КОНФІГУРАТОР

СИСТЕМА АВТОМАТИЧНОГО КОНФІГУРУВАННЯ ТЕЛЕМЕХАНІЧНИХ ЗАСОБІВ

ЗМІСТ

АНОТАЦІЯ

У даній роботі вперше розроблено систему автоматичного конфігурування або конфігуратор телемеханічних засобів. Конфігуратор призначений для створення та налагодження системи телемеханіки. Конфігуратор може застосовуватися на підприємствах, що спеціалізуються на розробці та впровадженні інформаційно-управляючих телемеханічних комплексів (ІУТК) для автоматизованих систем управління об’єктами різних галузей промисловості й непромислової сфери. Розроблені структурна схема конфігуратора, блок-схема алгоритму автоматичної побудови моделі багаторівневого ІУТК, програма. Конфігуратор телемеханічних засобів реалізовано на мові програмування C#, також були використані новітні архітектурні тенденції. Була проведена експериментальна перевірка розробленого конфігуратора, яка показала, що він виконує всі поставлені задачі та відповідає всім вимогам, що висуваються до подібних засобів. Конфігуратор дозволяє здійснювати налаштування телемеханічної системи швидко, зручно, з мінімальним залученням людських ресурсів. У теперішній час конфігуратор впроваджується на ПП «Науково-виробниче підприємство «Промекс» шляхом інтегрування в базове програмне забезпечення ІУТК.

ВСТУП

Системи телемеханіки або інформаційно-управляючі телемеханічні комплекси (ІУТК) мають широкий діапазон галузей застосування: енергооб`єкти, шахти, трамвайно-тролейбусні управління, зовнішнє освітлення міст, залізничний транспорт, метрополітени, великі промислові об`єкти, нафтопромисли, аеропорти, водоканали. Такі системи потрібно розробити та налагодити.

Існує цілий клас задач з налаштування, вирішувати які навіть фахівець високого рівня хотів би у пакетному режимі, шаблонно. Це – завдання початкового налаштування, які виникають при встановленні системи.

По-перше, під час встановлення виникає потреба налаштувати одразу всі параметри системи, або, принаймні, встановити їх в якийсь стан за замовчуванням. Кількість таких налаштувань обчислюється тисячами, і завжди існує ймовірність того, що людина припуститься помилки. Застосування шаблонів при налаштуванні зменшує ймовірність появи помилки, і, якщо навіть вона з’явиться, то буде мати мінімальний вплив, і її можна буде виправити потім.

По-друге, на повне прописування всіх параметрів сучасної системи вручну сьогодні, навіть із використанням відповідних довідників, таке налаштування займе надто багато часу.

По-третє, більшість (практика показує, що близько 80%) налаштувань за замовчуванням, якщо вони проставлені автоматично або на основі короткого діалогу, згодом змінювати не доведеться взагалі. Це налаштування мови, апаратури, мережі і т. д.

У роботі вперше розроблятиметься конфігуратор, призначений для роботи із телемеханічними засобами згідно із протоколами МЭК 870–5–101, МЭК 870–5–104, Modbus RTU, Modbus TCP, Гранит, Гранит-микро, Гранит Ethernet TCP/IP.

1 ОГЛЯД ТА АНАЛІЗ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1.1 Постановка задачі

Система автоматичного конфігурування або конфігуратор телемеханічних засобів – це програма (або додаток), яка виконує функцію інтерфейсу взаємодії із користувачем та дозволяє створити та налагодити систему телемеханіки швидко, зручно, без обов`язкового залучення персоналу з високим рівнем кваліфікації. Конфігуратор є реалізацією принципу використання шаблонів: якщо існує клас типових задач, то чому б їх не розв`язувати автоматично, з мінімальним залученням людських ресурсів, але разом із цим, конфігуратор не повинен бути тільки набором готових шаблонів.

Сучасні інформаційно-управляючі телемеханічні комплекси будуються за принципами SCADA-систем (Supervisory Control And Data Acquisition) – диспетчерських систем із супервізорним управлінням при зборі даних.

Відомо, що SCADA-системою називається людино-машинний інтерфейс, призначений для взаємодії людини (оператора, диспетчера) з автоматизованим технологічним процесом [5].

Сучасні SCADA-системи виконують безліч функцій, а саме: налаштування на конкретну задачу; диспетчерське управління; автоматичне управління; зберігання історії процесів; виконання функцій безпеки; виконання загальносистемних функцій.

Але основною ознакою SCADA-системи є наявність інтерфейсу з користувачем. Без цього інтерфейсу управління стає автоматичним, на відміну від диспетчерського.

Як система диспетчерського управління, SCADA повинна забезпечувати виконання таких завдань: взаємодію з оператором (видачу візуальної і звукової інформації, передачу до системи команд оператора); допомогу оператору в прийнятті рішень; автоматичну сигналізацію про аварії і критичні ситуації; видачу інформаційних повідомлень на пульт оператора; ведення журналу подій у системі; витягнення інформації з архіву і пред'явлення її оператору в зручному

для сприйняття вигляді; підготовку звітів; облік напрацювання технологічного обладнання.

Загальне диспетчерське (супервізорне) управління в ІУТК здійснюється повністю центральною приймально-передавальною станцією (ЦППС). Апаратні та програмні засоби ЦППС утворюють автоматизований оперативно–інформаційний комплекс (АОІК) зі вставками для реалізації автоматизованих робочих місць диспетчера (АРМ-Д) та обслуговуючого персоналу (АРМ-ОП).

Оброблювальний центр ЦППС реалізується на одній або декількох ПЕОМ. Бажана структура (архітектура) ЦППС із незалежною роботою всіх ПЕОМ ОЦ і автоматичним створенням синхронних баз поточних і ретроспективних даних. Такі ОЦ підвищують «життєздатність» ІУТК (здатність виконувати задані функції в повному обсязі при непрацездатності складових частин) завдяки практично повному виключенню інтервалів часу, протягом яких база даних веденої ПЕОМ не відповідає реальній і накопиченій до моменту виходу з ладу ведучої ПЕОМ[10].

1.2 Вимоги до програмного забезпечення сучасного ІУТК

До програмного забезпечення (ПЗ) сучасних інформаційно-керуючих телемеханічних комплексів висуваються такі основні вимоги:

1) використання для побудови ІУТК загальноприйнятих стандартних операційних систем, драйверів введення-виведення інформації, структур баз даних;
2) «відкритість» для користувача;
3) можливість створення на основі апаратних і програмних засобів SCADA-системи, яка реалізує функції автоматизованого оперативно–інформаційного управляючого комплексу (АОІУК);
4) включення до складу базового пакету програм інструментальних програм і графічних редакторів із мстою спрощення адаптації ІУТК до конкретних умов застосування;

5) наявність пакету програм для автоматизації документообігу диспетчера.

Зазвичай, базове ПЗ ІУТК забезпечує [5]:

- обмін інформацією між ЦППС і пристроями контрольованих пунктів (RTU (КП)) у відповідності з існуючими алгоритмами роботи пристроїв;
- обробку інформації, відображення її на екранах моніторів ПЕОМ, приладах диспетчерських щита та пульта, реєстрацію друкувальним пристроєм;
- «прив'язування» інформації пристрою контрольованого пункту до системного часу ПЕОМ АОІУК (астрономічного часу);
- створення ієрархічних структур.

Склад базового програмного забезпечення ІУТК показано на рисунку 1.1.

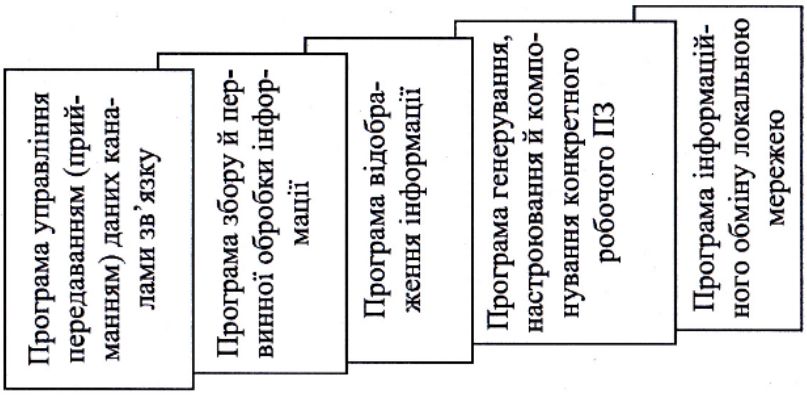


Рисунок 1.1 – Склад базового програмного забезпечення

Із рисунку видно, що до складу базового ПЗ ІУТК входять такі основні програми:

- програма управління передаванням (прийманням) даних каналами зв'язку;
- програма збору й первинної обробки інформації;
- програма відображення різноманітної інформації;
- програма генерування, налаштування й компонування конкретного робочого ПЗ зі стандартних програмних модулів;
- програма інформаційного обміну локальною мережею.

За допомогою базового програмного забезпечення ІУТК створюються ба-

зи поточних і попередніх даних. Від системи управління базами даних потребується:

- будувати графіки величин (станів) контрольованих і вимірюваних параметрів;
- фіксувати вибігання параметрів за встановлені межі;
- реєструвати нештатні ситуації за заданими критеріями;
- формувати таблиці попередніх даних за часом, подіями, впорядкованих за видами інформації, адресами об’єктів і т.д.;
- формувати зведення даних за встановленими формами;
- фіксувати дії диспетчера з прив’язуванням подій до поточного часу.

Інструментальні програми дозволяють створювати технологічні кадри – мнемосхеми всього або частин об’єкта й довільно виділяти на мнемосхемах місця відображення дискретних сигналів (стану обладнання), значень вимірювання або обчислювання параметрів. Такими програмами встановлюється відповідність між системними й технологічними адресами й іменами об’єктів. Програми дозволяють легко змінювати види мнемосхем (технологічних кадрів).

Інструментальні програми визначають адреси об’єктів, стани або значення яких виводяться на диспетчерський щит (електронне табло), встановлюють види відображуваної інформації й дозволяють коректувати раніше задані.

1.3 Інтерфейси і протоколи в ІУТК

Сучасні інформаційно-управляючі телемеханічні комплекси оснащені портами послідовного зв’язку інтерфейсу RS-232. Не зважаючи на широке розповсюдження, інтерфейсу RS-232 властиві обмеження, які стосуються швидкості обміну даними, довжини лінії зв’язку і можливостей створення мереж передачі даних. Тому в ІУТК використовуються й інші послідовні інтерфейси, що дозволяють подолати вказані обмеження, а також здійснювати міжмодульні обміни інформацією, а саме RS-485, «струмова петля», SPI. При цьому передача інформації здійснюється згідно з протоколами Modbus, HDLC, МЭК 60870–5–101(104), а також протоколами TCP/IP, CIP, UDP/IP.

Висновки до розділу 1

Сучасні інформаційно-управляючі телемеханічні комплекси (ІУТК) будуються за принципами SCADA-систем – диспетчерських систем із супервізорним управлінням при зборі даних.

Виходячи із понять Конфігуратора та SCADA-системи, випливає потреба у безпосередньому застосуванні «Програми Конфігуратора», але така програма до останнього часу не була розроблена. Це і є підставою для його створення та підсумовуючи все вищевикладене, функціональний і практичний Конфігуратор повинен відповідати таким основним вимогам:

1) опис об'єктів для протоколів «МЭК 870–5–101», «МЭК 870–5–104», «Modbus RTU», «Modbus TCP», «Гранит», «Гранит-микро», «Гранит Ethernet TCP/IP»;
2) інтерфейс користувача повинен бути наочним, ергономічним та дозволяти легко розібратись у структурі телемеханічної системи та швидко налаштувати необхідні елементи – додавати датчики, розширити або навпаки – видалити частину пристроїв;
3) кожен самодостатній об'єкт повинен мати унікальний ідентифікаційний номер;
4) структура повинна відображуватись стисло та розгорнуто;
5) структура проекта має редагуватися з клавіатури та контекстного меню;
6) розгорнутий опис структури RTU(КП) повинен мати графічний супровід;
7) збереження / завантаження у форматі xml, яка відповідає структурі проекта і дозволяє провести редагування файла без використання самого конфігуратора телемеханічних засобів;
8) повинен мати у своєму складі модуль автоматичної побудови моделі багаторівневого ІУТК за протоколами «Гранит», «Гранит-микро»;
9) повинна бути передбаченою локалізація на трьох мовах – українській, російській та англійській.

2 РОЗРОБКА СИСТЕМИ АВТОМАТИЧНОГО КОНФІГУРУВАННЯ ТЕЛЕМЕХАНІЧНИХ ЗАСОБІВ

2.1 Загальна структура сучасного ІУТК

Структура сучасної системи телемеханіки або однорівневого інформаційно-управляючого телемеханічного комплексу (ІУТК) містить такі основні складові частини (рисунок 2.1): ЦППС – центральна приймально-передавальна станція (пристрій пункту управління); RTU – remote terminal unit (пристрій контрольованого пункту); ЩД – диспетчерський щит (або інтерактивна відеостіна); ПД – пульт диспетчера; ДПМКС – датчики повідомлюваних, метрологічних, кодових сигналів; ВМ – виконавчі механізми; ПЕОМ – персональна ЕОМ; пристрої можуть сполучатися РЛЗ – радіальними лініями зв'язку; МЛЗ – магістральними лініями зв'язку; ТЛЗ – транзитними (або ланцюжковими) лініями зв'язку.

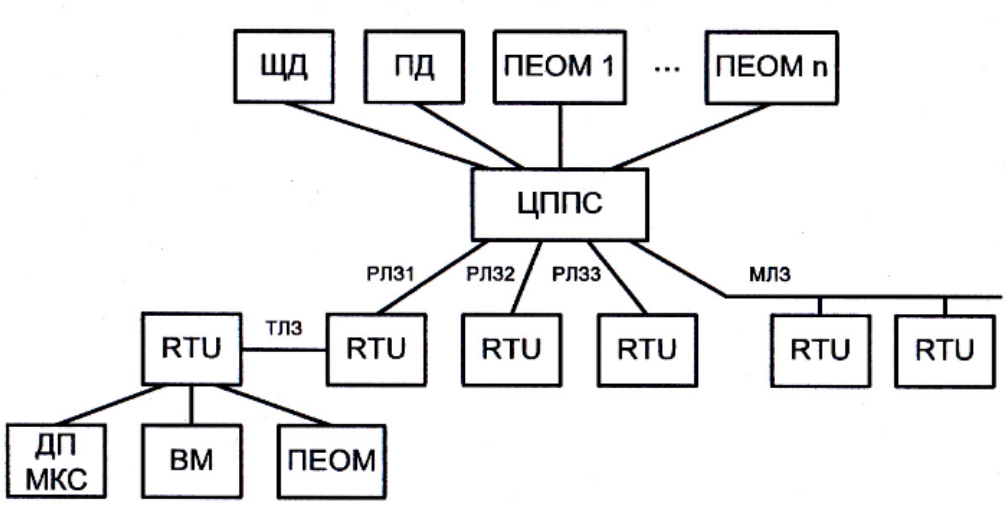


Рисунок 2.1 – Структура однорівневого ІУТК

Однорівневий ІУТК містить такі основні частини: одну центральну приймально-передавальну станцію або пристрій пункту управління (ПУ) та деяку кількість RTU або пристроїв контрольованого пункту (КП). Множина пристроїв КП будується відповідно до умов застосування ІУТК.

ЦППС і RTU можуть з`єднуватися лініями зв'язку (ЛЗ) радіальної, магістральної, транзитної (або ланцюжкової) структури. Типи та кількість ЛЗ визначається відповідно до умов застосування ІУТК.

Багаторівневі ІУТК (на відміну від декількох однорівневих) містять дві або більше ЦППС та забезпечують інформаційні обміни між ними. Якщо ранг пристроїв ЦППС багаторівневих ІУТК різний, то структура ІУТК є ієрархічною.

Найбільш важливою та відповідальною частиною інформаційно-управляючого телемеханічного комплексу є центральна приймально-передавальна станція.

Центральна приймально-передавальної станція складається із щита (електронних табло, екранів) диспетчера, пульта диспетчера з одної або декількох ПЕОМ, на базі яких створюється оброблювальний центр (ОЦ) ЦППС.

Структура ЦППС представлена на рисунку 2.2.

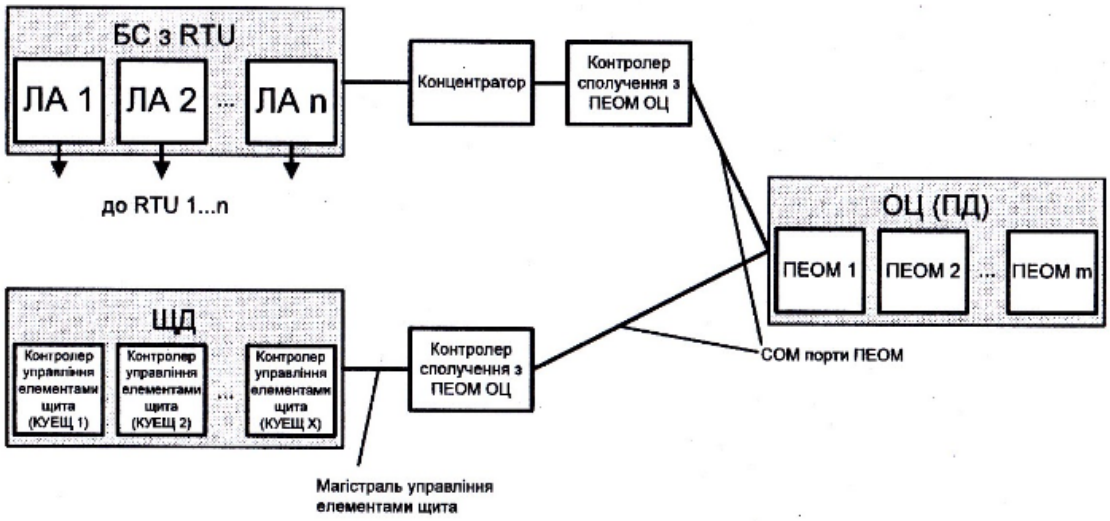


Рисунок 2.2 Структура ЦППС

На рисунку прийняті такі позначення (окрім тих, що введені раніше): БС – блок сполучення; ЛА – лінійний адаптер.

Розглянемо більш детально принципи побудови та функціонування ЦППС.

Для сполучення ЦППС із RTU застосовується відповідний блок БС. Блок сполучення утворюють лінійні адаптери ЛА 1, ЛА 2…, ЛА n. Лінійні адаптери, в першу чергу, являють собою модеми, призначені для здійснення інформаційних обмінів усіма використовуваними в ІУТК лініями зв’язку. Вигляд ЛА визначається типом використовуваної для сполучення з RTU ЛЗ, а їх кількість – кількістю ліній приймання-передавання інформації, що відходять від ЦППС.

Якщо всі контрольовані пункти (RTU) сполучаються з ЦППС радіальними лініями зв'язку, то кількість ЛА дорівнює кількості RTU; при використанні магістральних і транзитних ліній зв'язку кількість ЛА є меншою за кількість RTU.

Контролер внутрішньої магістралі являє собою концентратор супервізорного управління блоком ЛА. Він регулює обмін даними між RTU та оброблювальним центром ЦППС.

Дані концентратора надходять до ПЕОМ ОЦ через відповідний контролер сполучення. Зазвичай, для сполучення апаратури ЦППС із ПЕОМ використовуються COM порти, які, як відомо, реалізують інтерфейс RS-232. Таким чином, функція контролера сполучення полягає в перетворенні протоколу, використовуваного при зборі даних, у протокол COM порту.

Оброблювальний центр ЦППС реалізується на базі однієї або декількох ПЕОМ. Доцільно застосовувати архітектуру ЦППС із незалежним функціонуванням ПЕОМ і автоматичним створенням синхронних баз поточних і попередніх даних. Основні переваги пропонованої архітектури ОЦ такі:

1) підвищена «життєздатність», оскільки виключаються інтервали часу, протягом яких база даних у ОЦ не відповідає реальній, накопченій до моменту виходу з ладу основної ПЕОМ, базі даних;
2) розширення функціональних можливостей для диспетчера, який може користуватися «технологічними кадрами», що відображаються на екранах моніторів декількох ПЕОМ.

Контролер управління елементами щита являє собою двонаправлений ретранслятор даних, що надходять від (до) ПЕОМ оброблювального центра на щит диспетчера для відображення їх за допомогою індикаторів, наприклад, одно – або двокольорових світлодіодів. Індикатори під'єднуються безпосередньо до виходів контролерів панелей щита.

Контролери панелей щита розрізняють двох типів: контролер панелі «світлого» або «напівсвітлого» щита та контролер панелі «темного» щита. Контролер панелі «світлого» або «напівсвітлого» щита забезпечує відображення інформації (телесигналізації) за методом «світлого» або «напівсвітлого» щита. Кон-

тролер панелі «темного» щита відповідно забезпечує відображення інформації за методом «темного» щита, а також приймання сигналів станів командно-квитируючих ключів.

Розглянемо тепер склад пристрою контрольованого пункту ІУТК або RTU. Структура RTU показана на рисунку 2.3.

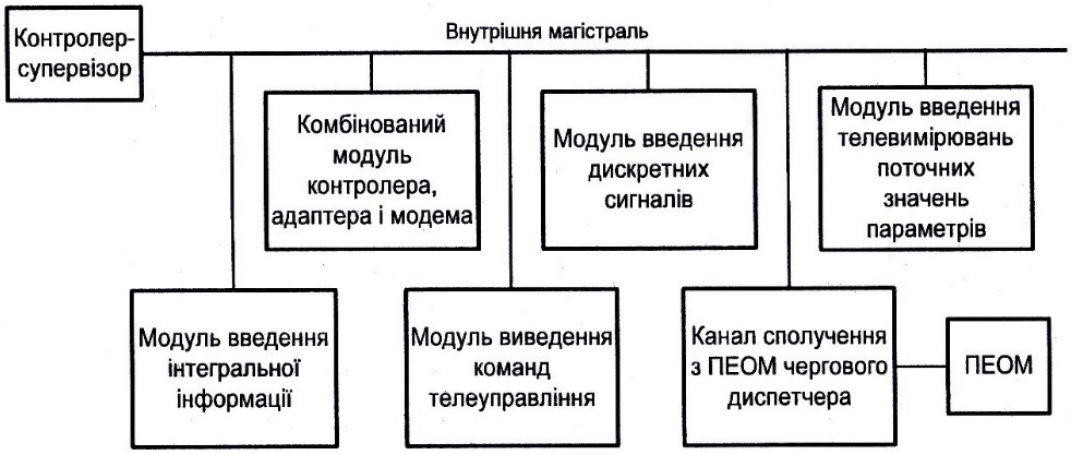


Рисунок 2.3 – Структура контрольованого пункту

Пристрій КП, так само як і пристрій ПУ (ЦППС), побудований за магістрально-модульним принципом. Згідно зі вказаним принципом RTU має у своєму складі такі основні частини: внутрішню магістраль; контролер внутрішньої магістралі; модулі різних типів, які забезпечують виконання відповідних функцій.

Лінії внутрішньої магістралі використовуються для передавання-приймання управляючих, адресних й інформаційних сигналів.

Функції контролера внутрішньої магістралі є аналогічними згадуваного вище контролера внутрішньої магістралі ЦППС.

2.2 Розробка структурної схеми системи автоматичного конфігурування телемеханічних засобів

Структурна схема системи автоматичного конфігурування телемеханічних засобів представлена у Додатку 1 і складається із таких блоків: «Конфігуратор», «Налаштування Конфігуратора», «Вибір параметрів шрифтів всіх функціональних елементів», «Локалізація на трьох мовах: українській, англійській і російській», «Дані», «Налаштування всіх елементів», «Збереження/ Заванта-

ження проекту у форматі xml», «Блок, що працює по протоколу «Гранит Ethernet TCP/IP», «Блок, що працює за протоколом «МЭК 870–5–101», «Блок, що працює за протоколом «МЭК 870–5–104», «Блок, що працює за протоколом «Гранит», «Блок, що працює за протоколом «Гранит-микро», «Модуль автоматичної побудови моделі реального багаторівневого ІУТК», «Блок відображення даних», «Блок відображення протоколів «МЭК 870–5–101» та «МЭК 870–5–104», «Блок відображення протоколу «Modbus TCP», «Блок відображення протоколів «Гранит», «Гранит-микро» та «Гранит Ethernet TCP/IP», «Налаштування ASDU», «Опис об`єктів», «Пристрої», «Опис об`єктів пристроїв», «Стисла структура у вигляді дерева», «Розгорнута структура КП (RTU) у вигляді дерева», «Графічне відображення структури КП (RTU)», «Детальний опис вмісту вибраного об`єкта», «Налаштування вмісту модулів КП (RTU)», «Редагування модулів».

Центром структурної схеми Конфігуратора телемеханічних засобів є блок «Конфігуратор».

Від блока «Конфігуратор» відходять блоки «Дані» (уособлює всі об’єкти у програмі) та «Налаштування Конфігуратора» (відповідає за налаштування конфігуратора).

З блоком «Налаштування конфігуратора» з’єднуються блоки «Вибір параметрів шрифтів всіх функціональних елементів» (відповідає за налаштування параметрів шрифтів у візуальному елементі у вигляді структури-дерева (далі у тексті використовується його назва у середовищі розробки Visual Studio – TreeView), у візуальному елементі відображення даних у вигляді: «назва поля класа» – «значення поля класа» (далі у тексті використовується його назва у середовищі розробки Visual Studio – PropertyGrid), графічної панелі та всіх форм) та «Локалізація на трьох мовах: українській, англійській та російській» (відповідає за встановлення мови, яка зручна для користувача).

Блок «Дані» з’єднуються із наступними блоками:

1) «Налаштування всіх елементів» – кожен об’єкт (крім ідентифікаційного номера) може редагуватися користувачем;

2) «Збереження/ Завантаження проекту у форматі xml» – блок збереження / завантаження проекту;
3) «Блок, що працює по протоколу «Гранит Ethernet TCP/IP» – описує об’єкти протоколу «Гранит Ethernet TCP/IP»;
4) «Блок, що працює по стандарту «МЭК 870–5–101» – описує об’єкти протоколу «МЭК 870–5–101»;
5) «Блок, що працює по стандарту «МЭК 870–5–104» – описує об’єкти протоколу «МЭК 870–5–104»;
6) «Блок, що працює по протоколу «Гранит» – описує об’єкти протоколу «Гранит»;
7) «Блок, що працює по протоколу «Гранит-микро» – описує об’єкти протоколу «Гранит-микро»;
8) «Блок відображення даних» – відповідає за взаємодію із користувачем – відображає необхідну йому інформацію про об’єкти та змінює їх відповідно до вказівок користувача.

«Блок відображення даних» містить такі блоки:

а) «Блок відображення стандартів «МЭК 870–5–101» та «МЭК 870–5–104», до якого ланцюжком під’єднані «Налаштування ASDU» і потім «Опис об’єктів» – блок відображає інформацію про об’єкти стандартів «МЭК 870–5–101» та «МЭК 870–5–104» та змінює їх відповідно до вказівок користувача;
б) «Блок відображення стандарту «Modbus TCP», до якого ланцюжком під’єднані «Пристрої» і потім «Опис об’єктів пристрою» – блок відображає інформацію про об’єкти стандартів «Modbus TCP» та змінює їх відповідно до вказівок користувача;
в) «Блок відображення стандартів «Гранит», «Гранит-микро» та «Гранит Ethernet TCP/IP» – блок відображає інформацію про об’єкти стандартів «Гранит», «Гранит-микро» та «Гранит Ethernet TCP/IP» та змінює їх відповідно до вказівок користувача.

У свою чергу, «Блок відображення стандартів «Гранит», «Гранит-микро» та «Гранит Ethernet TCP/IP» містить такі ланцюжки блоків:

1) «Стисла структура у вигляді дерева» – відповідає за відображення елементом TreeView стислої структури у вигляді дерева у головній формі Конфігуратора (далі у тексті використовується назва FormBase), яка наведена у Додатку 3;
2) «Розгорнута структура КП (RTU) у вигляді дерева» – відповідає за відображення елементом TreeView розгорнутої структури у вигляді дерева у формі, призначеної для розгорнутого налаштування модулів RTU і каналів зв'язку (далі у тексті використовується назва FormModules), яка наведена у Додатку 6;
3) «Графічне відображення структури КП (RTU)» – відповідає за відображення структури КП (RTU) у вигляді графічної панелі у вікні FormModules;
4) «Детальний опис вмісту вибраного об`єкта» – відповідає за опис вмісту об'єкта у елементі PropertyGrid;
5) «Налаштування вмісту Модулів КП (RTU)» – функція можлива з графічної панелі, або елемента PropertyGrid, або спеціалізованої форми для додавання модулів у КП (RTU) (далі у тексті використовується назва Form_AddingModule);
6) «Редагування модулів» – відповідає за налаштування модулів.

Вузол «Модуль автоматичної побудови моделі багаторівневого ІУТК» випливає із блоків, які використовує у роботі – це «Блок, що працює по протоколу «Гранит»» та «Блок, що працює по протоколу «Гранит-микро».

2.3 Розробка блок-схеми алгоритму роботи модуля автоматичної побудови моделі багаторівневого ІУТК

Розглянемо тепер алгоритм роботи модуля автоматичної побудови моделі багаторівневого ІУТК за структурною схемою, наведеною у Додатку 2.

Блок-схема розпочинається з блока «Початок».

Блок «Створення з’єднання відповідно до заданих характеристик» – створює з’єднання, використовуючи дані підоб’єкта «Serial Options», що знаходиться у вузлі «Granit Com Port» елемента TreeView форми FormBase.

Далі здійснюється «Отримання даних від користувача про адреси RTU (КП) для пошуку». Якщо користувач ввів діапазон номерів RTU (КП), то він передається для сканування, інакше передається діапазон номерів RTU (КП) від 0 до 64.

Після цього відбувається опитування нульового місця модулів кожного RTU (КП) обраного діапазону адрес, і якщо це місце має модуль, то опитуються ще 8 місць (це найбільший універсальний розмір) цього ж RTU (КП).

Опитування розпочинається із створення запиту (вузол «Відправлення запиту типу модуля відповідно до позиції та протокола») по протоколу «Гранит-микро» (для новіших модулів) відповідно до місця модуля і адреси.

Після цього працює підпрограма «Підпрограма пошуку та обробки пакета».

Якщо відповіді на запит немає, то програма повертає «Ні», якщо ж відповідь є, то відбувається пошук заголовку пакета (чотири байти). Якщо заголовок не знайдений, то перевіряється переповнення тайм-ауту, якщо він переповнений, то програма повертає «Ні», а якщо не переповнений, то знову відбувається пошук заголовку. Якщо заголовок знайдений, то зчитується інформаційна частина пакета, яка складається із даних та двох байтів перевірки завадостійким циклічним кодом (CRC16). Програма сама генерує два байти CRC16 з отриманих даних та порівнює з пакетними. Якщо вони співпадають, то це означає, що пакет цілісний і підпрограма повертає «Так», інакше знову перевіряється переповнення тайм-ауту.

Після цього здійснюється «Перевірка коректності інформаційних байтів» шляхом порівняння адреси, місця та каналу очікуваної послідовності та отриманих байтів. Якщо вони не співпадають, то відбувається перевірка переповнення тайм-ауту, інакше виокремлюється байт типу модуля.

Якщо тип модуля співпадає з набором очікуваних, то модуль додається у програму до RTU (КП), якщо такий модуль є першим (орієнтовно може бути: комбінованим модулем контролера, адаптера і модема (далі у тексті – КАМ); модулем контролера, адаптера мережі (сетей) (далі у тексті – КАС) або модулем контролер-накопичувач-шлюз (далі у тексті – КНШ)), то перед додаванням модуля створюється і сам RTU (КП).

Якщо тип модуля є модуль розгалуження (КАМ, М2М (модуль двоканального модема), М4А1 (модуль чотириканального лінійного адаптера), М4А2 (модуль чотириканального лінійного адаптера), КНШ), то відправляється запит діапазону адрес ретрансляції даних відповідно до позиції та каналу. Після цього працює підпрограма «Підпрограма пошуку та обробки пакета». Після отримання коректних діапазонів адрес RTU (КП), на їх основі відбувається опитування нульового місця модулів кожного RTU (КП) обраного діапазону адрес, інакше кажучи, програма спускається на нижчий рівень RTU (КП) та далі виконується рекурсивно.

Коли весь діапазон адрес першого рівня пройдений, це означає, що вся структура побудована і досягнуто блока «Кінець».

Висновки до розділу 2

Детально розглянуто структури сучасного інформаційно-управляючого телемеханічного комплексу, а також його найбільш важливих та відповідальних частин – центральної приймально–передавальної станції та RTU, з метою врахування їх складу і зв’язків між ними при розробці відповідного конфігуратора телемеханічних засобів.

Розроблена структурна схема конфігуратора.

Також розроблена блок-схема алгоритму роботи модуля автоматичної побудови моделі багаторівневого ІУТК.

3 РОЗРОБКА ПРОГРАМНОЇ ЧАСТИНИ

3.1 Вибір і обґрунтування мови програмування

Мова програмування С

С – універсальна, процедурна, імперативна мова програмування загального призначення.

Як і більшість імперативних мов, заснованих на традиції АЛГОЛ, С має можливості для структурного програмування і дозволяє здійснювати рекурсії, у той час, як система статичної типізації даних запобігає виникненню багатьох непередбачуваних операцій. У С увесь виконуваний код міститься у функціях. Параметри функції завжди передаються за значеннями. Передача параметрів за вказівником реалізовується шляхом передачі значення вказівника. Гетерогенні сукупності типів даних (структури) дозволяють пов'язаним типам даних бути об'єднаними і маніпулювати ними, як єдиним цілим.

Мова програмування С++

С++ – мова програмування високого рівня з підтримкою до кількох парадигм програмування: об'єктно-орієнтованої, узагальненої та процедурної.

Мову використовують для системного програмування, розробки програмного забезпечення, написання драйверів, потужних серверних та клієнтських програм, а також для розробки розважальних програм, наприклад, відеоігор.

Перевагами мови С++ є: швидкодія;масштабованість;можливість роботи на низькому рівні з пам'яттю, адресами, портами. (Що, при необережному використанні, може легко перетворитися на недолік);можливість створення узагальнених алгоритмів для різних типів даних, їхня спеціалізація, і обчислення на етапі компіляції, з використанням шаблонів;підтримуються різні стилі та технології програмування, включаючи традиційне директивне програмування, ООП, узагальнене програмування, метапрограмування (шаблони, макроси).

До недоліків мови С++ належать: наявність безліч можливостей, які порушують принципи безпеки типів, що призводить до важко помітних

помилок;погана підтримка модульності;нестача інформації про типи даних під час компіляції (CTTI);мова C++ є складною для вивчення і для компіляції; деякі перетворення типів неінтуїтивні. Зокрема, операція над беззнаковим і знаковим числами видає беззнаковий результат; препроцесор C++ (успадкований від C) дуже примітивний. хоча декларується, що C++ – мультипарадигмена мова, реально в мові відсутня підтримка функціонального програмування.

Мова програмування C#

C# – об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET.

Синтаксис C# близький до C++ і Java. Мова має чітку статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. Перейнявши багато чого від своїх попередників – мов C++, Delphi, Модула і Smalltalk – C#, спираючись на практику їхнього використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад, множинне спадкування класів (на відміну від C++);

C# успадкувала від Java концепції віртуальної машини (середовище .NET), байт-коду (MSIL) і більшої безпеки вихідного коду програм, плюс врахувала досвід використання програм на Java.

Нововведенням C# стала можливість полегшеної взаємодії, порівняно з мовами-попередниками, з кодом програм, написаних на інших мовах, що є важливим при створенні великих проектів. Якщо програми на різних мовах виконуються на платформі .NET, .NET бере на себе завдання щодо сумісності програм (тобто типів даних, за кінцевим рахунком).

Мова програмування Java

Java – об'єктно-орієнтована мова програмування, випущена 1995 року компанією «Sun Microsystems» як основний компонент платформи Java.

Мова значно запозичила синтаксис із C і C++. Зокрема, взято за основу об'єктну модель C++, проте її модифіковано. Усунуто можливість появи деяких

конфліктних ситуацій, які могли виникнути через помилки програміста та полегшити сам процес розробки об'єктно-орієнтованих програм. Ряд дій, які в C/C++ повинні здійснювати програмісти, доручено віртуальній машині. Передусім Java розроблялась як платформо-незалежна мова, тому вона має менше низькорівневих можливостей для роботи з апаратним забезпеченням, що в порівнянні, наприклад, з C++ зменшує швидкість роботи програм. За необхідності таких дій Java дозволяє викликати підпрограми, написані іншими мовами програмування.

Враховуючи все вищевикладене, було обрано мову C#, тому що вона підтримує багато сучасних технологій: відображення інформації, взаємодії з різними базами даних тощо. C# подібна до C, C++. Простіша і разом із цим чіткіша робота із «посиланнями» на об’єкт, більша надійність мови – виключає деякі моделі, які зарекомендували себе як проблематичні при розробці програмних систем.

Станом на сьогодні C# визначено флагманською мовою корпорації Microsoft, бо вона найповніше використовує нові розширення .NET, що дозволяє C# використовувати CLR – це надає багато можливостей, яких позбавлені «класичні» мови програмування. Наприклад, «збір сміття» не реалізований в самому C#, а проводиться CLR для програм, написаних на C#. Решта мов програмування, хоч і підтримуються, але визнані такими, що мають спадкові прогалини щодо використання .NET.

3.2 Опис конфігуратора телемеханічних засобів

Пропонується розробка конфігуратора, який відповідає всім вищевказаним вимогам і працює із телемеханічними засобами згідно із протоколами МЭК 870–5–101, МЭК 870–5–104, Modbus RTU, Modbus TCP, Гранит, Гранит-микро.

Знайомство користувача з конфігуратором розпочинається із основної форми програми – FormBase (Додаток 3).

Зверху розташований функціональний елемент меню (далі у тексті використовується його назва у середовищі розробки Visual Studio – MenuStrip). У

ньому є вкладка «File» із розширеннями «New», «Open», «Save», які, відповідно, створює новий файл, відкриває існуючий файл та зберігає файл у форматі xml (Додаток 9), що дозволяє у разі необхідності без застосування Конфігуратора здійснити редагування структури телемеханічної системи.

Наступна вкладка «Project» має розширення «Export» і «Import». Вони створені для майбутнього розширення функціональності.

Вкладка «Options» викликає форму «Options» (Додаток 4), за допомогою якої користувач може налагодити програму так, як йому зручно (мову, шрифти та відображення кнопок-стрілок для змінення місця модуля у RTU (КП) – реалізовано для форми FormModules (Додаток 6).

Та, нарешті, вкладка «Quit», яка надає додаткову можливість виходу із програми, яка запитує підтвердження дій від користувача.

Під елементом MenuStrip розташований елемент ToolStrip, який функціонально виконує все те саме, що і MenuStrip, але на відміну від нього, кожен елемент має малюнок-кнопку із графічним зображенням, яке інтуїтивно дозволяє користувачу знайти потрібний елемент. Таке дублювання функціональності створене для того, щоб користувач міг вибрати ті інструменти для роботи, якими йому зручніше оперувати.

Зліва (у Додатоку 3) розташований елемент TreeView. Дані у програмі представлені у вигляді вузлів, які об’єднані у дерево. У програмі існує два дерева – TN_T_Proto (містить повністю всю інформацію про дані і не показується повністю користувачу) та зменшену версію TN_T_Proto, яка і відображається в TreeView основної форми. Це дозволяє збільшити чіткість аналізу системи людиною.

Перший рівень дерева має два основних вузла – «АРМ» та «Server». Їх може бути декілька. «АРМ» має шість вузлів, які відповідають за роботу із протоколами:

1) «Granit» – відповідає за з’єднання через послідовний порт по протоколах «Гранит» та «Гранит-микро». Має вузли «Ведомые» (групує всі вузли «Ведомый») і «Ведущие» (групує всі вузли «Ведущий»). Всі нижчі

вузли можуть бути тільки «RTU» (у вікні FormModules відображаються вузли модулів та каналів);

2) «IEC 870–5–101» – відповідає за зв’язок по стандарту МЭК 870–5–101. Нижчі вузли – «SlavesList» та «Master» – мають підпорядковані вузли «Slave» (підпорядкований). Вузол «Slave» має підконтрольні вузли «ASDU» (Application Service Data Unit – Блок даних служби додатків);
3) «Granit TCP» – відповідає за з’єднання через Ethernet по протоколах «Гранит» та «Гранит-микро». Нижчі вузли – «ClientsList» та «Server TCP» – мають підпорядковані вузли «Client», які містять RTU;
4) «IEC 870–5–104» – відповідає за зв’язок по стандарту МЭК 870–5–104. Нижчі вузли – «ClientsList» та «Server 104» – мають підпорядковані вузли «Client 104» та «Client» відповідно. «Client» має підконтрольні вузли «ASDU»;
5) «Modbus-TCP» – відповідає за зв’язок по стандарту Modbus TCP. Нижчі вузли – «ClientsList» та «Server TCP» – мають підпорядковані вузли «Client», які, у свою чергу, мають вузли «Device».

Кожен об`єкт у системі має індивідуальний ідентифікаційний номер (ID), що дозволяє ефективно працювати з об`єктами, які не зв’язані структурно. ID створюється програмно і користувач не може його змінити, а тільки подивитись його значення – це зроблено для уникнення помилок.

Вибравши будь-який вузол дерева правою кнопкою миші, можна побачити елемент випадаюче меню (contextMenuStrip). Відповідно до типу об’єкта, його можна видалити, вирізати, вставити, копіювати, а якщо це RTU(КП) – то ще буде й вкладка «Подробно…», яка відкриває форму FormModules (Додаток 6). Для елемента «Ведущий» передбачена вкладка «Сканировать», вибравши яку з’являється форма Form_Address_range_of_polls (рисунок 3.1) для введення діапазону адрес RTU (КП), які будуть скануватися. Після введення діапазону, конфігуратор автоматично будує модель багаторівневого ІУТК по протоколам «Гранит» та «Гранит-микро».

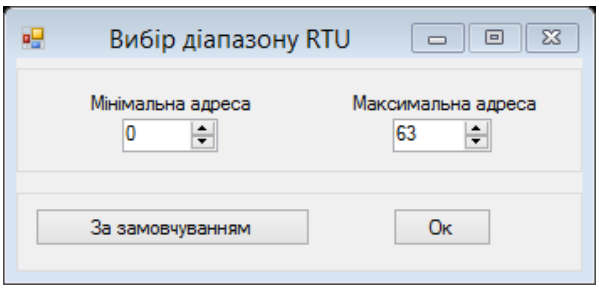


Рисунок 3.1 – Форма налаштування діапазону адрес Form_Address_range_of_polls

Алгоритм роботи такий. На основі даних, які введено у елемент «Ведущий» створюється з'єднання по послідовному порту із вказаними характеристиками. Після цього створюються запит, відповідь на який очікує програма у вигляді пакету даних, у якому буде ID типу модуля відповідно до номера RTU і місця модуля у ньому. Якщо відповіді немає або надходить інформація, але не та, що потрібно (можуть з'являтися системні пакети), то конфігуратор це розпізнає і RTU (також паралельно з цим модуль) не додається у програму. Якщо ж байт ID типу модуля існує і у списку ID-байтів модулів, то цей модуль додається до RTU (якщо модуль перший у ньому, то створюється і сам RTU).

У випадку, якщо коли доданий модуль KAM, M2M, M4A2 або M4A1, то додатково створюється спеціальний запит (або запити) для перевірки діапазонів каналів (кількість каналів та діапазонів відома одразу відповідно до типу модуля), що дає змогу дізнатися, скільки є під`єднаних RTU через цей модуль – або жодного (приклад інформаційних байтів відповіді 00-FF), або один – згідно із радіальним принципом (приклад інформаційних байтів відповіді 03–03), або два і більше – відповідно до магістрально-модульного принципу (приклад інформаційних байтів відповіді 03–05). Система діапазонів введена для можливості гнучкого розширення системи, щоб не переналаштовувати її всю через адреси нових RTU (КП). Якщо отримано інформацію про існування ще не створених у програмі RTU (КП), то алгоритм перетворюється у рекурсію – знову створюється запит, відповідь на який очікує програма у вигляді пакету даних, у якому буде ID типу модуля відповідно до номера RTU і місця модуля у ньому.

Вибравши будь-який вузол дерева лівою кнопкою миші, з'явиться докладна інформація про об'єкт правіше від дерева у елементі PropertyGrid. Інформа-

ція (як показано у Додатку 3) подається у вигляді: «назва поля класа» – «значення поля класа». Кожне таке поле можна редагувати (крім ID об'єкта). В Додатку 3 показано, що у елементі PropertyGrid відображається RTU, яке має назву – «Лук`янівська», ID – 237, Адресу КП – 0, та багато інших налаштувань. Також є блок редагування існуючих модулів. Якщо в цьому блоці вибрати «Подробнее…», то відкриється форма FormModules (Додаток 6).

У вікні FormModules елемент TreeView відображає частину TN_T_Proto, перший вузол якої може бути тільки RTU, який найвищий ієрархічно у дереві. Принцип роботи TreeView і PropertyGrid у цьому вікні такий самий, як і у формі FormBase. Зліва від TreeView розташоване графічне відображення RTU, яке вибране у TreeView (вузол додатково виділяється сірим кольором), та RTU, які знаходяться вище та нижче відповідно до ієрархії. Як показано в Додатку 6, відображаються також і модулі, які містять RTU, назви RTU та назви каналів. Якщо увімкнено «керування стрілками» в опціях, то можна переміщувати модулі на інші місця не тільки стрілками-кнопками, а й з клавіатури. Ця графічна панель масштабується та пересувається у разі необхідності. Якщо користувач згоден із змінами, які він виконав, то потрібно натиснути кнопку «Применить», інакше все залишиться як і було.

Для заповнення RTU модулями, створена форма Form_AddingModule (Додаток 5), яка викликається або із PropertyGrid, якщо виділений RTU у TreeView, або ж у графічній частині FormModules через локальне контекстне меню. Для того, щоб додати модуль (або видалити), необхідно його вибрати зліва у формі та натиснути «Добавить модуль». Справа відображається коротка інформація про вибраний модуль.

Якщо модуль вже існує, то викликається форма Choice_Form_for_module (Рисунок 3.2), яка дозволяє здійснити вибір подальшої роботи. Це тимчасове рішення – у майбутньому планується замінити це вікно на контекстне меню у PropertyGrid. При натисненні на кнопку «Перехід на форму заміни модуля» з'явиться вікно Form_AddingModule, а якщо натиснути «Перехід на форму властивостей модуля», то з'явиться форма для цього модуля.

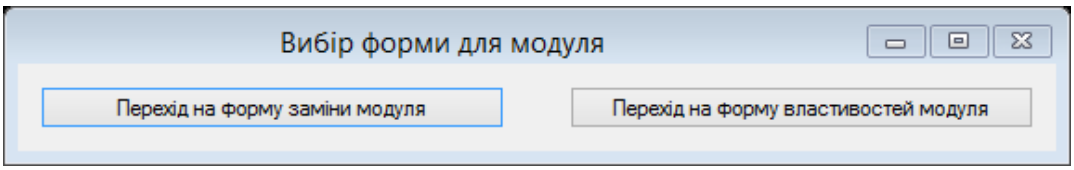


Рисунок 3.2 – Форма «Вибір форми для модуля»

Кожен модуль має певну форму для його налаштування.

Для модулів КАМ, КАС, КНШ, М2М, М4А1, М4А2, Модуль введення телевимірювань інтегральних(МТІ), Модуль введення телевимірювань поточних значень вимірюваних параметрів(МТТ), Модуль введення дискретних сигналів (МДС), Модуль контролера щита (МКЩ), Device або ASDU передбачена форма Detail_of_Objects (Додаток 7).

Для редагування параметрів об`єктів та керування самими об`єктами Device або ASDU створена форма Redaction_Sensors (Додаток 8).

ASDU і Device мають два типи датчиків (цифрові, аналогові) та управляючі механізми. Для цих всіх типів підходить форма Redaction_Sensors. При натисненні кнопки «Добавить датчик» додається відповідний датчик із індивідуальним ID (крім глобального) на весь елемент ASDU. Натиснення на кнопку «Удалить датчик» призведе до видалення об’єкта.

Програма має локалізацію на трьох мовах – українській, англійській та російській. В Додатку 3 показані ті ж об’єкти, але різними локалізаціями.

Для збереження даних проекту можливо було б використовувати базу даних (БД), що дозволяє працювати швидше, чим з xml файлами. Але базу даних потрібно постійно підтримувати, конфігуратор має достатньо складну ієрархію і через це потрібно використовувати деревоподібну БД, що робить її заплутаною та незручною, перенесення БД потребує додаткових зусиль, виникає складність із сумісностями версій – через ці недоліки було обрано збереження проекту у форматі xml. Це надає можливість легко доповнювати або змінювати модель, без суттєвих витрат, що дозволяє спеціалісту налаштувати телемеханічну систему віддалено або проаналізувати її без використання якихось спеціалізованих програм. Приклад серіалізації представлено в Додатку 9.

Також розглядаються можливі залежності між об'єктами рівня користувальницької і системної моделей. Коли зміна малої частини верхнього рівня призводить до зміни залежної від неї частини нижнього, то ця зміна відіб'ється і на інших частинах верхнього рівня.

Дуже гостре питання – це залежність модулів програми (тобто конфігуратора). Тому в подальшому в конфігураторі буде створено спосіб розв’язання ситуації конфлікту модулів по ресурсам, інакше розробка кожного нового доповнення буде справою все більш трудомісткою і ненадійною.

3.3 Експериментальна перевірка розробленого конфігуратора

Основні форми Конфігуратора показано у Додатках 3–8, а їх роботу та основні функції і можливості описано вище.

Безпосередньо експериментальну перевірку доцільно проводити, розглядаючи роботу модуля автоматичної побудови моделі багаторівневого ІУТК за протоколами «Гранит» та «Гранит-микро».

Експериментальні дослідження проводились із двома RTU. «Перший» (правий RTU (КП) у Додатку 10) може містити три модулі та під’єднаний по RS-232 до комп’ютера, а також через власний модуль КАМ – до «другого». «Другий» (лівий RTU (КП) у Додатку 10) може містити чотири модулі та сполучається із «першим» також через власний модуль КАМ.

Як видно з Додатку 10, у першому випробуванні «перший» RTU має у складі такі модулі: КАМ (0 місце), М4А2 (1 місце) і МДС (2 місце), а «другий» RTU – КАМ (0 місце), КНШ4 (1 місце), КАМ (3 місце).

У другому випробуванні (Додаток 10) «перший» RTU має у складі наступні модулі: КАМ (0 місце), КНШ (1 місце) і М4А2 (2 місце), а «другий» RTU – КАМ (0 місце), МДС (1 місце), М4А1 (2 місце), КАМ (3 місце).

Щоб побудувати моделі багаторівневого ІУТК у Конфігураторі, потрібно для вузла «Ведущий» (Додаток 3) викликати контекстне меню, в якому вибрати «Сканировать». Тоді з’явиться форма налаштування діапазону адрес Form_Address_range_of_polls (рисунок 3.2). Необхідно вибрати діапазон; нехай

він буде від 0 до 12 (припускається, що RTU з випробовувань будуть мати адреси з цього діапазону). Опитуються спочатку нульові місця із заданого діапазону адрес RTU (КП) і, якщо це місце не пусте, то опитуються ще 8 місць RTU (це найбільша універсальна кількість). Це займає деякий час, в залежності від обраного діапазону і реально існуючих RTU сканованої структури.

Після того, як процес завершився, для перегляду розгорнутої структури необхідно перейти у форму FormModules (це можна зробити подвійним кліком на RTU в дереві TreeView форми FormBase або у PropertyGrid (вибравши перед цим у дереві TreeView RTU) через кнопку «Подробнее…»). Результати представлені в Додатку 11 для двох випробувань, де під стрілкою 1 показано графічне відображення досліджуваної структури і можливо оцінити, на яких місцях RTU знаходяться його модулі; під стрілкою 2 досліджувана структура у вигляді дерева і під стрілкою 3 наведено відображення параметрів «другого» RTU (КП) у елементі PropertyGrid, де також можна оцінити, на яких місцях RTU знаходяться його модулі.

З представлених результатів (Додаток 11) і наведених досліджуваних структур (Додаток 10), можна зробити висновок, що модуль автоматичної побудови моделі багаторівневого ІУТК за протоколами «Гранит» та «Гранит-микро» відповідає поставленим вимогам і правильно та чітко виконує поставлені перед ним задачі.

Зрозуміло, що структуру можна значно розширити (додавши до 62 RTU (а то і більше) на 9 місць і відповідно матимемо 9*62 = 558 модулів), але і ця спрощена модель дозволяє показати саму суть свого призначення: замість того, щоб людина створювала структуру багаторівневого ІУТК і прописувала параметри об’єктів вручну, це все виконується автоматично (вписати потрібно тільки назви RTU) та без зайвих помилок (звичайно, якщо у фізичних налаштуваннях їх немає, але і тут можлива модернізація модуля автоматичної побудови, який зможе виявити, навіть, такі помилки, що призведе до зменшення їх кількості, яка і так є мінімальною).

Висновки до розділу 3

Розроблений конфігуратор телемеханічних засобів має такі переваги:

1) конфігуратор розроблений із застосуванням мови C#, яку визначено флагманською мовою корпорації Microsoft, тому що вона найповніше використовує нові можливості .NET, підтримує багато сучасних технологій відображення інформації, взаємодії з різними базами даних тощо;
2) застосування шаблонів проектування – один з інструментів розробника, який допомагає йому заощадити час і зробити рішення якіснішим;
3) наведена наочність інтерфейсу користувача робить конфігуратор ергономічним та дозволяє легко розібратись у структурі телемеханічної системи та швидко налаштувати необхідні елементи, додавати датчики, розширити або навпаки – видалити частину пристроїв;
4) збереження проекту у форматі xml надає можливість легко доповнювати або змінювати модель, без суттєвих витрат, що дозволяє спеціалісту налаштувати телемеханічну систему віддалено або проаналізувати її без використання якихось спеціалізованих програм. При завантаженні великого проекту із файлу xml можна чекати декілька секунд (а той ще більше). Для усунення даної незручності використовується метод відкладеного завантаження, що дозволяє завантажувати не весь проект одразу, а тільки ті елементи, які цікавлять користувача (завантаження елемента виконується в момент його вибору, затримка завантаження конкретного елемента мізерна);
5) оскільки конфігуратор – це програмний продукт, то одноразово написане доповнення зберігає працездатність у будь-яких майбутніх версіях програми, якщо тільки новий конфігуратор декларує зворотну сумісність зі старою версією. При цьому поява будь-яких інших додатків не заважає роботі раніше написаного (за винятком випадку, коли нове доповнення створюється замість старого).

ВИСНОВКИ

У даній роботі вперше розроблено систему автоматичного конфігурування телемеханічних засобів. Розроблені структурна схема конфігуратора телемеханічних засобів та блок-схема алгоритму автоматичної побудови моделі реально існуючого багаторівневого ІУТК. Розроблений конфігуратор телемеханічних засобів виконує всі поставлені задачі відповідно до висунутих вимог, що було підтверджено експериментальною перевіркою на реальному ІУТК.

При створенні конфігуратора враховані такі важливі моменти. Оскільки конфігуратор – це програмний продукт, то одноразово написане доповнення буде зберігати працездатність в будь-яких майбутніх версіях програми, якщо тільки новий конфігуратор декларує зворотну сумісність зі старою версією. При цьому поява будь-яких інших додатків не буде заважати роботі раніше написаного (за винятком випадку, коли нове доповнення робиться замість старого).

Також розглядаються можливі залежності між об'єктами рівня користувальницької і системної моделей. Коли зміна малої частини верхнього рівня призводить до зміни залежної від неї частини нижнього, то ця зміна відіб'ється і на інших частинах верхнього рівня.

Конфігуратор надає зручні інструменти для моделювання користувальницької і системної областей, для написання логіки перетворення зверху вниз і знизу вгору та для створення інтерфейсу, а також інструменти взаємодії між модулями і управління ними.

Інтерфейс користувача не виходить за межі користувальницької об`єктної моделі, щоб можна було швидко, зручно і з найменшою ймовірністю похибки налагодити телемеханічну систему.

Розроблена блок-схема алгоритму роботи модуля автоматичної побудови моделі багаторівневого ІУТК, яка відповідає всім вище вказаним вимогам.

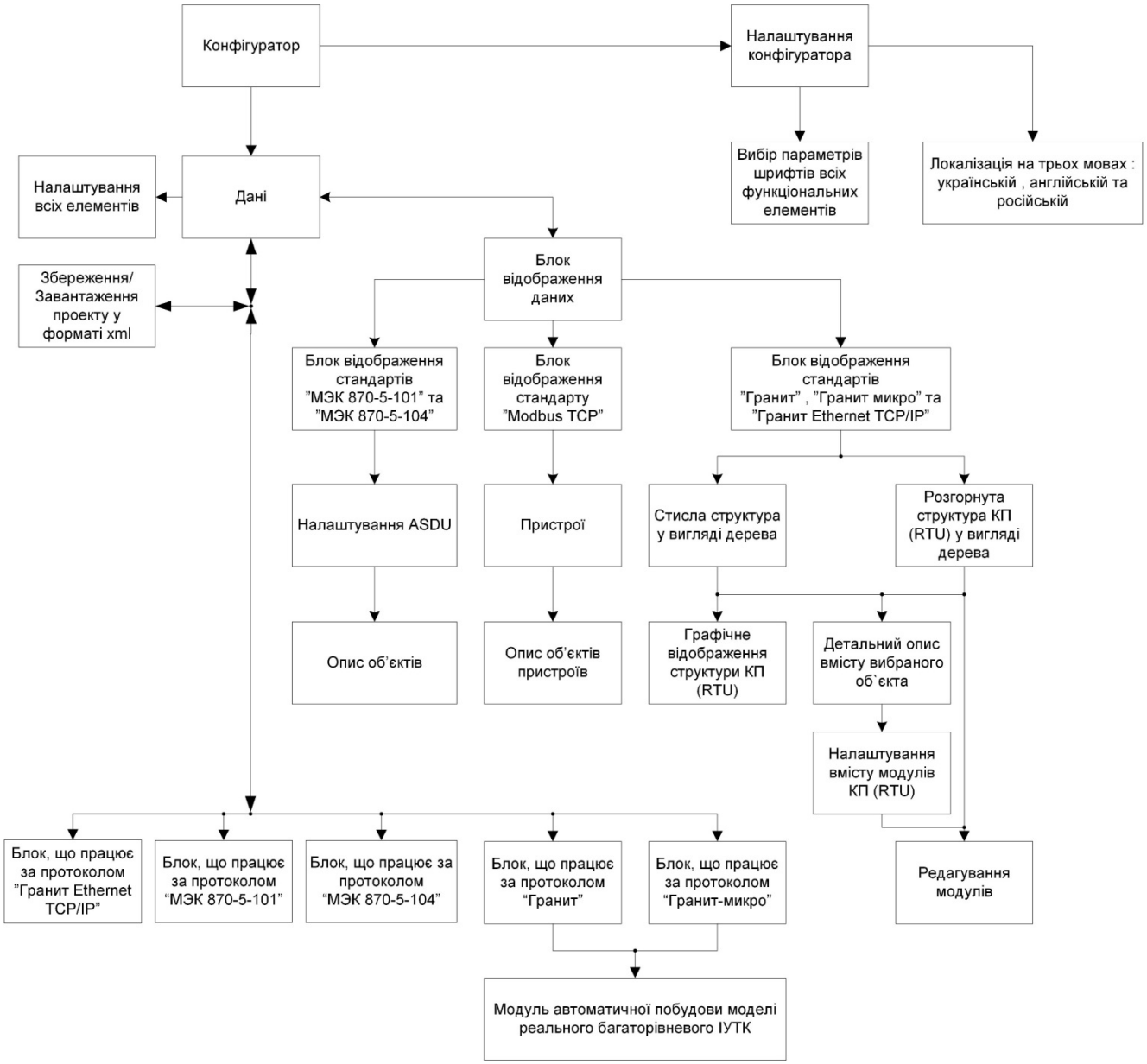
Програма постійно модернізується. У даний час конфігуратор впроваджується на ПП «Науково-виробниче підприємство «Промекс» шляхом інтегрування в базове програмне забезпечення ІУТК.

ДЖЕРЕЛА ІНФОРМАЦІЇ

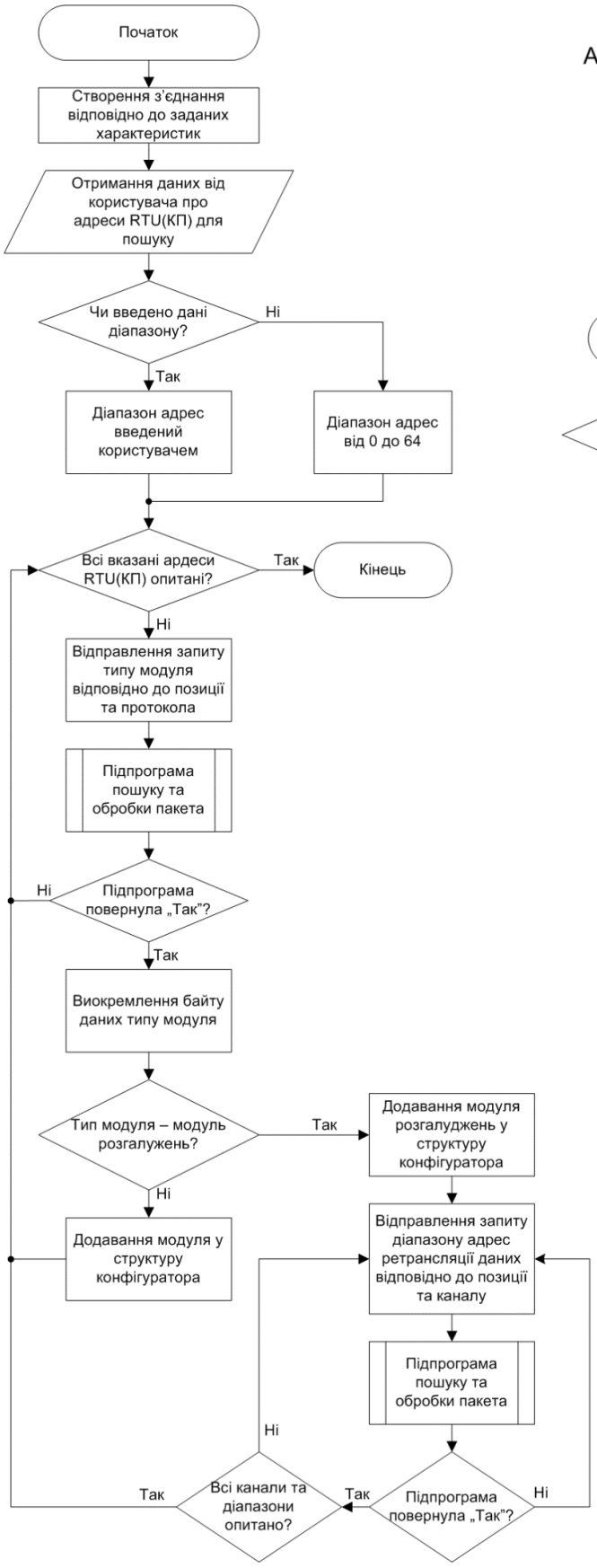
1. КОНФІГУРАТОР// Вісник інженерної академії України. – 2018. – №3.
2. КОНФІГУРАТОР: Тези Всеукраїнської науково-практичної on-line конференції аспірантів, молодих учених та студентів, присвяченої Дню науки (м. Житомир, 16–18 травня 2018 року). – Житомир, 2018. – С. 83–84.
3. КОНФІГУРАТОР. Дослідження сучасного стану та тенденцій розвитку інформаційно-управляючих телемеханічних комплексів: Тези доповідей IX Міжнародної науково-технічної конференції «Інформаційно-комп'ютерні технології 2018» (м. Житомир, 20–21 квітня 2018 року). – Житомир, 2018. – С. 176–177.
4. Мартин, Р. Чистый код: создание, анализ и рефакторинг [Текст]: Библиотека программиста / Р. Мартин. – М.: Питер, 2018. – 446 с.
5. Локтікова, Т.М. Методи та засоби обробки і передачі інформації в системах і мережах передачі даних [Текст]: навчальний посібник / Т.М. Локтікова, А.В. Морозов, В.А. Большой, Н.О. Кушнір. – Житомир: ЖДТУ, 2015. – 162 с.
6. Рихтер, Дж. CLR via C#. Программирование на платформе Microsoft. NET Framework 4.5 на языке C# [Текст]: учебное пособие / Дж. Рихтер. – М.: Питер, 2013. – 896 с.
7. Сороко, В. И. Автоматика, телемеханика, связь и вычислительная техника на железных дорогах России [Текст]: Энциклопедия. Т. 1. / В. И. Сороко, В. М. Каинов, Г. Д. Казиев – М.: НПФ «ПЛАНЕТА», 2006. – 736 с.
8. Портнов Е.М., Ищенко А.С. Системотехника современных информационно-управляющих комплексов // Вісник інженерної академії України. – 2006. – №1. – С. 39–46.
9. Болтенков, В.И. Конфигурирование и настройка автоматизированных информационных систем[Текст]: учебное пособие / В.И. Болтенков, А.Л. Литвинов, Н.В. Лычева. – Белгород: Изд-во БелГУ, 2004. – 81 с.
10. Портнов Е.М. Анализ состояния производства, принципов построения и тенденций развития информационно-управляющих комплексов для АСУ распределенных энергообъектов и производств [Текст]/Е.М. Портнов. – М.: МИЭТ, 2002. – 78 с.
11. Мартин, Д. XML для профессионалов [Текст]: учебное пособие / Д. Мартин, М. Бирбек, М. Кэй и др. – М.: Лори, 2001. – 866 с.

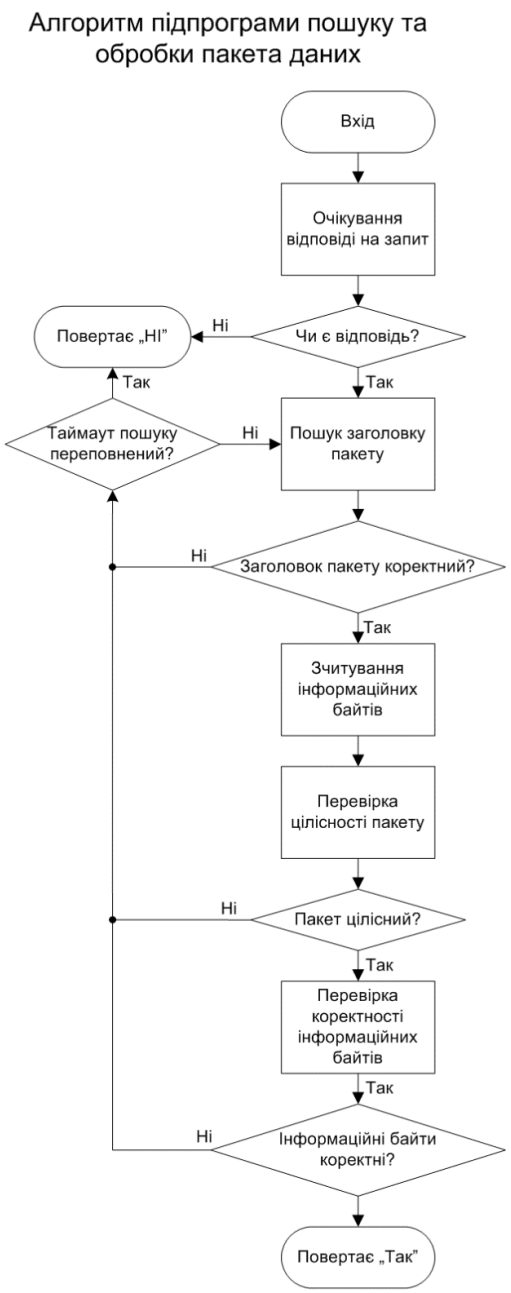
ДОДАТКИ

Структурна схема системи автоматичного конфігурування телемеханічних засобів

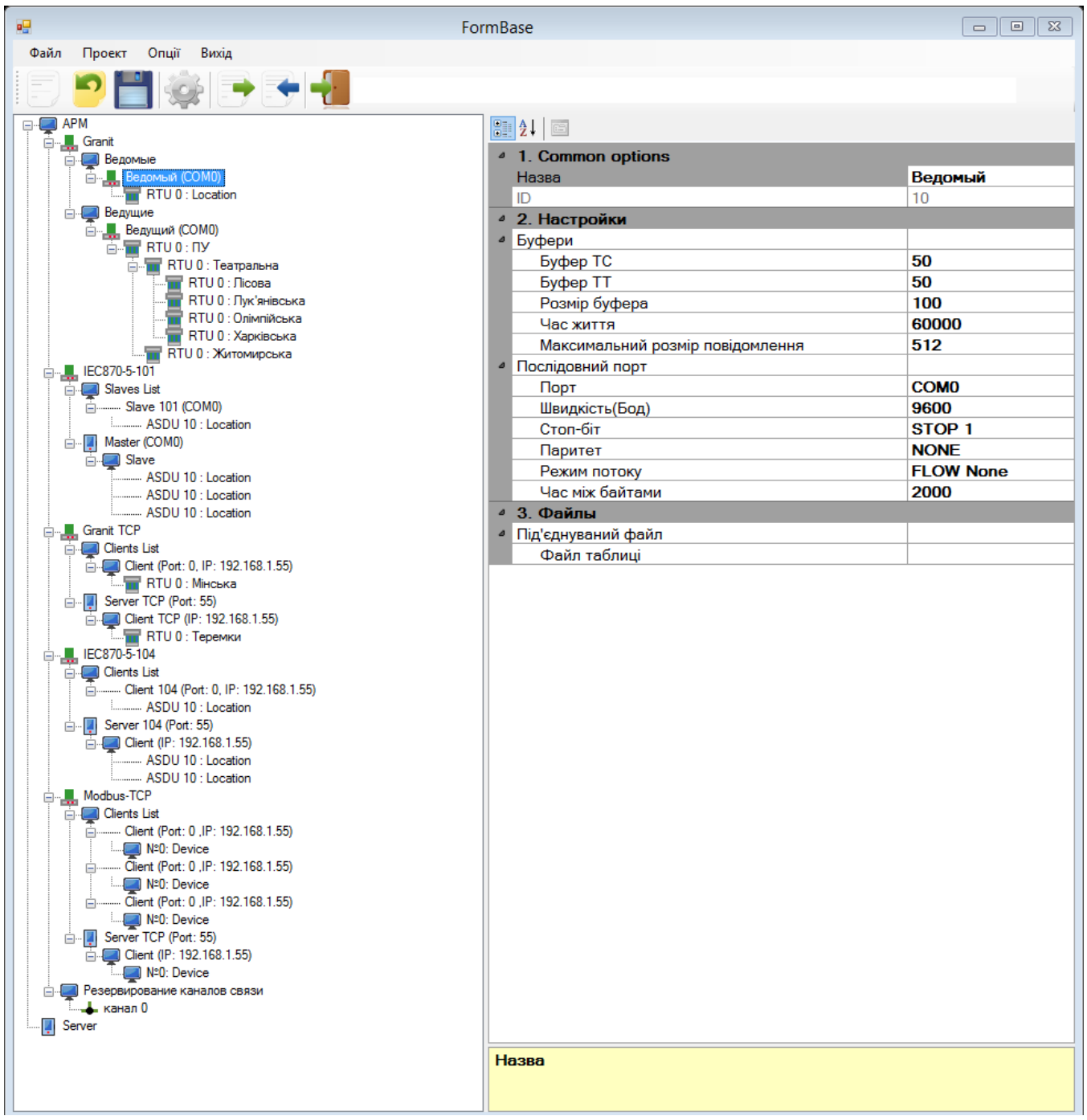


Алгоритм роботи модуля автоматичної побудови моделі багаторівневого ІУТК

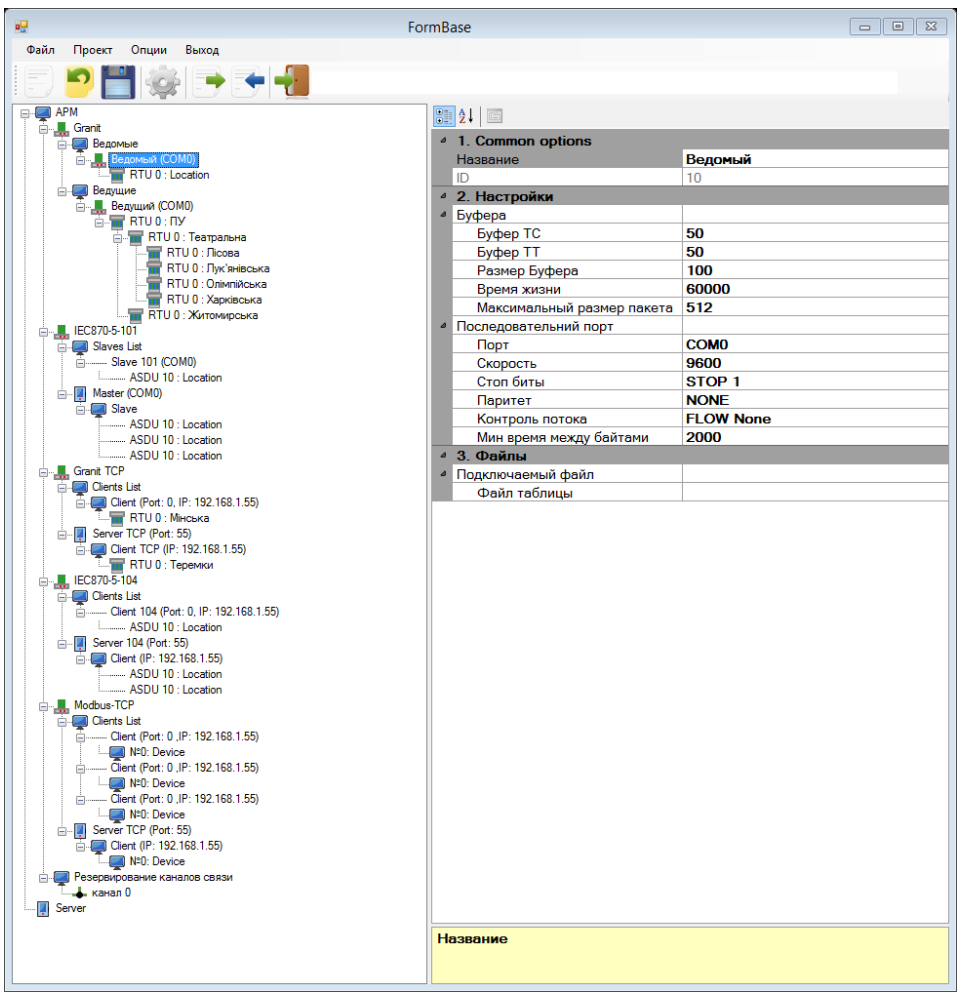




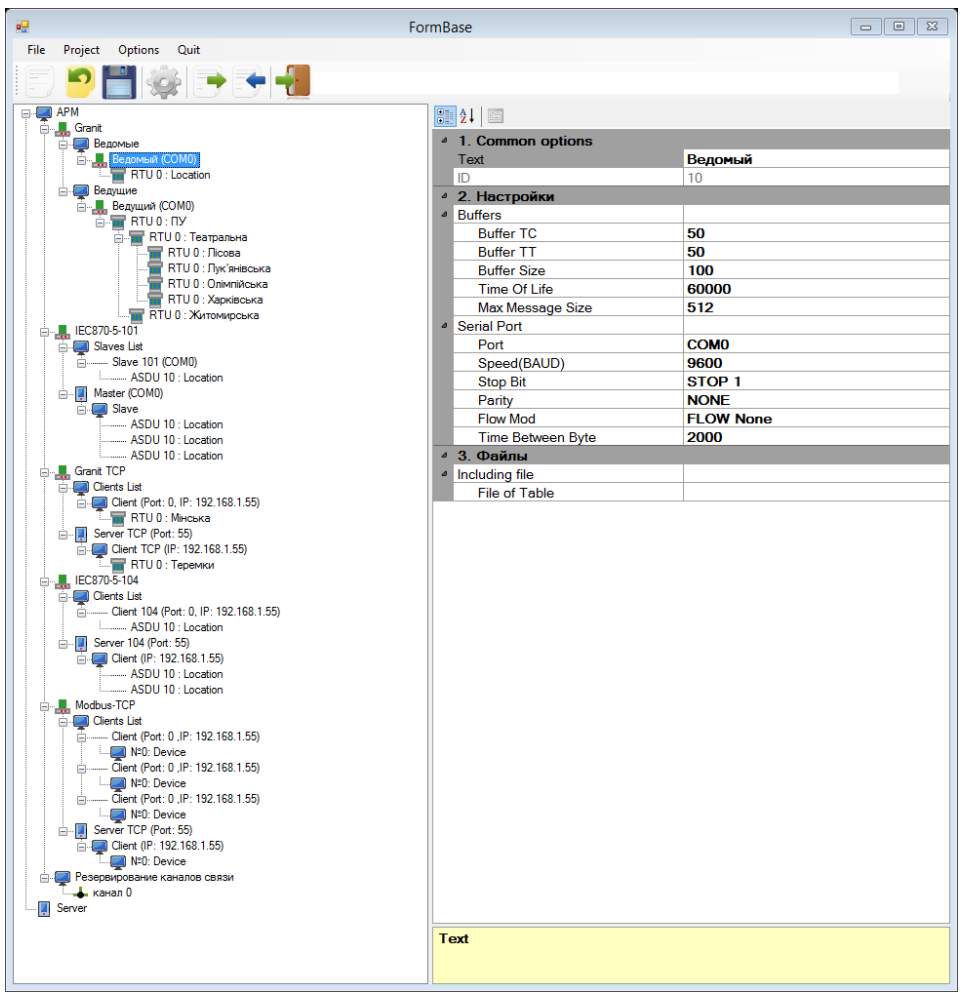
Локалізація українською мовою



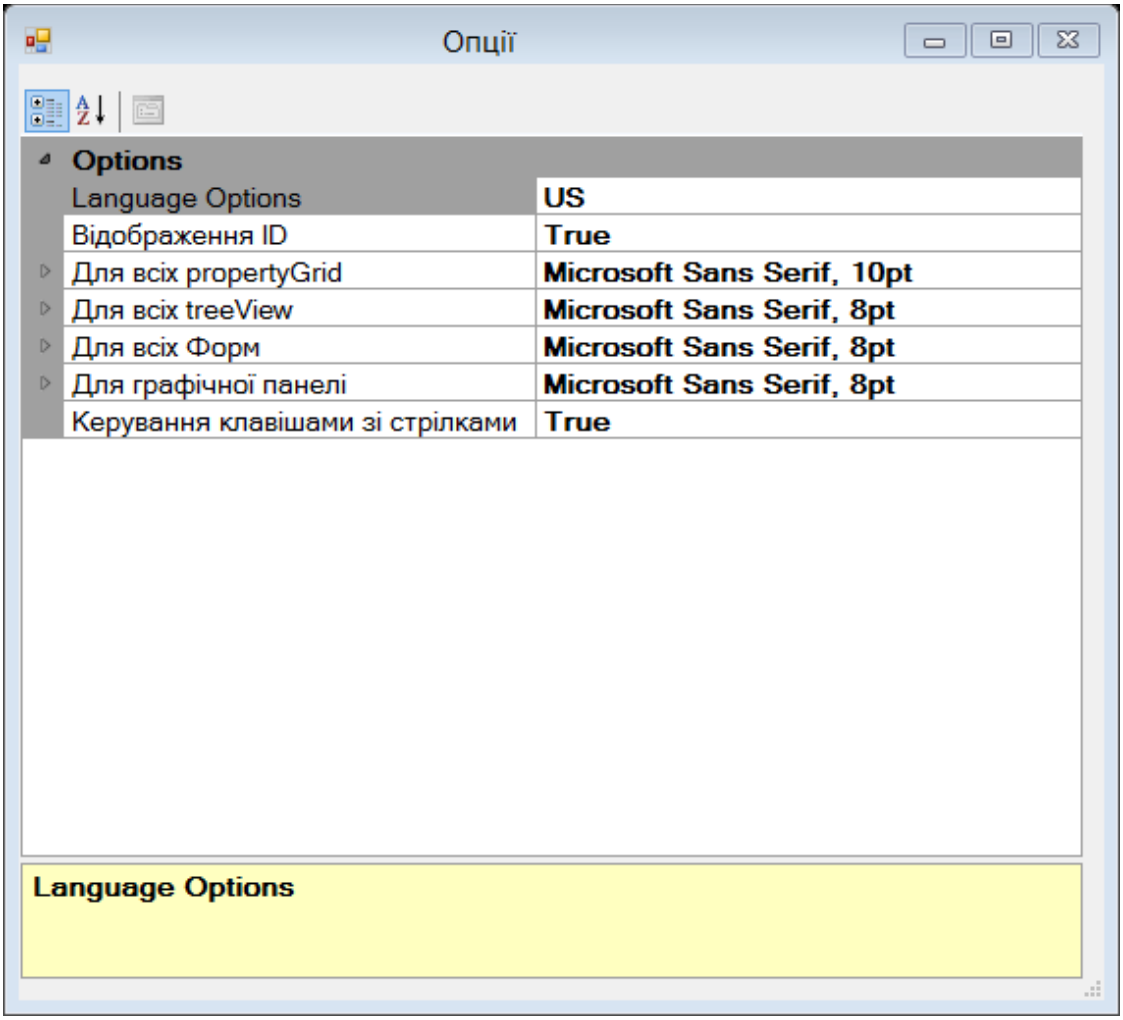
Локалізація російською мовою



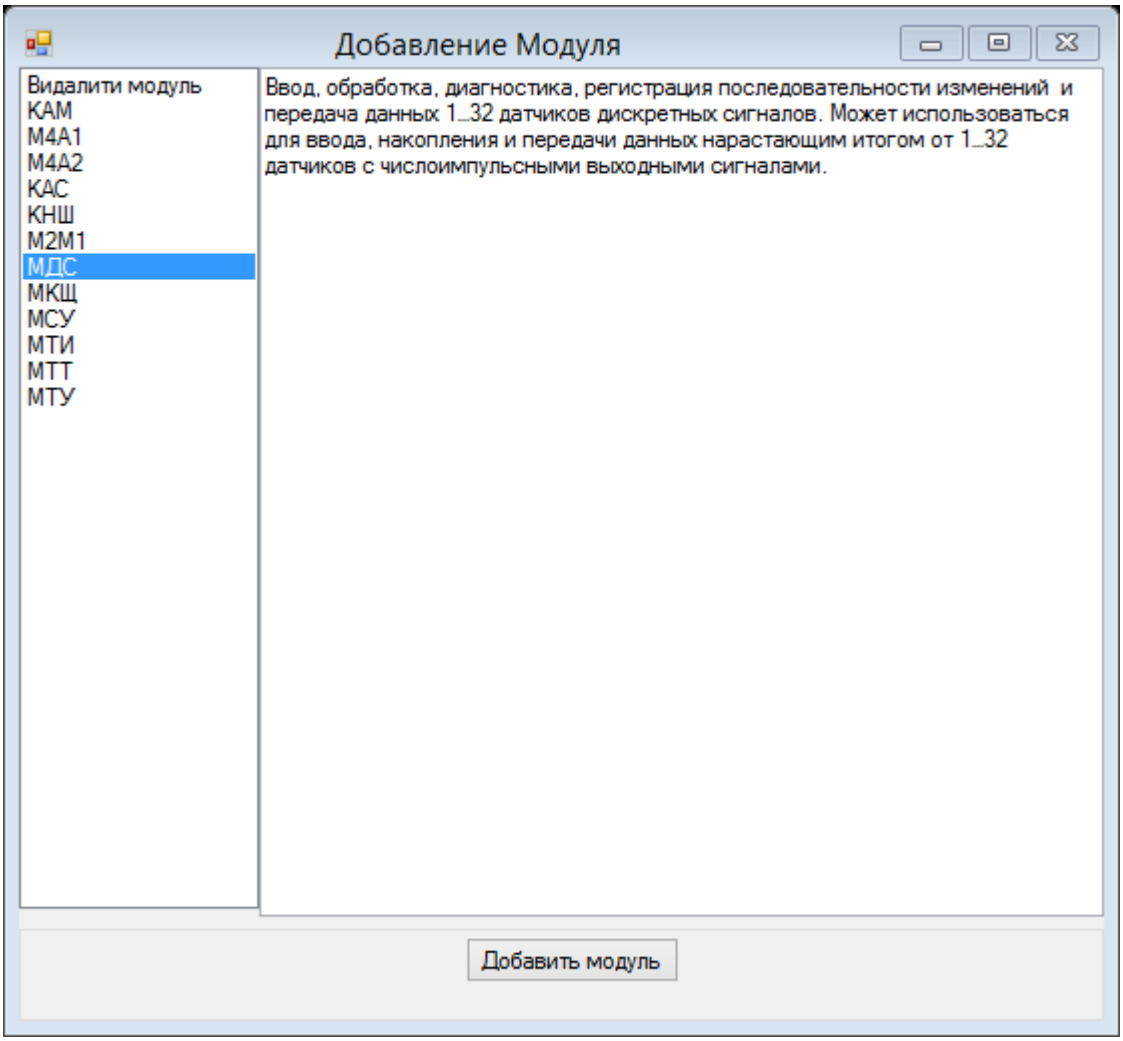
Локалізація англійською мовою



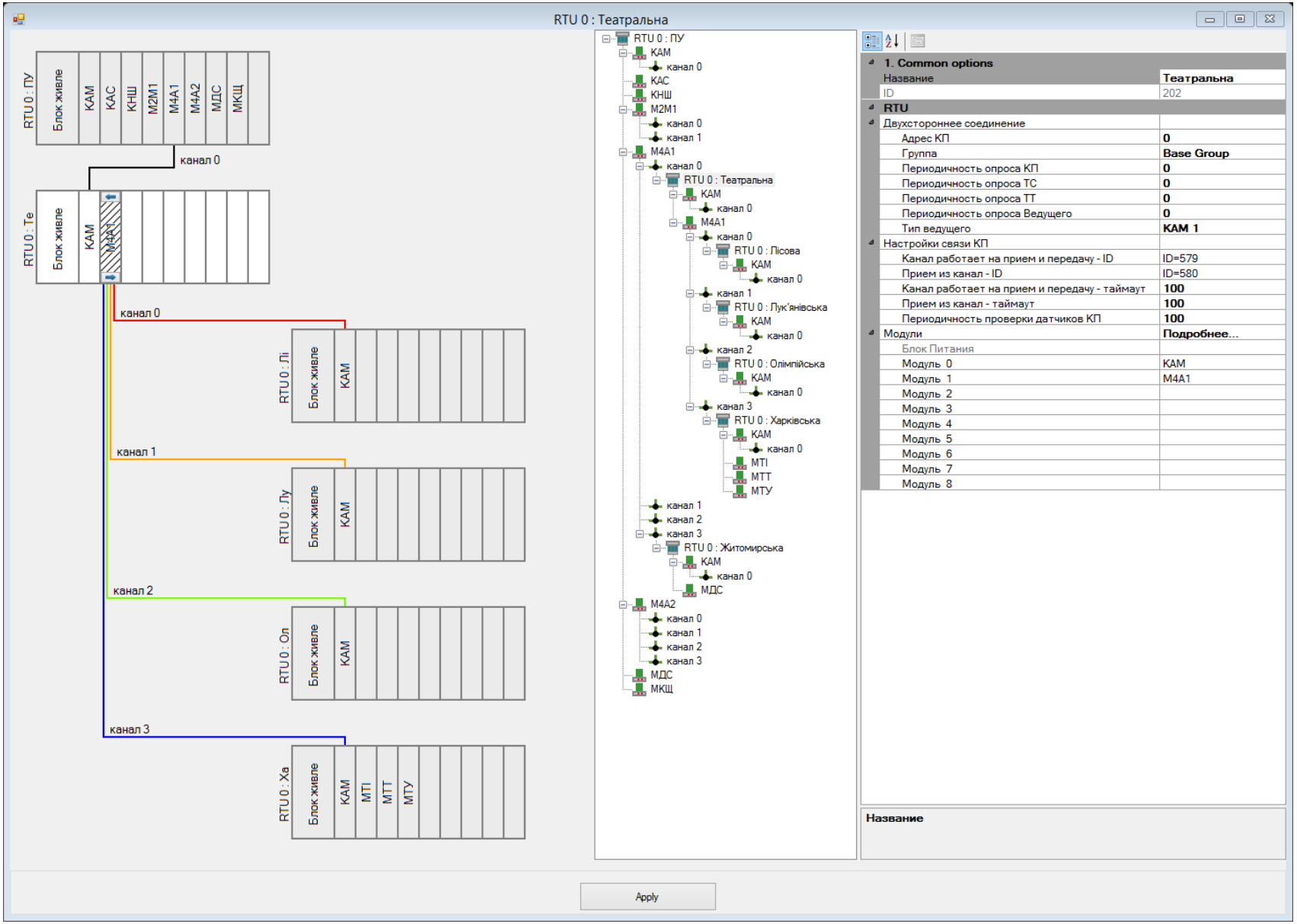
Форма «Options» – дозволяє налаштувати параметри Конфігуратора



Форма Form_AddingModule –для додавання модулів у КП (RTU)

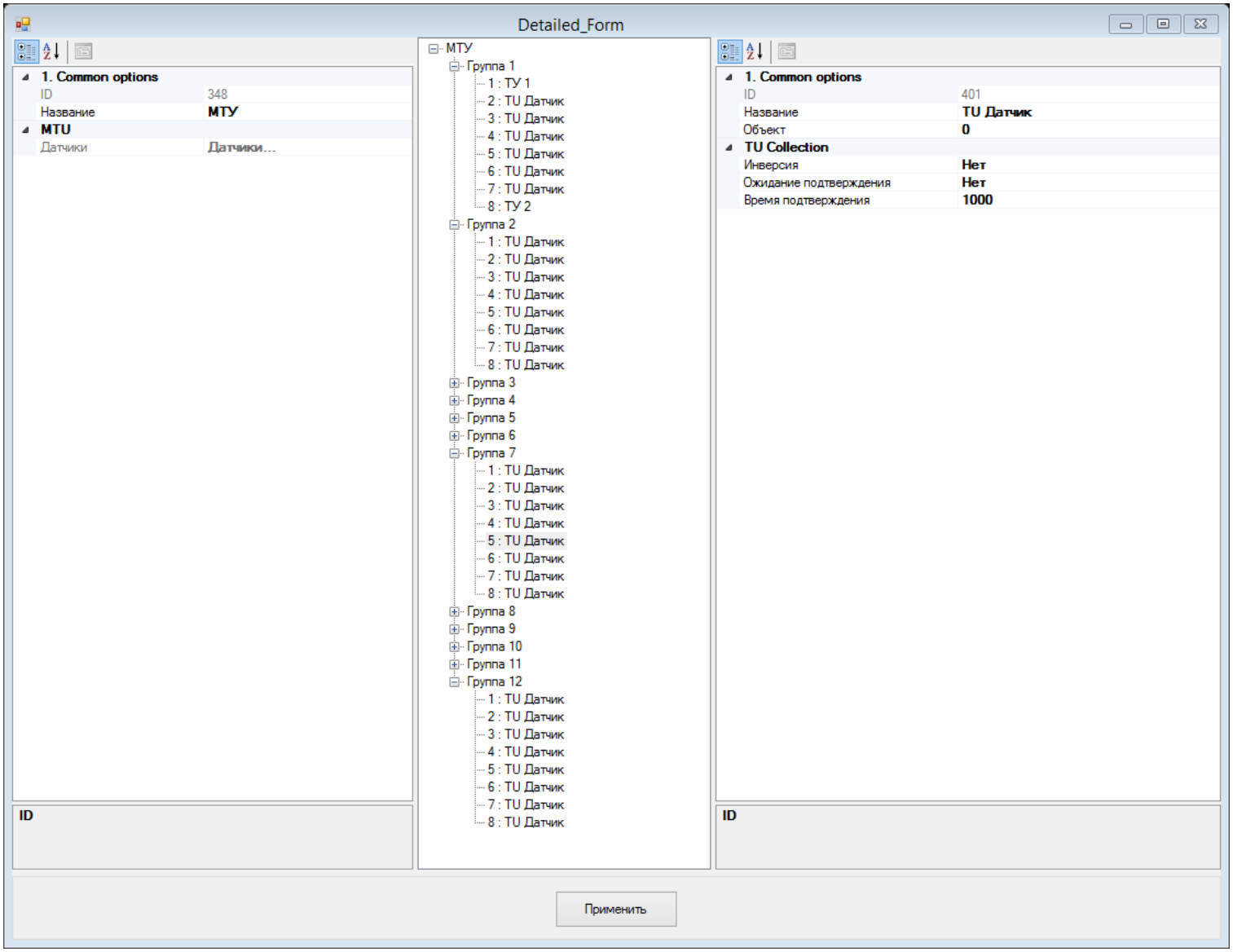


Форма FormModules — призначена для розгорнутого налаштування модулів RTU і каналів зв’язку

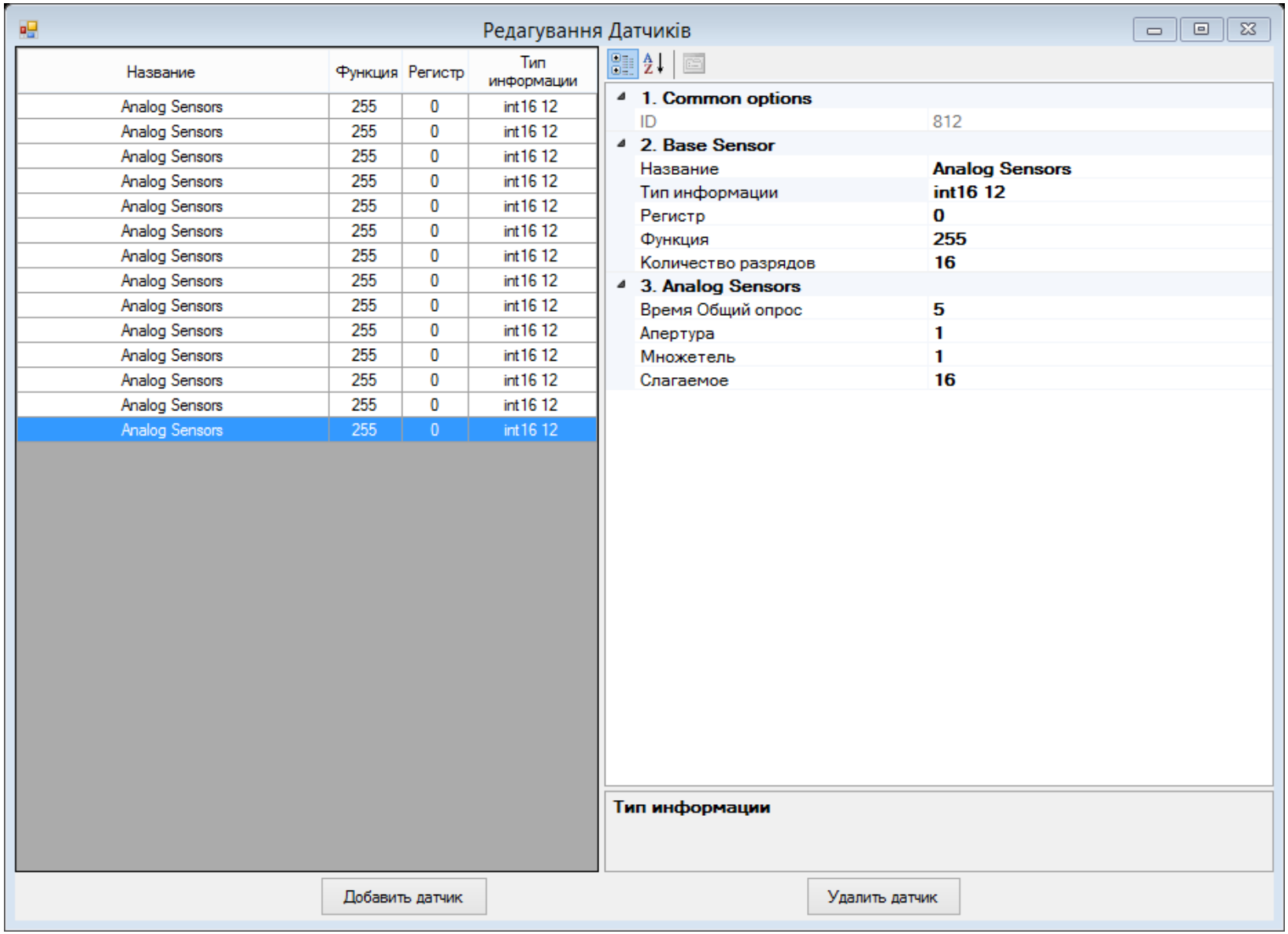


Додаток 6

Форма Detail_of_Objects – призначена для детального опису модулів і спеціалізованих об’єктів

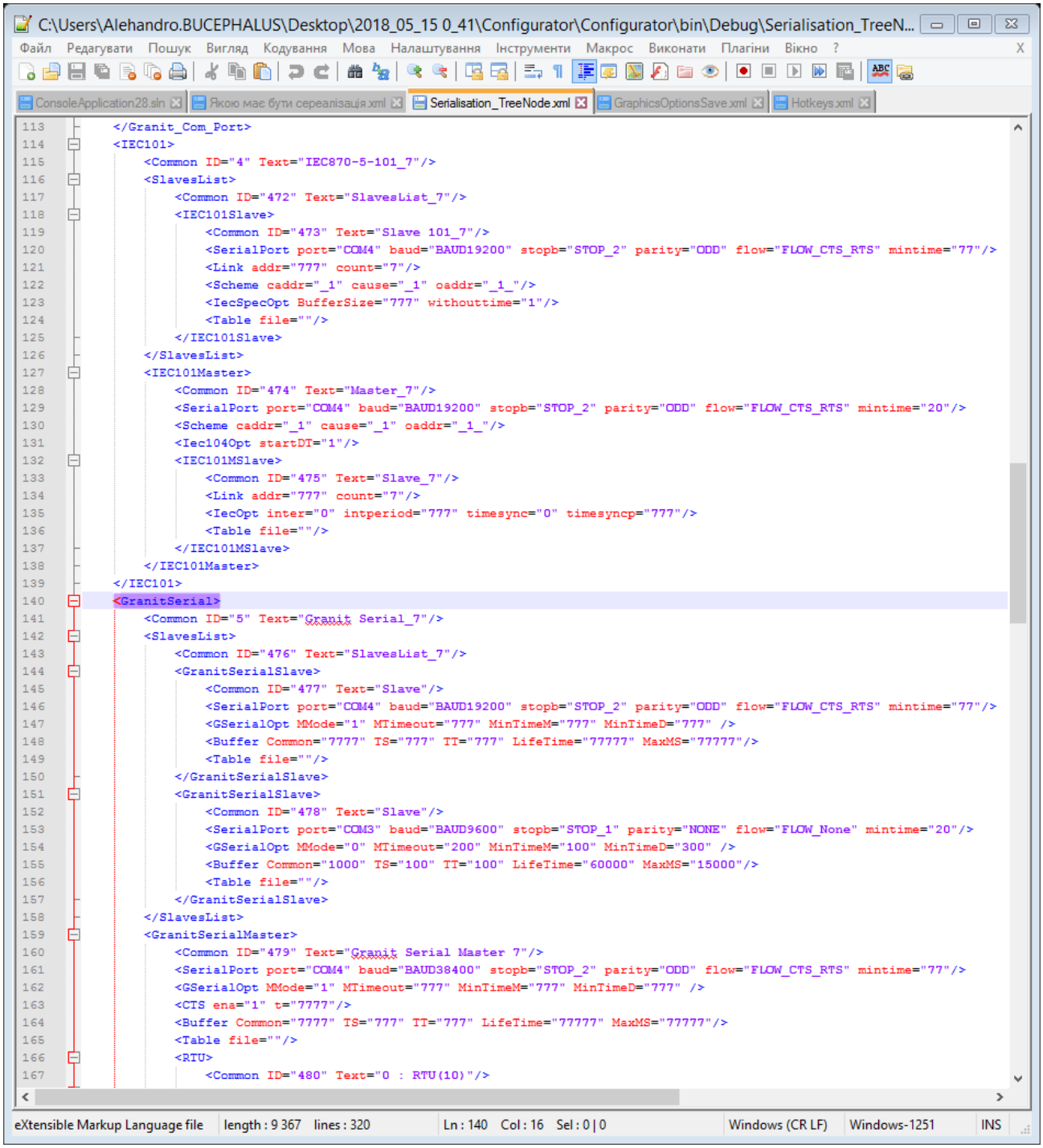


Додаток 7

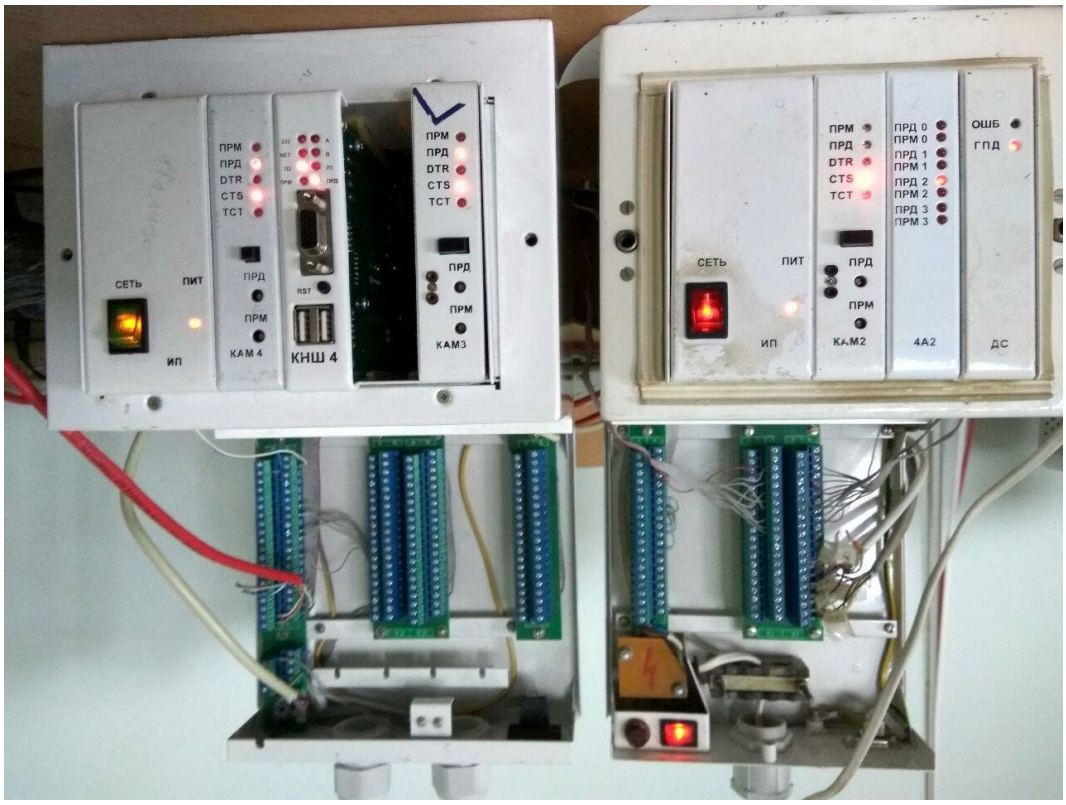

Редагування Датчиків
Название
Функция
Регистр
Тип информации
Analog Sensors 255 0 int16 12
1. Common options
ID 812
2. Base Sensor
Название Analog Sensors
Тип информации int16 12
Регистр 0
Функция 255
Количество разрядов 16
3. Analog Sensors
Время Общий опрос 5
Апертура 1
Множетель 1
Слагаемое 16
Тип информации
Добавить датчик
Удалить датчик

Додаток 8

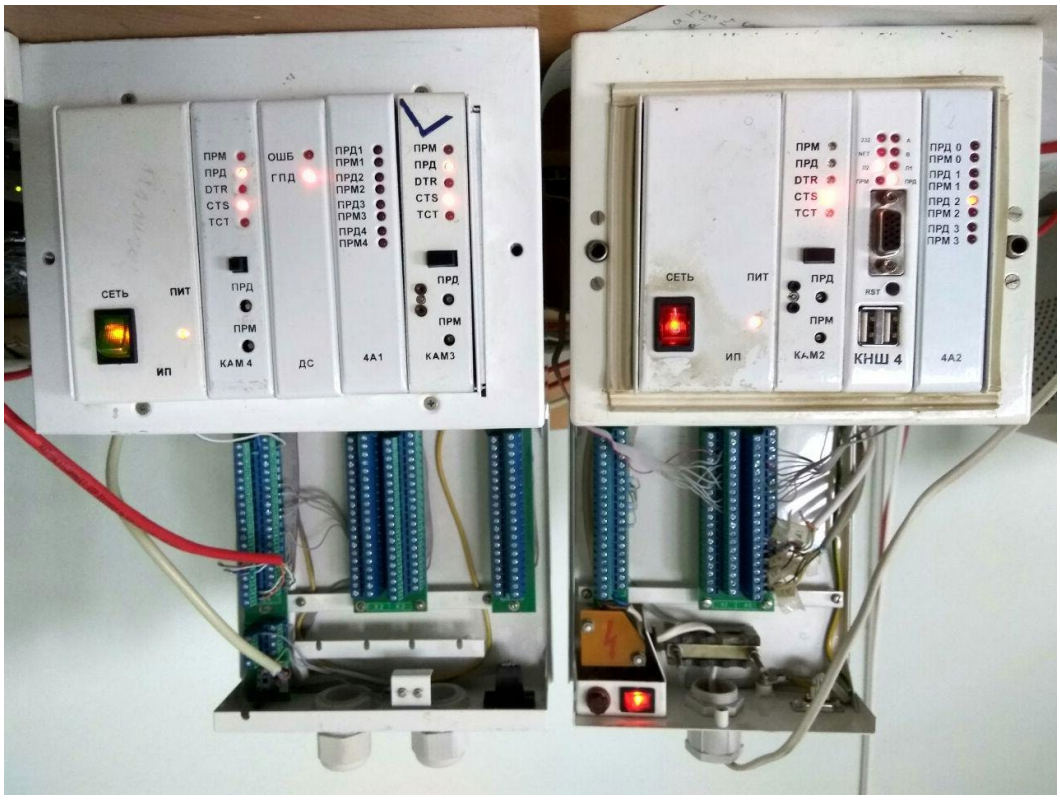
Приклад серіалізації



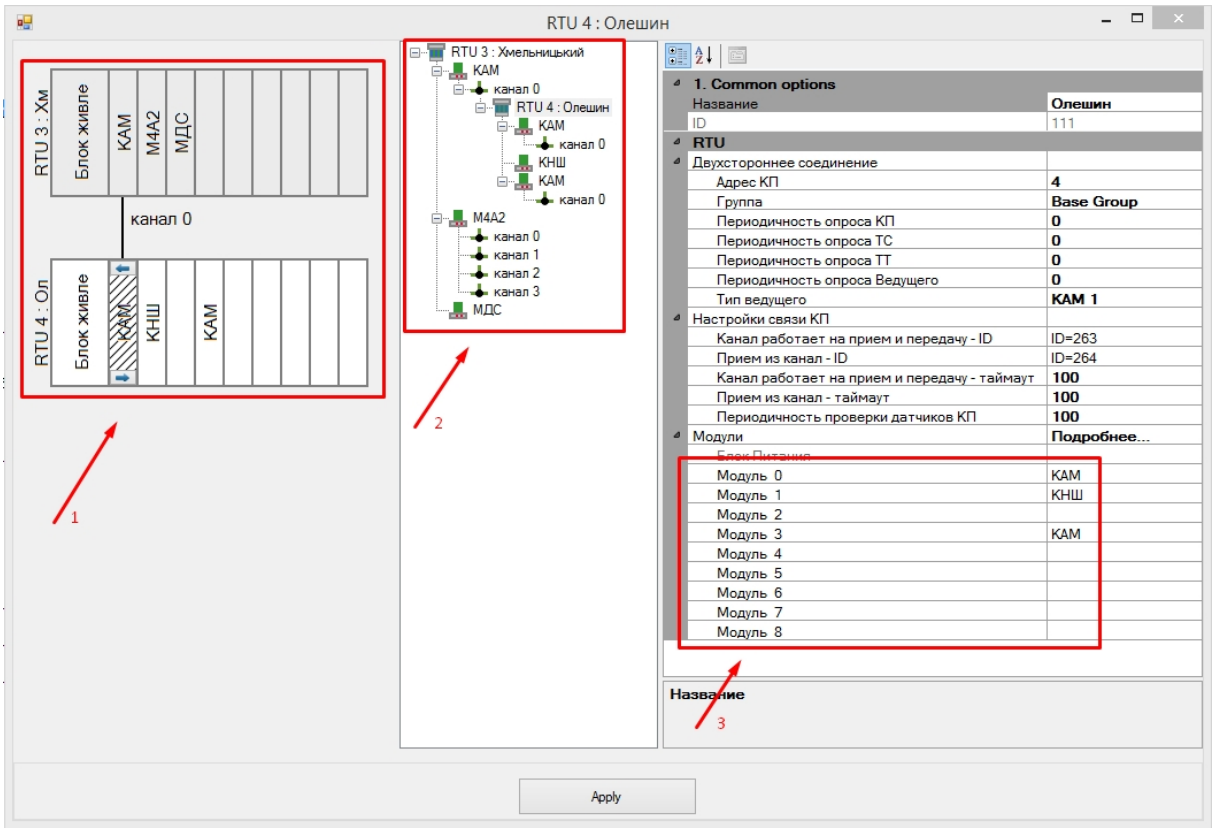
Експериментальна перевірка розробленого конфігуратора (перше випробування)



Експериментальна перевірка розробленого конфігуратора (друге випробування)



Результати першого випробування



Результати другого випробування