням через який-небудь мережевий протокол, можлива реалізація OPC-сервера, який отримує дані з цього протоколу.

Ще один різновид OPC-сервера – шлюз до мережі типу польової шини, такої, як Profibus. На комп'ютері з ОС Windows буде встановлюється адаптер fieldbus-мережі, а OPC-сервер буде взаємодіяти з цією мережею через драйвер адаптера. Мережа польової шини працює в жорсткому режимі реального часу, а OPC надає менш вимогливий шлюз до цієї мережі з додатків більш високого рівня.