Міністерство освіти і науки України
Криворізький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерних систем та мереж

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА
за спеціальністю 123 «Комп'ютерна інженерія»

Тема наукової роботи: МОДЕЛЬ ЗБОРУ ДАНИХ В РЕАЛЬНОМУ ЧАСІ ЗА ДОПОМОГОЮ ЗАСОБІВ AVEVA SYSTEM PLATFORM

Виконав _____ О. О. Романенко

Керівник роботи _____ С. В. Сьомочкина

Нормоконтроль _____ Д. І. Кузнєцов

Завідувач кафедри _____ А. І. Купін

Кривий Ріг
2023

Криворізький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерних систем та мереж

Ступінь вищої освіти магістр
Спеціальність 123 «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри, голова циклової комісії

_____ А. І. Купін

"____" _____ 20__ року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ МАГІСТРА

_____
(прізвище, ім'я, по батькові)

1. Тема роботи _____
_____
_____

керівник роботи _____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від "___"_____ 20__року №__

2. Строк подання студентом роботи _____

3. Вихідні дані до роботи _____
_____
_____
_____
_____
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)_____
_____
_____
_____
_____
_____

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
_____
_____
_____
_____

6. Дата видачі завдання_____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка

Студент _____ _____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка: 100 сторінок, 128 рисунків, 60 використаних джерел.

Об’єкт дослідження — система збору даних, побудована засобами промислової програмної платформи Aveva System Platform.

Кваліфікаційна робота складається з 4 розділів.

Перший розділ присвячений аналізу основних понять про систем збору даних, їх функцій та компоненти, розглядаються процеси, що протікають в таких системах.

У другому розділі проводиться огляд програмного забезпечення Aveva System Platform. Розглядається технологія ArchestrA.

У третьому розділі здійснюється розробка джерела даних для розроблюваної системи.

В четвертому розділі здійснюється створення та конфігурація програмної частини системи. Створено додаток Galaxy.

AVEVA SYSTEM PLATFORM, ARCHESTRA, SCADA, ARDUINO, СИСТЕМА ЗБОРУ ДАНИХ, ВИМІРЮВАННЯ

Master’s work: 100 pages, 128 figures, 60 used sources.

Object of research — a data collection system built using the Aveva System Platform industrial software platform.

The paper consists of 4 sections.

The first chapter is devoted to the analysis of the basic concepts of data collection systems, their functions and components, the processes taking place in such systems are considered.

The second section provides an overview of the Aveva System Platform software. ArchestrA technology is considered.

In the third section, the data source for the developed system is developed.

In the fourth section, the creation and configuration of the software part of the system is carried out. Galaxy app created.

AVEVA SYSTEM PLATFORM, ARCHESTRA, SCADA, ARDUINO, DATA COLLECTION SYSTEM, MEASUREMENT

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

АЦП – аналого-цифровий перетворювач
ПЗ – програмне забезпечення
ПК – персональний комп’ютер
ПЛК – програмований логічний контролер
ЛОР – леткі органічні речовини
МК – мікроконтролер
СЗД – система збору даних
ЦАП – цифро-аналоговий перетворювач
DAQ – Data acquisition (збір даних)
MES – Manufacturing Execution System
SCADA – Supervisory Control And Data Acquisition

ВСТУП

Автоматизація є одним з основних рушіїв науково-технічного прогресу. Інтенсивне застосування мікропроцесорної техніки, персональних комп'ютерів та обчислювальних мереж дозволяють впроваджувати комп'ютерні технології в автоматизацію та створювати комп'ютерно-інтегровані системи керування, що є основним напрямком розвитку автоматизації на сучасному етапі.

Автоматизація дозволяє значно підвищити продуктивність праці, забезпечує економію матеріальних та енергетичних ресурсів, підвищення якості та зниження собівартість продукції.

Область застосування автоматизації включає такі сфери діяльності як: виробничі процеси, розробка програмного забезпечення, проєктування, управління бізнесом, логістика, та інші сфери людської діяльності.

Автоматизація виробничих процесів є однією із ключових галузей застосування та розвитку технологій автоматизації. Одним із напрямів автоматизації виробництв є автоматизація керування технологічними процесами. Впровадження автоматизованих систем керування технологічними процесами (АСК ТП) дозволяє створювати високопродуктивні технологічні процеси, оптимізувати процес виробництва, зменшити витрати та збільшити прибутки.

Одним із ключових компонентів автоматизованих систем керування є системи збору даних.

Системи збору даних — це апаратно-програмні системи. До апаратної частини входить вимірювальне обладнання, наприклад різноманітні датчики, перетворювачі сигналів (аналогово-цифрові та цифро-аналогові перетворювачі), кабельна система, контролери. До програмної частини відноситься спеціалізоване програмне забезпечення для збору даних.

Актуальність роботи. Необхідність розробки та впровадження систем збору даних обумовлена значними потоками даних, які генеруються в процесі експлуатації об'єкта виробництва (дослідження).

Збір даних відіграє значну роль в процесах, де потрібно вживати заходів на основі вимірювань. Системи збору даних застосовуються в наукових дослідженнях, аерокосмічній сфері, промисловості тощо. Зібрані дані використовуються для проведення досліджень, контролю за виробничими процесами, формування стратегій розвитку, оптимізації процесів.

Мета і завдання дослідження. Метою кваліфікаційної роботи є створення системи збору даних засобами Aveva System Platform.

Для досягнення поставленої мети необхідно виконати такі завдання:

- проведення аналізу систем збору даних, визначення їх основних складових та функцій, які виконують такі системи;
- створення системи збору даних, яка відповідає вимогам до систем реального часу.

Об’єкт дослідження. Об’єктом дослідження є система збору даних, побудована засобами Aveva System Platform.

Предмет дослідження — процес збору, передачі та обробки даних в режимі реального часу.

Методи досліджень. В ході проведення дослідження було використано метод аналізу, який полягає у розкладанні об’єкта на складові та вивченні кожної складової окремо у межах єдиного цілого.

Наукова новизна одержаних результатів. Було проведено дослідження такого класу інформаційних систем як системи збору даних. Виконано систематизацію теоретичної інформації про даний клас систем, досліджено процеси перетворення даних в ході роботи таких систем.

Практичне значення отриманих результатів. Розроблено проект (Galaxy) системи збору даних від мікроконтролера засобами платформи Aveva System Platform.

Апробація роботи. Апробація роботи відбувалась на конференції КІСМ 2023 від Криворізького національного університету.

1 СИСТЕМИ ЗБОРУ ДАНИХ ЯК КЛАС ІНФОРМАЦІЙНИХ СИСТЕМ

1.1 Загальні поняття про системи збору даних

В даній роботі системи збору даних будуть розглядатись здебільшого в контексті автоматизації виробництва та технологічних процесів.

Будь яке виробництво складається із ланцюга технологічних процесів, які необхідно контролювати. Контролю також підлягає і технічне устаткування, яке застосовується в ході виробництва. Суть контролю полягає у порівнянні поточного стану об’єкта, який характеризується певними фізичними величинами, із нормою (певними гранично допустимими значеннями), з метою встановлення поточної якісної характеристики об’єкта [1, 2]. Відповідно, для здійснення контролю необхідно здійснити вимірювання фізичних величин, якими характеризується об’єкт. В свою чергу, вимірювання — це процес отримання кількісної інформації про об’єкт [3].

Інтенсифікація автоматизації процесів виробництва, все більше ускладнення технологічних процесів та необхідність здійснювати контроль значної кількості параметрів призвели до того, що кількість даних, вимірювання та обробку яких необхідно здійснювати для забезпечення працездатності виробництва, зросла в рази. В зв’язку з цим, постало питання створення засобів, які б здійснювали автоматичний збір та обробку значних інтенсивних потоків даних. Такими засобами стали системи збору даних.

Система збору даних (Data acquisition system) — сукупність функціонально об’єднаних вимірювальних, обчислювальних та інших допоміжних пристроїв, а також програмного забезпечення, призначених для автоматичного отримання (збору) вимірювальної інформації безпосередньо від об’єкта дослідження, її обробки з метою подальшого зберігання та представлення користувачу в певному вигляді [4, 5].

Варто зазначити, що системи збору даних часто є складовою конкретних приладів, установок та інших систем, а не конструктивно відокремленою одиницею, але можуть застосовуватись і автономно. Тобто, на практиці такі системи у явному вигляді не виділяються. Яскравим прикладом системи, складовою якої є система збору даних є SCADA (Supervisory Control And Data Acquisition — диспетчерське управління і збір даних), де явно вказано наявність системи збору даних як складової.

Збір даних (Data acquisition, DAQ), у широкому розумінні, це процес отримання та накопичення відомостей про ті чи інші об’єкти, явища, події тощо [6, 7].

В контексті інженерно-наукової діяльності збір даних — це процес, отримання значень вимірюваних параметрів досліджуваних систем з метою їх подальшої обробки [8].

Виходячи із визначення, даного вище, стає зрозуміло, що системи збору даних містять три основних компоненти — джерело даних, апаратне забезпечення та програмне забезпечення. На рисунку 1.1 наведено найбільш загальну структурну схему системи збору даних.

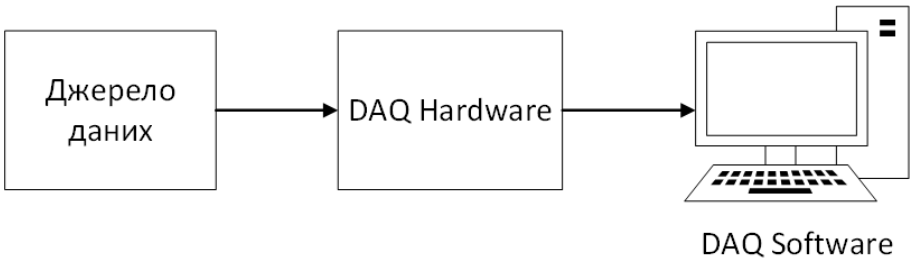

Рисунок 1.1 —Загальна структурна схема системи збору даних

Виділення класу систем збору даних на множині інформаційних систем можливе через виділення завдань, які виконують такі системи:

- отримання первинної вимірювальної інформації безпосередньо від джерела (об’єкта дослідження), тобто здійснення вимірювань;
- забезпечення вимірювань із визначеною точністю;
- робота в режимі реального часу;
- передача отриманих від джерела даних до місця обробки та використання даних;
- накопичення даних;
- візуалізація даних.

Беручи до уваги наведені вище задачі, що має виконувати СЗД, можна розробити більш детальну структурну схему (рисунок 1.2).

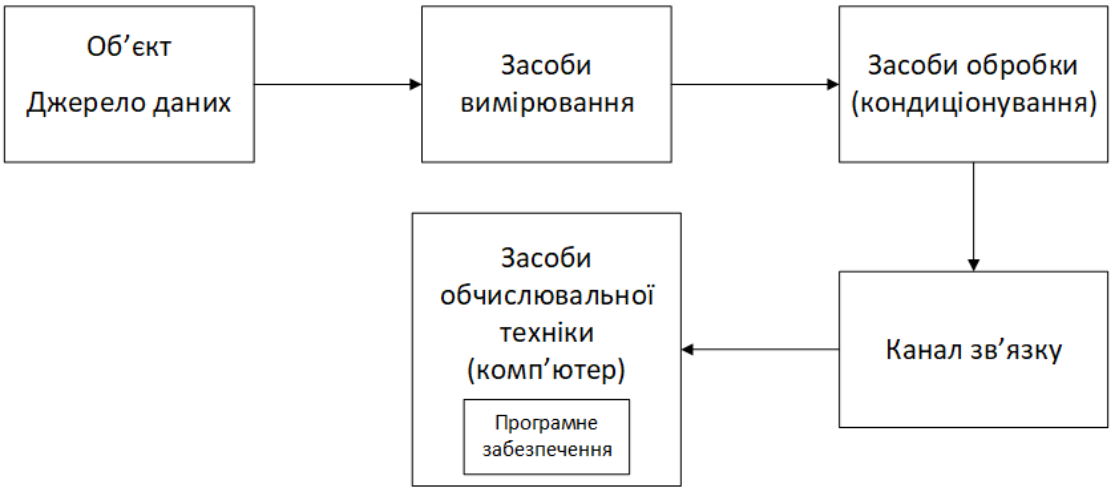

Рисунок 1.2 — Структурна схема системи збору даних

На рисунку 1.3 наведено схему, яка відображає функції виконувані системами збору даних, як процеси, що послідовно здійснюються в ході функціонування системи.

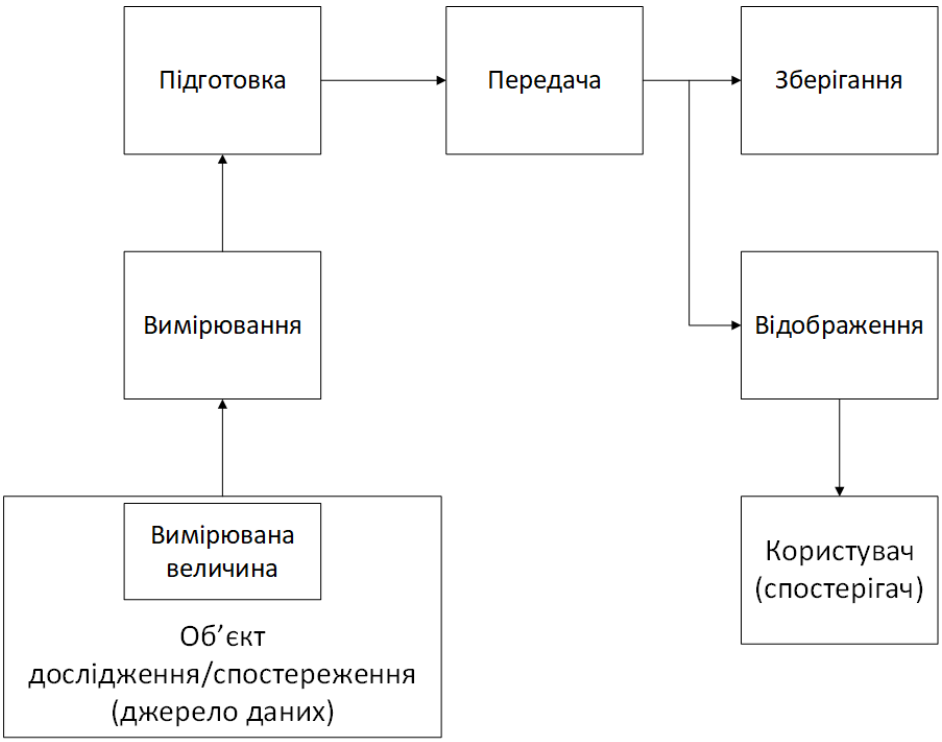

Рисунок 1.3 — Процеси у системі збору даних

Для більш детального відображення взаємозв'язку функцій, наведених на рисунку, розроблено функціональну діаграму в нотації IDEF0 (рисунок 1.4). Дана діаграма показує функції (процеси), дані, пов'язані з функціями, правила, які визначають як застосовуються функції.

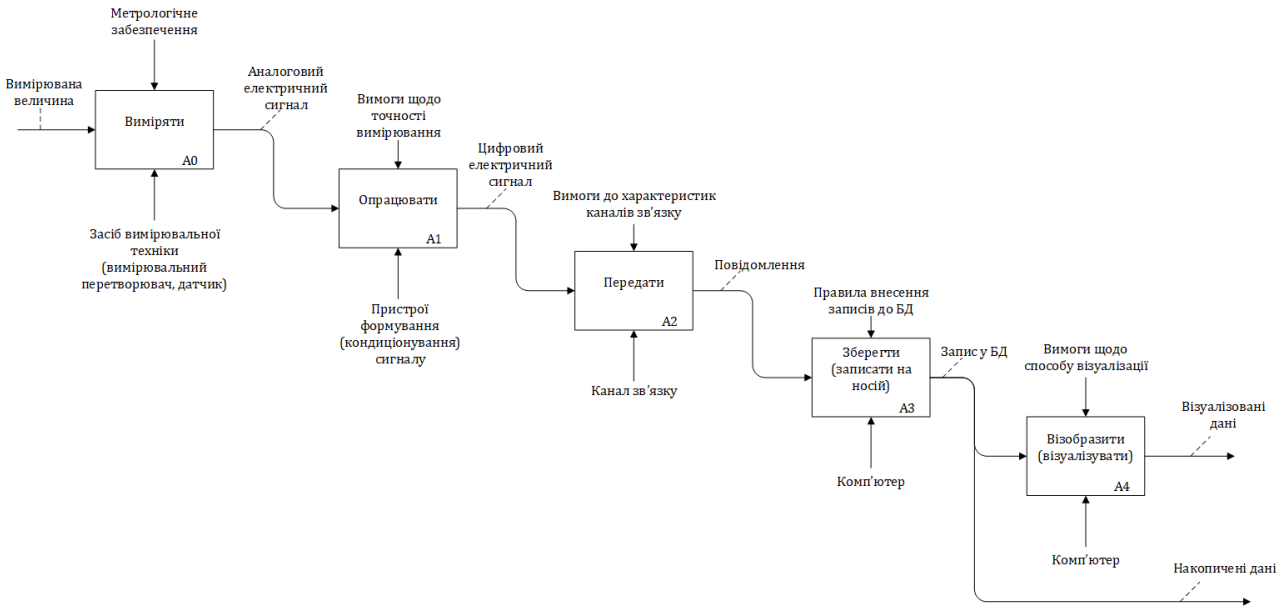

Рисунок 1.4 — IDEF0-модель для системи збору даних

Системи збору даних це програмно-апаратні системи. До апаратного забезпечення систем збору даних відносяться: [9, 10]:

- засоби вимірювання (такі як датчики);
- цифро-аналогові перетворювачі;
- аналогово-цифрові перетворювачі;
- підсилювачі;
- частотні фільтри;
- мультиплексори;
- демультиплексори;
- схеми кондиціонування сигналів;
- мережеві адаптери;
- кабельна система;
- контролери та інша комп’ютерна техніка.

Програмне забезпечення системи збору даних є важливою її складовою. Від програмного забезпечення залежить ефективність використання апаратного забезпечення.

Програмне забезпечення збору даних може бути як програмою для мікроконтролера до якого підключено декілька датчиків, так і масштабною програмною системою (SCADA), яка використовується у виробничих умовах для збору даних із сотень або тисяч джерел.

1.2 Процес збору даних

Процес збору даних починається із вимірювання досліджуваних параметрів певної системи. Без отримання вимірювальної інформації, тобто кількісних даних про стан об’єкта дослідження, неможливе здійснення технологічних процесів у виробництві та будь якої іншої інженерної-наукової діяльності.

Так як реальним носієм вимірювальної інформації є сигнал, то процеси, збору даних, які показані на рисунку 1.3, це процеси перетворення вимірювального сигналу. На рисунку 1.5 зображено, як змінюється сигнал під час проходження вимірювальним каналом.

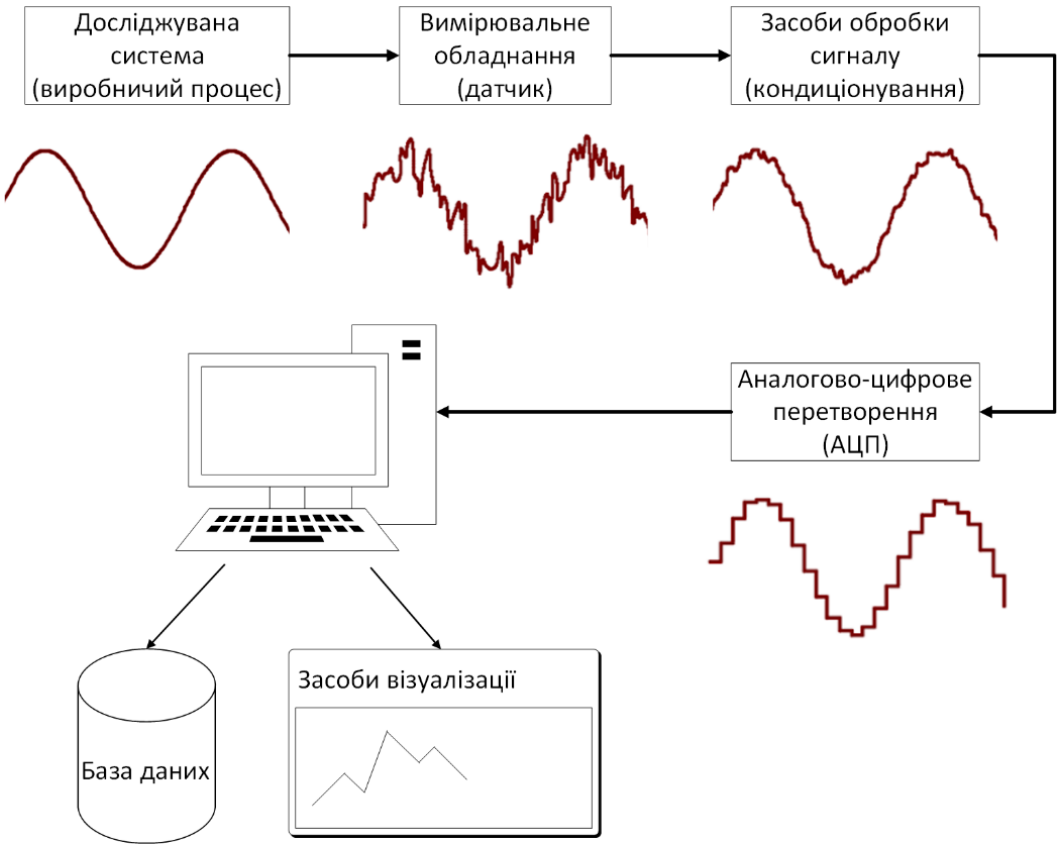

Рисунок 1.5 — Процес перетворення вимірювального сигналу

Вимірювання — це процес отримання кількісної інформації про об’єкт [3]. Об’єктом вимірювання можуть виступати явища, тіла і речовини різної природи. Як об’єкт вимірювання можна розглядати виробничу систему або виробничий процес, що характеризуються певної кількістю параметрів, які описуються фізичними величинами, такими як наприклад температура, тиск, швидкість, рівень концентрації речовини, різноманітні електричні величини (сила струму, напруга).

Вимірювання здійснюється за допомогою спеціальних технічних засобів вимірювання. Процес вимірювання здійснюється, як правило, через перетворення деякої величини, значення якої необхідно встановити, у величину іншої природи, яку можна зручно виміряти, обробити та передати. Наразі, найбільшого поширення дістали електричні засоби вимірювання, які перетворюють неелектричні вимірювані величини в електричні [3].

Використання електровимірювальної апаратури пояснюється можливістю здійснення неперервних в часі вимірювань, здійснення дистанційного вимірювання шляхом передачі виміряного сигналу системами зв’язку, високою точністю вимірювання, широким діапазоном вимірювання [3].

Вимірювання фізичних величин електричними вимірювальними засобами уможливлюється після попереднього перетворення досліджуваних фізичних величин на функціонально зв’язані з ними електричні величини за допомогою відповідних вимірювальних перетворювачів. Поряд з терміном «вимірювальний перетворювач» в технічній літературі часто зустрічається

інший термін — «датчик», що має однакове з першим значення. Датчики є найбільш розповсюдженим і часто використовуваним засобом вимірювання в системах збору даних [11].

Наступна після вимірювання стадія — це обробка, отриманого в результаті вимірювання, сигналу або ж кондиціонування (англ. conditioning).

Сигнал, який генерується чутливим елементом датчика (сенсором), зазвичай має малу амплітуду, є зашумленим або ж містить певні небажані компоненти. Окрім цього, вихідний сигнал сенсора може бути несумісним із вхідними параметрами обладнанням системи збору даних. З метою узгодження функціональних параметрів датчика та обладнання збору даних здійснюється кондиціонування сигналу, згенерованого сенсором [12].

Кондиціонування — процес приведення сигналу, отриманого від вимірювального пристрою, у вигляд придатний для подальшої обробки іншими пристроями [9, 13].

Метою кондиціонування є приведення первинного сигналу до вигляду, який є сумісним із наступним пристроєм обробки. Часто, таким пристроєм є аналогово-цифровий перетворювач [12].

Поняття кондиціонування є загальним та описує процес попередньої обробки сигналу. Кондиціонування охоплює такі маніпуляції над сигналами: [14]

- підсилення;
- фільтрація;
- лінеаризація;
- гальванічна розв’язка.

Сенсори генерують сигнали, які мають амплітуду порядку мілівольт, а для подальшої обробки такими пристроями як аналогово-цифрові перетворювачі, модулятори, мікроконтролери тощо, амплітуда сигналу повинна бути порядку вольт, отже для того, щоб уможливити подальшу обробку сигналу, тобто виміряної величини, необхідно здійснити підсилення цього сигналу [12].

Підсилення сигналу здійснюється з метою збільшення розрядності вимірюваного сигналу. Якщо низько амплітудний сигнал від сенсора подати, наприклад, на вхід дванадцяти розрядного аналогово - цифрового перетворювача із діапазоном вхідних значень від нуля до десяти вольт, то втрати в точності будуть катастрофічним. Такий АЦП має розрядність за напругою 2,44 мВ, тобто, найменше значення напруги, яке може розрізнити такий АЦП, становить 2,44 мВ. Підвищення рівня точності перетворення може бути досягнуте шляхом підсилення вхідного, по відношення до АЦП сигналу, таким чином, щоб максимальне коливання вхідної напруги відповідало максимальному вхідному діапазону АЦП [9].

Інша важлива задача підсилення полягає у збільшенні відношення сигнал - шум. Відношення сигнал – шум це відношення амплітуди корисного сигналу до амплітуди шуму [15]. Посилення сигналу низького рівня перед передачею зменшує вплив шумів на цей сигнал. Збільшення відношення

сигнал-шум сприяє покращенню характеристик системи в контексті точності перетворення виміряних сигналів.

Функція гальванічної розв'язки полягає в забезпеченні відсутності впливу петель заземлення або синфазних напруг на точність вимірюваних сигналів. Петлі заземлення та струми що в них протікають, спричинені різницею потенціалів між джерелом заземлення та опорним заземленням вимірювального пристрою, можуть стати причиною викривлення виміряного сигналу або, якщо ці струми занадто великі, вони можуть пошкодити обладнання [16].

Процес фільтрації (рисунок 1.6) полягає у повному або частковому усуненні небажаної компоненти із сигналу, що означає видалення певних небажаних частот [17]. Фільтрація усуває небажаний шум із вимірюваного сигналу.

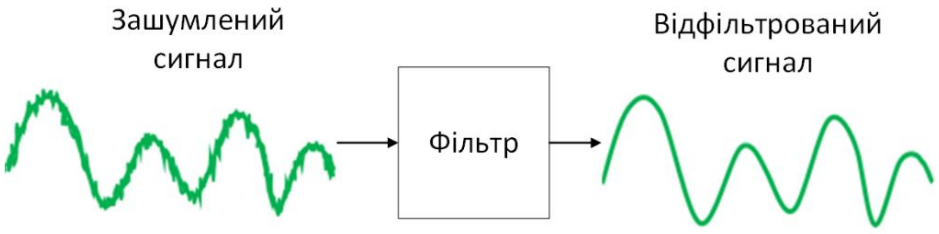

Рисунок 1.6 — Фільтрація сигналу

В ідеальному випадку, фільтри мають усувати всі дані на частотах за межами зазначеного діапазону частот, забезпечуючи дуже різкий перехід між частотами, які пропускаються, і частотами, які відфільтровуються. Але в реальності більшість фільтрів не є ідеальними і зазвичай не усувають усі небажані амплітудні компоненти за межами заданого діапазону частот [4].

Вихідні сигнали деяких типів датчиків не пов'язані лінійно з фізичною величиною, яку вони вимірюють. Лінеаризація – це процес встановлення лінійної залежності сигналу від датчика та значенням фізичної величини. Лінеаризація може бути здійснена за допомогою апаратних засобів обробки сигналу або за допомогою програмного забезпечення. Класичним прикладом датчиків, які потребують лінеаризації є термопари [18].

Для спряження цифрової системи (комп'ютера), яка застосовується для обробки та аналізу виміряних даних, із аналоговими сигналами, які генеруються сенсорами, необхідний деякий пристрій, який би виконував перетворення аналогового сигналу у цифровий. Такий пристрій називається аналого-цифровим перетворювачем або скорочено АЦП.

АЦП бувають різних типів, але всі вони призначені для виконання єдиної задачі — перетворення аналогового вхідного сигналу у вихідний цифровий код.

Далі отриманий цифровий сигнал необхідно передати до місця подальшої обробки та використання. Для передачі застосовуються системи зв'язку.

Системою зв’язку називається комбінація технічних засобів та середовища розповсюдження сигналу, що забезпечують передачу повідомлень від джерела до отримувача [19].

Процес передавання інформації полягає в тому, що повідомлення від джерела перетворюється на сигнал, який передається отримувачу. Очевидно, що на стороні отримувача повинно здійснюватись зворотнє перетворення сигналу в повідомлення.

Отже, в будь-якій системі зв’язку повинні бути наявні елементи, які виконують функції перетворення повідомлення в сигнал (на стороні передавача) та перетворення сигналу у повідомлення (на стороні отримувача) [19].

Рисунок 1.7 — Узагальнена структурна схема системи зв’язку

Очевидно, що для передавання сигналу від джерела до отримувача між ними повинно існувати деяке фізичне середовище, по якому б розповсюджувався сигнал. Таке середовище називається лінією зв’язку [19, 20]. Фізичним середовищем може бути металевий провідник або кабель (в дротовому зв’язку), атмосфера (в радіозв’язку), скляні світловоди (в оптичних системах зв’язку) тощо [21].

Каналом зв’язку називається поєднання фізичного середовища та сукупності технічних засобів, які забезпечують передавання сигналів від джерела до отримувача. [19, 20, 22, 23].

Як правило, повідомлення, що генеруються джерелом, не пристосовані для їх безпосереднього передавання через канал зв’язку без певних перетворень. Тому вони піддаються кодуванню і, дуже часто, неодноразово. Кодування джерела ще називають статистичним або ж економним кодуванням. Його мета — підвищення ефективності передачі повідомлень. Економне кодування полягає в зменшенні надлишковості [24].

В ході кодування повідомлення перетворюється на послідовність кодових символів [25].

Кожному елементу повідомлення привласнюється певна сукупність кодових символів, яка називається кодовою комбінацією. Сукупність кодових комбінацій, що позначають дискретні повідомлення, утворює код. Правило кодування може бути виражене кодовою таблицею, у якій наводяться алфавіт кодованих повідомлень і відповідні їм кодові комбінації. Множина можливих кодових символів називається кодовим алфавітом, а їх кількість — основою коду [26]. У загальному випадку при основі коду m правила кодування К елементів повідомлення зводиться до правил запису К різних чисел у m-ковій системі числення. Число розрядів n, що утворюють кодову комбінацію, називається розрядністю коду чи довжиною кодової комбінації [27]

Найбільш широкого застосування в системах передачі інформації набули коди з основою 2 — двійкові коди. Основними перевагами використання двійкових кодів є простота схемотехнічної реалізації елементів бінарної логіки, надійність та висока швидкодія таких елементів, невелика чутливість до зовнішніх завад, а також простота двійкової арифметики [28].

Кодування може здійснюватися як вручну, так і автоматично, засобами електронної техніки. Пристрій, який виконує операції кодування називається кодером.

Пристрій, який виконує задачу відновлення початкового повідомлення з кодових комбінацій (така задача називається декодуванням), називається декодером. Декодування — це процес відновлення дискретного повідомлення за вихідним сигналом демодулятора, яке здійснюється з урахуванням правила кодування. Якщо для передачі був застосований завадостійкий або коригуючий код, то на виході декодера утворюються кодові комбінації первинного (простого) коду. [22]

Якщо кодер та декодер конструктивно об’єднані в один пристрій, то він називається – кодек [29]. Кодування повідомлень, отриманих від джерела повідомлень, називається кодуванням джерела [29].

Для передачі лінією зв’язку код необхідно перетворити на сигнал. Процес такого перетворення називається модуляцією (цифровою модуляцією, маніпуляцією). Зворотне перетворення називається демодуляцією.

На рисунку 1.8 наведено структурну схему системи зв’язку.

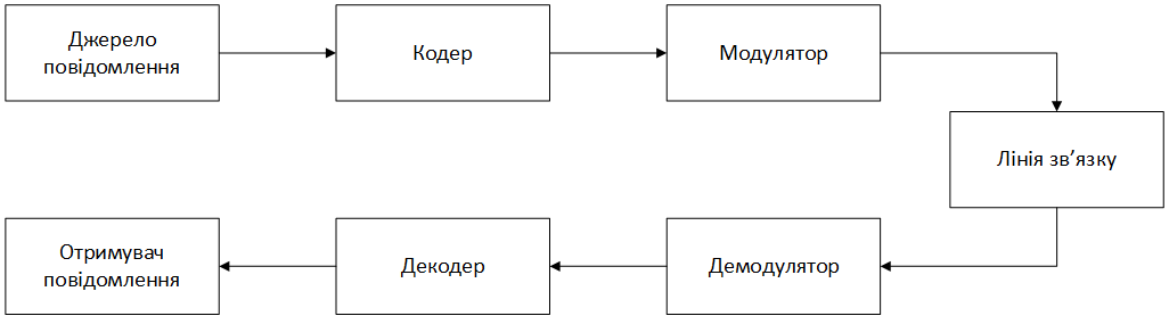

Рисунок 1.8 — Структурна схема передачі дискретних повідомлень

Останнім етапом у процесі збору даних є обробка отриманих даних та їх накопичення. Обробка даних здійснюється, зазвичай, на робочих станціях, тобто на потужних персональних комп’ютерах, якими оперують відповідні фахівці.

До даних, отриманих системами збору даних, застосовуються різноманітні методи обробки. Вони можуть варіюватися від простої побудови графіків до застосування складних алгоритмів, наприклад статистичного аналізу. Велика кількість комерційних пакетів програмного забезпечення мають багато з цих вбудованих можливостей. Також, на основі зібраних даних можуть генеруватись сигнали управління устаткуванням та процесами, які відбуваються у досліджуваній системі [7].

1.3 Апаратне забезпечення систем збору даних

Апаратне забезпечення систем збору даних виступає в ролі інтерфейсу між джерелами даних та комп'ютерами, на яких виконується зберігання та обробка отриманих даних.

Апаратне забезпечення збору даних можна визначити як компонент системи збору даних , який виконує будь-яку з таких функцій [9]:

- введення, обробку та перетворення в цифровий формат за допомогою АЦП аналогових сигналів, отриманих від засобів вимірювання, передачу даних на комп'ютер для відображення, зберігання та аналізу;
- введення цифрових сигналів, які містять інформацію від системи або процесу;
- обробка, перетворення в аналоговий формат за допомогою ЦАП цифрових управляючих сигналів від комп'ютера для керування системою або процесом
- виведення цифрових сигналів управління.

Апаратні засоби систем збору даних представлені різноманітними датчиками, мікроконтролерами, промисловими контролерами, перетворювачами, спеціалізованим обладнанням.

Першочерговою задачею систем збору даних є отримання цих даних від джерела. Зазвичай, під отриманням даних розуміють вимірювання певних показників в досліджуваній системі. Для вимірювання застосовуються спеціалізовані пристрої — вимірювальні перетворювачі, які в технічній літературі ще називають датчиками або давачами [11].

Датчик (вимірювальний перетворювач) — засіб вимірювання, призначений для перетворення вхідної вимірюваної величини у електричний сигнал, форма якого є зручною для передачі, подальшого перетворення, обробки та (або) зберігання [3, 30].

Вимірювальні перетворювачі можуть бути реалізовані по-різному. Найпростішим варіантом реалізації вимірювального перетворювача є первинні вимірювальні перетворювачі. Вимірювальний перетворювач, на який безпосередньо впливає, вимірювана фізична величина, тобто перший перетворювач у вимірювальному ланцюгу (каналі) засобу вимірювань, називається первинним вимірювальним перетворювачем або просто первинним перетворювачем [3]. В англомовній літературі відповідним терміном є «sensing element» [31]. В даній роботі для означення первинного вимірювального перетворювача використовується термін «сенсор».

Здебільшого сенсори продукують виміряні дані у вигляді електричної величини: активної (електрорушійної сили, струму) чи пасивної (активний опір, ємність, індуктивність) [11].

Наступним кроком у процесі розвитку вимірювальних перетворювачів стали інтелектуальні вимірювальні перетворювачі (далі – інтелектуальні датчики, датчики). Розвиток датчиків обумовлений все більшим ускладненням систем, в яких вони застосовуються.

Інтелектуальні вимірювальні перетворювачі це складні електронні пристрої. Такий пристрій може складатись із сенсора, певної схеми обробки сигналу від сенсора (схема кондиціонування) та деякого процесорного елемента, наприклад мікроконтролера [31]. На рисунку 1.9 наведено загальну структурну схему датчика.

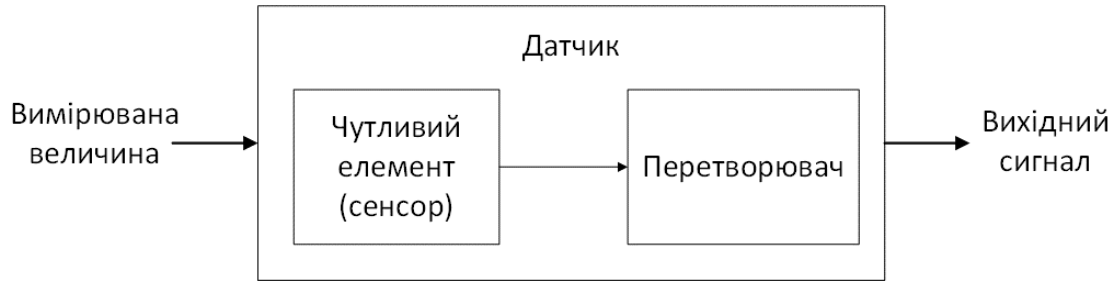

Рисунок 1.9 — Структурна схема датчика

На рисунку 1.7 зображено елемент «перетворювач». Під ним слід розуміти частину датчика, яка виконує кондиціонування первинного сигналу та формування вихідного сигналу.

Інтелектуальні датчики можуть містити в своєму складі окрім сенсора ще додаткові підсистеми, такі як наприклад [31, 32]:

- підсилювач;
- фільтр;
- схема корекції похибки;
- схема компенсації (наприклад температурної);
- процесорні елементи (мікроконтролер);
- комунікаційні інтерфейси.

В деяких публікаціях [33] терміном «інтелектуальний датчик» (smart sensor) визначають пристрій до складу якого входять: сенсор (чутливий елемент), аналого-цифровий перетворювач, певна схема кондиціонування сигналу та інтерфейсна шина. На рисунку 1.10 наведено варіант структурної схеми на основі мікроконтролера.

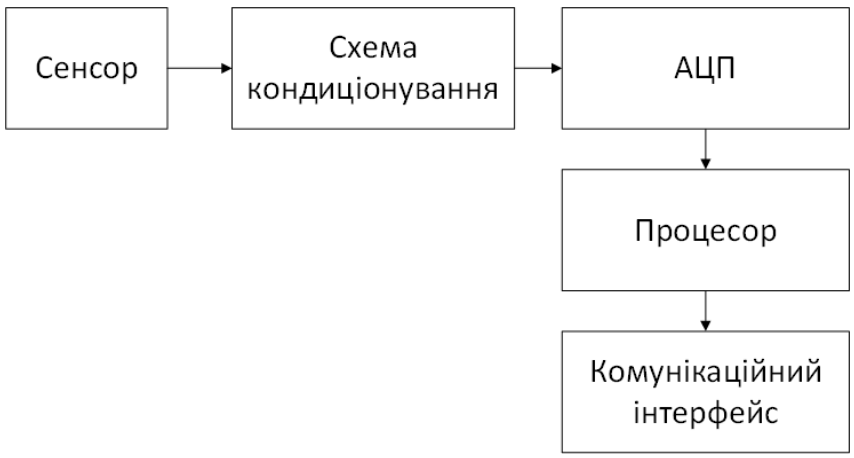

Рисунок 1.10 — Структурна схема датчика на основі мікропроцесора

При такому варіанті побудови датчика сенсорний елемент сприймає цільову вимірювану величину. Далі сигнал від сенсора передається на схему кондиціонування, де він піддається певним маніпуляціям перед тим як буде виконано аналого-цифрове перетворення цього сигналу. Аналого-цифровий перетворювач перетворює неперервний сигнал на цифровий. Процесор виконує певні обчислення на основі отриманого від АЦП коду. Наприклад, якщо вимірюється температура, то задачею процесора може бути визначення значення температури із заданою точністю у заданих одиницях вимірювання. Нарешті, інформація від процесора повинна бути передана до зовнішнього пристрою. За це відповідає комунікаційний інтерфейс [12].

Із особливостей щойно наведеного варіанту структури датчика є те, що наявність процесора (мікроконтролера) надає можливість здійснювати передачу даних від зовнішнього пристрою. Ця особливість може бути використана, наприклад, для калібрування.

Один датчик може містити більш ніж один первинний вимірювальний перетворювач. Структурна схема такого пристрою наведена на рисунку 1.11 [31].

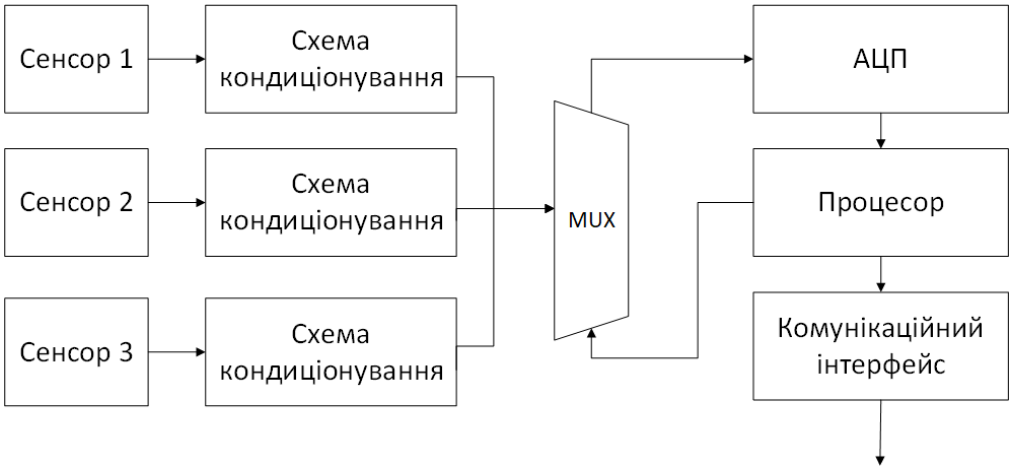

Рисунок 1.11 — Структурна схема датчика з декількома сенсорами

В схемі такого датчика є декілька сенсорів. Для кожного сенсора впроваджується власна схема кондиціонування. Для почергової передачі сигналу від сенсорних елементів до АЦП застосовується мультиплексор, управління яким здійснюється процесорним елементом датчика.

На рисунку 1.13 наведено одноплатний комп’ютер Raspberry Pi із платою розширення, яка містить аналогово-цифровий перетворювач та певну кількість виводів для вхідних сигналів [34]. Такий пристрій у поєднанні із відповідним програмним забезпеченням може бути використаний для збору даних від невеликого числа джерел.

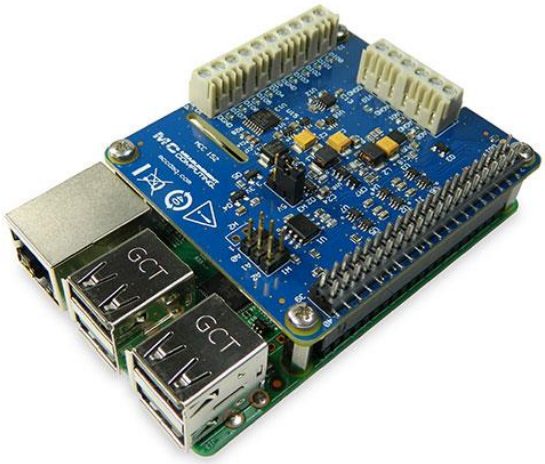

Рисунок 1.12 — Одноплатний комп’ютер RaspberryPi із платою розширення

Прикладом більш просунутих засобів збору даних є спеціалізовані пристрої. Вони можуть бути модульними або виконаними як єдиний пристрій.

Модульні пристрої складаються із різних модулів, які відповідають за обробку різних типів сигналів. До модулів підключається вимірювальне обладнання, а пристрій збору даних підключається до комп’ютера, який відповідає за збереження, обробку та візуалізацію даних [35]. Приклад такого модульного пристрою наведено на рисунку 1.13.

Рисунок 1.13 — Модульний пристрій збору даних

До складу DAQ пристрою входить схема кондиціонування сигналів, аналого-цифрові перетворювачі, комунікаційна шина (USB, PCIe, Ethernet тощо).

1.4 Програмне забезпечення для збору даних

Програмне забезпечення систем збору даних є не менш важливим ніж апаратна частина. Саме програмне забезпечення дозволяє маніпулювати даними від апаратного забезпечення.

Програмне забезпечення для збору даних може контролювати хід збору даних, надавати засоби для їх аналізу та візуалізації. Без програмного

забезпечення системи збору даних були б не більш ніж простими реєстраторами даних.

Додатковою сферою використання програмного забезпечення є можливість здійснення контролю. В процесі обробки отриманих даних можуть генеруватись управляючі команди, які можуть здійснювати управління деякими аспектами систем, наприклад, в автоматизованих засобах керування промисловими процесами [36].

Програмне забезпечення збору даних виконує такі задачі:

- контроль апаратного забезпечення збору даних;
- надання програмних інтерфейсів для взаємодії апаратного забезпечення та прикладних програм;
- надання програмних інтерфейсів для взаємодії з іншими програмними системами;
- накопичення та збереження отриманих даних на носіях;
- візуалізація даних в реальному часі [9, 36, 37].

До класу програмного забезпечення збору даних може бути віднесено як програми для мікроконтролерів, так і масштабні програмні системи, які використовується у виробничих умовах для контролю за процесом збору даних від сотень або тисяч джерел. Прикладом промислової програмної системи, яка може бути використана для створення системи збору даних є Aveva System Platform.

Висновок за розділом

Розвиток науки, удосконалення виробничих процесів, розвиток майже всіх без виключення галузей народного господарства, оборонної промисловості, в значній мірі залежить від своєчасного та якісного збору вимірювальної інформації. До сфер застосування систем збору даних відносяться системи управління виробництвом, системи життєзабезпечення, метеорологічні дослідження, дослідження у сфері машинобудування, системи контролю тощо. Виокремлення систем збору даних як окремого класу систем необхідне для розвитку такого типу систем.

Було проведено дослідження основних положень про системи збору даних та виділено ці системи як клас інформаційних систем. Виконано розробку структурних схем. Створено функціональну модель системи збору даних. Визначено, що даний клас систем представляє собою програмно-апаратні системи, які застосовуються для вимірювання, обробки на накопичення даних. Першочерговою задачею таких систем є отримання даних від досліджуваної системи. З цією метою застосовуються вимірювальні перетворювачі – датчики. Датчики оперують на нижньому рівні систем збору даних. Вони виконують функцію вимірювання кількісних показників, якими характеризується досліджувана система. Після одержання виміряного сигналу він проходить процес кондиціонування, тобто сигнал, отриманий від вимірювального пристрою, приводиться у вигляд придатний для подальшої

обробки іншими пристроями. Кондиціонування може включати підсилення, фільтрацію, лінеаризацію тощо. Так як система обробки даних є цифровою, а виміряний сигнал, зазвичай, є аналоговим, то в процесі збору даних має місце етап аналого-цифрового перетворення, яке здійснюється за допомогою АЦП. Після аналого-цифрового перетворення виміряний сигнал надходить у комп’ютерну систему для подальшої обробки.

2 ОГЛЯД ПРОГРАМНОГО ПРОДУКТУ AVEVA SYSTEM PLATFORM

У попередньому розділі було сказано, що програмне забезпечення систем збору даних є ключовою складовою та багато у чому визначає ефективність такої системи. Наразі існує велика кількість програмних комплексів, які застосовуються для розгортання систем збору даних. Прикладом такої системи є Aveva System Platform. Далі буде проведено огляд даного програмного забезпечення.

2.1 Загальні поняття про System Platform

Aveva System Platform — це програмна платформа, яка призначена для створення промислових програмних систем. В основі даного програмного забезпечення лежить технологія ArchestrA.

Технологія ArchestrA базується на технологіях розроблених корпорацією Microsoft. До складу ArchestrA входять: репозиторій об’єктів, середовище виконання програм, засоби конфігурації та комунікації, розгортання, діагностування, засоби розробки програм та візуалізації. Загалом, набір перерахованих вище компонентів формує фреймворк, тобто структуру, яка спрощує розробку програм [38].

Фреймворк ArchestrA призначений для розробки систем диспетчерського контролю і виробничих інформаційних систем. Він є відкритим, що означає, що існує можливість його розширення. Також, варто зазначити, що даний фреймворок використовує об’єктно-орієнтований підхід для створення додатків автоматизації [39].

До складу програмного комплексу System Platform входять сервіси для створення гнучких масштабованих програмних рішень для підприємств, а саме: SCADA, засоби організації людино-машинного інтерфейс для диспетчерського управління, система управління виробничими процесами (MES), засоби для впровадження інтелектуального виробництва на підприємстві (EMI). Перелічені сервіси (рисунок 2.1) дозволяють створити єдину модель підприємства, в якій використовуються логічні представлення процесів виробництва та виробничого обладнання. Система Платформа передбачає можливість інтеграції навіть вже застарілих систем, що робить проектування та обслуговування цих систем більш ефективним та гнучким, дозволяє зменшити ризики під час експлуатації таких систем. Логічне представлення підприємства у вигляді моделі значно полегшує процес менеджменту [40].

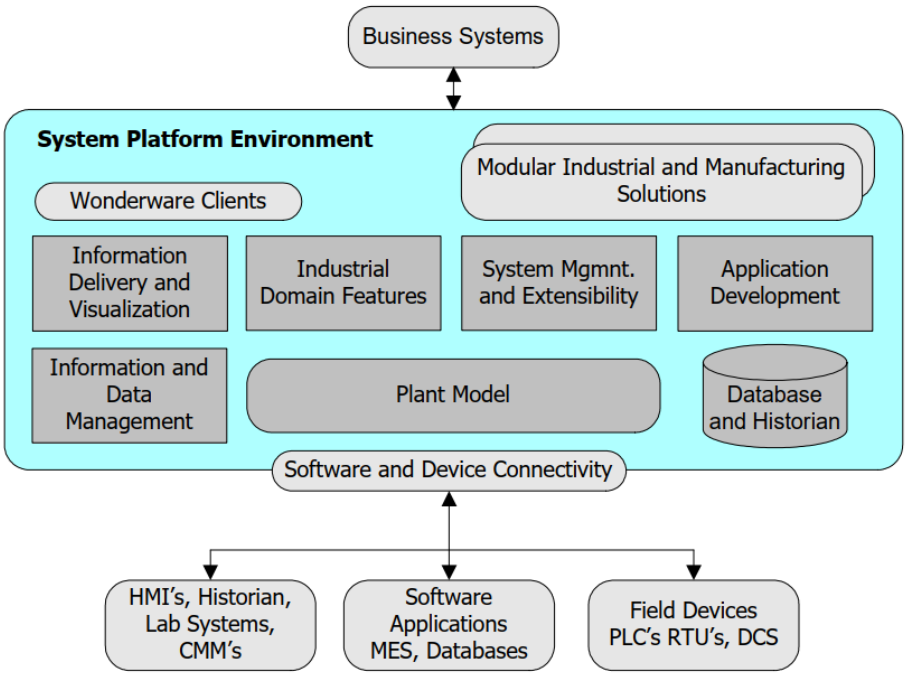

Рисунок 2.1 – Сервіси, які надаються Системною Платформою

Основними складовими частинами, які формують Системну Платформу є

- Application Server;
- Historian Server;
- Information Server;
- InTouch.

На рисунку 2.2 наведено функціональне представлення Aveva System Platform.

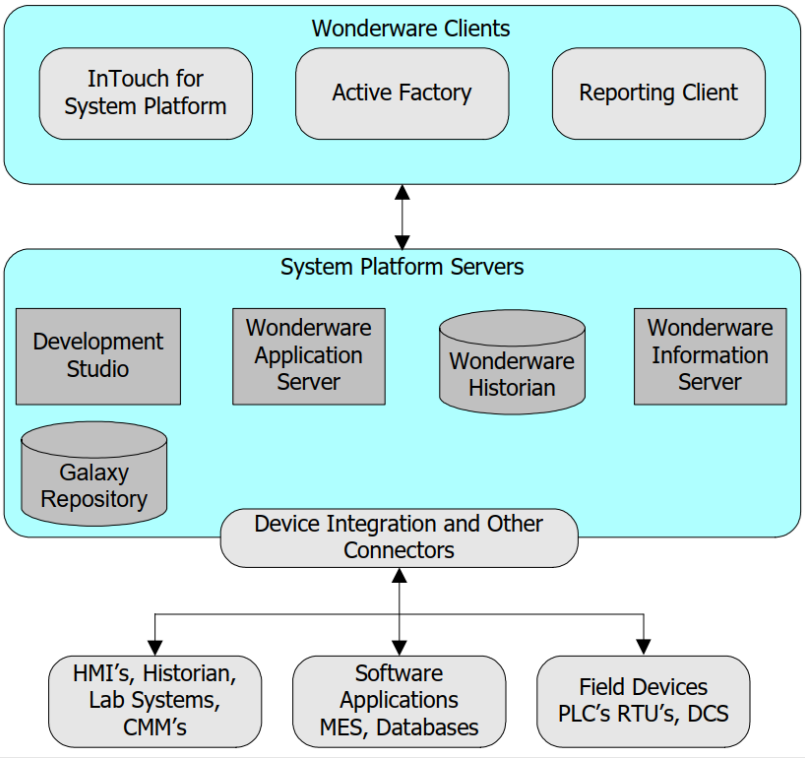

Рисунок 2.2 – Функціональне представлення Aveva System Platform

Application Server — це середовище, яке забезпечує потреби візуалізації, збору архівних даних, об’єднання пристроїв зв’язку в єдину мережу, інтеграції програм в рамках Системної Платформи. Application Server організує і управляє роботою програмних продуктів Aveva для реалізації необхідного функціоналу візуалізації, збереження даних, інтеграції обладнання. Application Server — це ядро системи. Application Server розміщує в собі всі об'єкти додатку, тут же здійснюється обробка вхідних даних, їх буферизація. Дані, якими оперують всі компоненти Платформи надходять саме від Серверу Додатків[41, 42].

Накопичення зібраних даних здійснюється із використанням компоненту Historian. Historian — це високопродуктивна база даних, яка здатна зберігати величезну кількість інформації, що продукується в ході роботи підприємства [43]. На рисунку 2.3 наведено функціональне представлення того, як зберігаються дані в Historian.

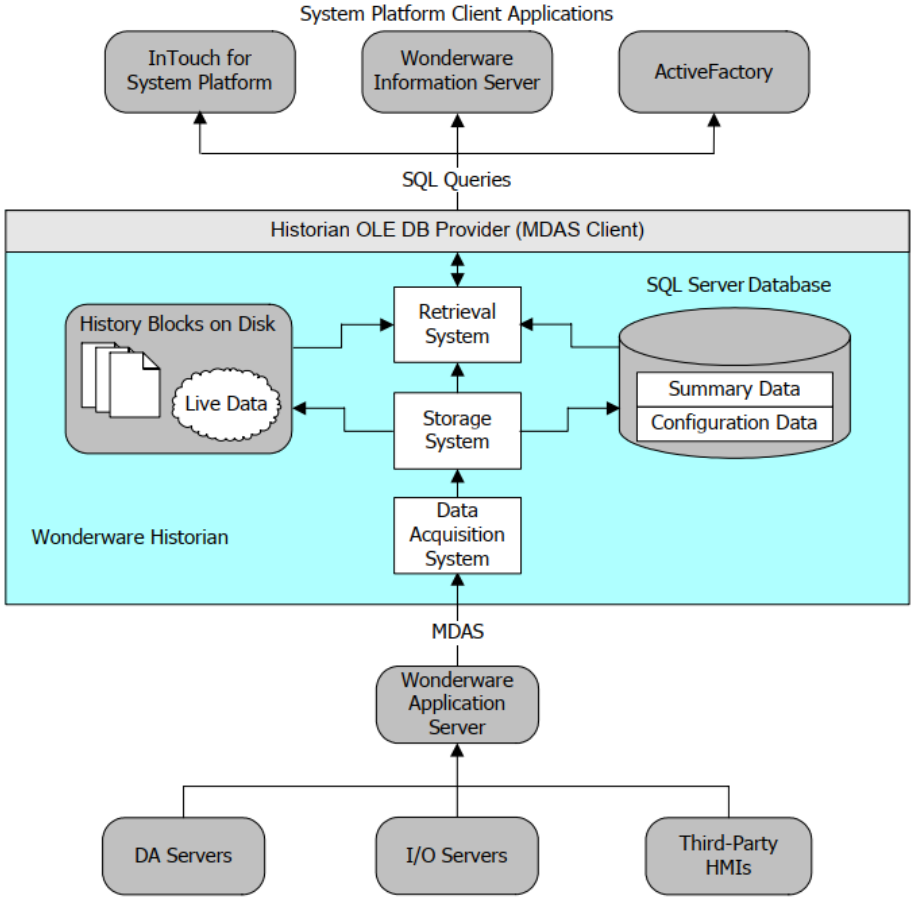

Рисунок 2.3 – Функціональна схема Historian

Автоматизація збору даних будь-якого типу, що генеруються в ході виробництва, дозволяє краще розуміти продуктивність підприємства. Зібрані дані можуть використовуватись для формування звітів, вдосконалення стратегії розвитку підприємства. У компонент Historian закладено ефективні сучасні методи стиснення та зберігання інформації, які у комбінації із стандартизованим інтерфейсом запитів дозволяє забезпечити швидкий доступ

до даних і зменшити час на прийняття рішень. Historian поєднує можливості систем збору даних та реляційних баз даних. Розширюючи технологію Microsoft's SQL Server, Historian дає можливість одночасно контролювати та управляти процесами виробництва та бізнесу, за допомогою засобів адміністрування [44].

Historian характеризується високою відмовостійкістю, дозволяє збирати інформацію із територіально віддалених об'єктів, забезпечує надійну передачу даних навіть за умов передачі по мережах із низькою пропускною здатністю. Функціональні можливості Historian дозволяють без значних зусиль та затрат поєднати виробничу частину підприємства із офісом управління [44].

Aveva Information Server — це інформаційний web-портал для представлення виробничих даних, який надає фахівцям підприємства зручний доступ до даних, які надходять до системи.

Information Server відображає інформацію про стан обладнання, яка надходить від програм HMI, баз даних, серверів введення-виведення та серверу архівації Historian. На рисунку 2.4 узагальнено типові джерела інформації, яка відображається засобами Aveva Information Server.

Використовуючи інформаційний сервер, працівники можуть швидше виявляти та вирішувати проблеми, так як мають доступ до повної, актуальної та точної інформації в режимі реального часу.

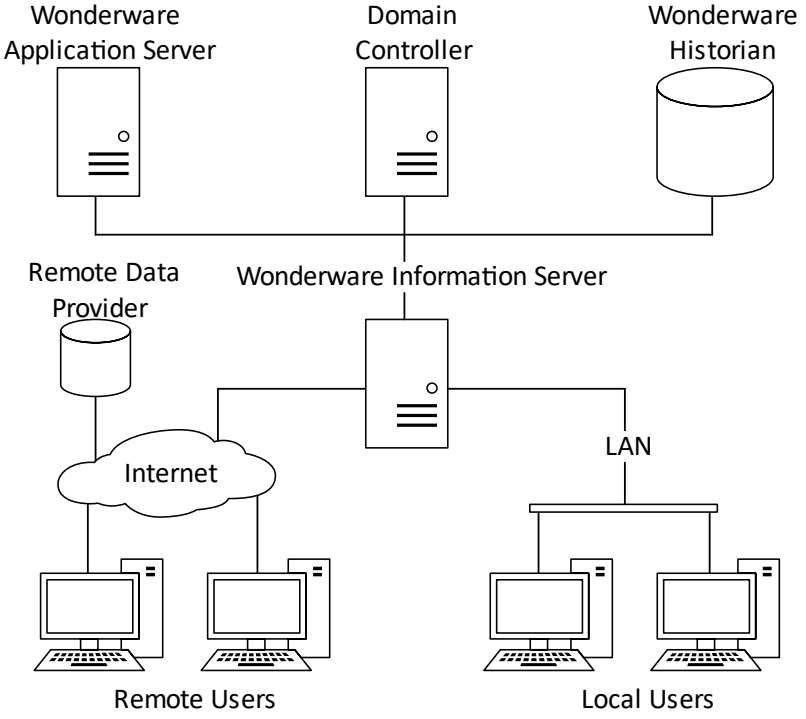

Рисунок 2.4 – Джерела інформації для Information Server

Завдяки тому, що Information Server реалізовано як веб-додаток, фахівці мають змогу віддалено підключаються до нього практично з будь-якого місця. Information Server складається із таких складових частин:

- Web Content Server — надає засоби для створення вмісту веб-сторінок, інформаційних панелей та звітів, може використовувати графіки та таблиці із запитом до бази даних;
- Web Server framework;
- Web Clients — клієнтський додаток, що дозволяє персоналу отримати доступ до широкого спектру операційної інформації [45].

InTouch — рішення для людино-машинного інтерфейсу SCADA та інших систем автоматизації, яке дозволяє швидко розробляти операторські інтерфейси. InTouch це гнучке та високопродуктивне рішення, що працює у режимі реального часу та забезпечує високий рівень безпеки [46].

Програмне рішення InTouch складається з двох основних компонентів, а саме середовища розробки та середовища виконання. Середовище розробки призначене для розробки мнемосхем, прив’язки вхідних та вихідних сигналів до апаратних засобів, розробки алгоритмів управління, адміністрування. Середовище виконання призначене для забезпечення функціонування програм, створених у середовищі розробки. Для обміну даними із апаратними засобами, такими як промислові контролери, необхідне використання серверу введення-виведення [47].

Використання сервісу InTouch дозволяє вирішувати або спростити вирішення таких задач:

- отримання даних від датчиків та контролерів;
- візуалізація зібраних даних;
- автоматичний контроль за значеннями визначених параметрів, генерація сигналів попередження у випадку виходу за межі заздалегідь визначеного діапазону;
- розробка та виконання програм управляння виробничим процесом;
- організація авторизованого доступу до системи для забезпечення безпеки;
- автоматичне ведення журналу подій;
- генерація звітів [46].

InTouch HMI зв’язаний із інтегрованим середовищем розробки ArchestrA. Якщо ArchestrA IDE інстальовано разом із InTouch, то програми можуть створюватись з використанням графічних компонентів ArchestrA [48].

2.2 Основні поняття та складові ArchestrA IDE

У Aveva System Platform для розробки додатків використовується об'єктно-орієнтований підхід. Об'єкт автоматизації це, як правило, логічне представлення якогось реального фізичного пристрою в Системній Платформі, наприклад насоса, вентилятора, PID-регулятора, аналогового або дискретного датчика тощо. При конфігурації об'єкта автоматизації описуються:

1. вхідні та вихідні сигнали, що зв'язують об'єкт автоматизації з польовими пристроями;
2. налаштування аварійних станів, пріоритети тих чи інших аварійних ситуацій;
3. скрипти та логіка обробки даних усередині об'єкт автоматизації;
4. перелік атрибутів, що архівуються, та параметри оптимізації архівування;
5. дозволи на доступ до атрибутів об'єкт автоматизації в режимі виконання;
6. текстовий опис об'єкта автоматизації;
7. графічне представлення.

ArchestrA IDE — це середовище розробки додатків автоматизації для Application Server. Це середовище надає інструменти для проектування, розробки та конфігурації об’єктів ArchestrA, конфігурації параметрів об’єктів, із яких складається програма [49].

Технологія ArchestrA використовує загальний простір імен, який називається Galaxy. Galaxy — це особливе середовищем виконання програм, розроблених в ArchestrA IDE. До складу Galaxy входять комп’ютери та компоненти, на яких працює додаток. Галактика це вся програма в цілому, яка складається з єдиного простору логічних імен, які визначаються базою даних Galaxy, та набору об'єктів платформ, обробників та інших об'єктів. Набір перелічених компонентів зберігається в базі даних Galaxy [39].

Рисунок 2.5 – Простір імен Galaxy

База даних Galaxy може знаходиться на окремому виділеному комп'ютері. Окремий комп'ютер, на якому розташована база даних Galaxy, називається Galaxy репозиторієм. Децентралізація та розбиття бази даних

Галактики не передбачена та не може бути здійснета. Різні компоненти Galaxy можна розгортати на декількох комп’ютерах для розподілу експлуатаційного навантаження під час роботи. База даних Галактики може бути розгорнута на будь-якому комп'ютері в мережі, якщо на цьому ПК встановлено ПЗ сервера SQL та ПЗ початкового завантаження репозиторію Galaxy. Простір імен Galaxy це набір унікальних ідентифікаторів об'єктів та атрибутів. Простір імен і значення кожного з його ідентифікаторів визначають додаток, розміщений на сервері додатків Aveva. Перевагою загального простору імен сервера додатків є те, що це дозволяє звертатися до даних об'єктів і процесів сервера додатків за допомогою скриптів, анімаційних посилань тощо з будь-якої робочої станції в системі без необхідності впровадження механізму організації посилань на об'єкти, які знаходяться на віддаленому вузлі [39].

У Aveva System Platform для розробки додатків використовується об'єктно-орієнтований підхід. Об'єкт автоматизації це, як правило, логічне представлення якогось реального фізичного пристрою в Системній Платформі, наприклад насоса, вентилятора, PID-регулятора, аналогового або дискретного датчика тощо. При конфігурації об'єкта автоматизації можуть описуватись:

1. вхідні та вихідні сигнали, що зв'язують об'єкт автоматизації з польовими пристроями;
2. налаштування аварійних станів, пріоритети тих чи інших аварійних ситуацій;
3. скрипти та логіка обробки даних усередині об'єкт автоматизації;
4. перелік атрибутів, що архівуються, та параметри оптимізації архівування;
5. дозволи на доступ до атрибутів об'єкт автоматизації в режимі виконання;
6. текстовий опис об'єкта автоматизації;
7. графічне представлення [41].

Створення об'єктів автоматизації організовано за шаблонним принципом. У шаблон зазвичай заноситься відомості, які є спільними для групи однотипних об’єктів. На основі цих шаблонів створюються екземпляри об’єктів, що представляють реальні пристрої системи автоматизації.

Шаблони — це елементи сервера додатків Aveva, які містять спільні параметри конфігурації для екземплярів об'єкта [50]. Поняття шаблону схоже на поняття класу в об’єктно-орієнтованому програмуванні.

Екземпляри — це об’єкти, створені на основі шаблонів. Екземпляри об’єктів це конкретні об’єкти, які існують у середовищі: процеси, клапани, конвеєрні стрічки, резервуари, датчики тощо. Значення параметрів цих об’єктів можуть надходити від, наприклад, датчиків, розміщених на реальному пристрої, або генеруватись в процесі роботи програми на сервері додатків. Разом шаблони та екземпляри називаються об'єктами [39]. На рисунку показано різні типи об'єктів та їх організація.

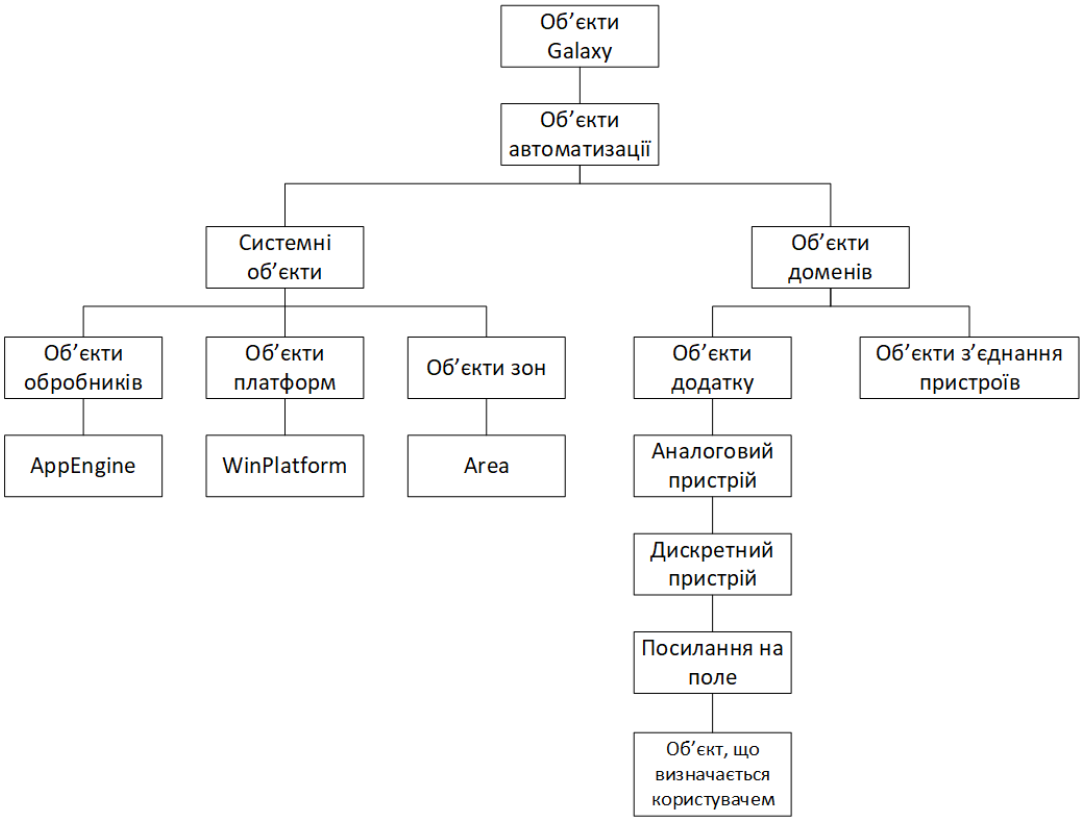

Рисунок 2.6 – Типи об’єктів

Шаблони можуть бути розширені та індивідуально налаштовані шляхом додавання атрибутів (User Defined Attributes, UDA), скриптів або розширень для задоволення конкретних вимог, обумовлених середовищем користувача.

Створення шаблонів та екземплярів схоже на об'єктно-орієнтоване програмування. Як і класи в ООП, шаблони можуть бути наслідувані від інших шаблонів. Коли шаблон є нащадком іншого шаблону, то він успадковує всі атрибути батьківського шаблону.

У шаблоні може бути вказана логіка програми, порогові значення при яких необхідно вмикати аварійні сигнали, також можуть бути вказані функції безпеки і дані, які необхідно реєструвати.

Application Server включає визначений набір шаблонів що називаються базовими шаблонами. Базові шаблони не можуть бути змінені. Всі шаблони, що створюються користувачем, створюються на основі базових шаблонів. Крім того, можна вкладати або вбудовувати одні шаблони в інші. Вбудовані шаблони складаються з вкладених шаблонів об'єктів, що є складними пристроями, які складаються з менших і більш простих пристроїв. Шаблони використовуються лише у середовищі розробки.

Для кожного шаблону передбачено набір атрибутів та значення за замовчанням. При створенні екземпляру, він успадковує атрибути батьківського шаблону та їх значення за замовчанням. Конфігурація атрибутів екземпляра, які були успадкованих від батьківського шаблону, доступна для зміни за умови, що можливість їх зміни не була заблокована у батьківському шаблоні.

Об'єктно-орієнтований підхід дозволяє скоротити час розробки та модифікації об’єктів. Також, принцип наслідування дозволяє створювати нові шаблони на основі вже існуючих, дозволяючи описувати підтипи однотипних пристроїв, тим самим збільшуючи гнучкість системи.

2.3 Огляд базових шаблонів та об’єктів

Базові шаблони є недоступними для зміни. Вони можуть використовуватись як такі, або для створення дочірніх шаблонів на їх основі. В ArchestrA IDE доступні такі класи шаблонів:

- шаблони додатків;
- шаблони об’єднання пристроїв;
- системні шаблони.

Базові шаблони додатків дозволяють легко створювати пристрої у додатку Galaxy. В ці шаблони включені властивості, характерні для кожного типу пристроїв. Наприклад, пристрій DiscreteDevice містить усі параметри налаштування, необхідні для дискретного пристрою. Крім того, до шаблонів можуть бути додані атрибути, скрипти та налаштування, які визначаються користувачем. Такий підхід дозволяє індивідуально налаштувати будь-який пристрій, створений на основі базового шаблону під конкретну задачу [39].

Шаблони об’єднання пристроїв відображають взаємодію із зовнішніми пристроями. Об’єкти об’єднання пристроїв (Device Integration, DI) інкапсулюють функції сервера введення-виведення в середовищі ArchestrA. Об’єкти DI – це моделі мережі та пристроїв, які пов’язані із певним додатком HMI. Ієрархія реальних пристроїв така ж, як ієрархія об’єктів DI. Наприклад, об'єкт DINetwork є віртуальним представленням мережевого порту, який використовується для з’єднання з пристроями через сервер доступу до даних. Об'єкт DIDevice відповідає об'єкту, що є реальним зовнішнім пристроєм, таким як от наприклад контролер (ПЛК) або, пов'язаним з об'єктом DINetwork [39, 51].

Системні шаблони дозволяють створювати додаткові групи на системному рівні та комп’ютерів, наприклад області, до яких додаються об’єкти, або інші хости AppEngine [50].

Фундаментальним об’єктом в рамках Галактики є об'єкт WinPlatform. Цей об'єкт є репрезентацією одиничного комп’ютера в Galaxy. WinPlatform складається із компонента, що призначений для обміну повідомленнями в межах усієї системи, набору основних служб операційної системи та апаратного забезпечення. Цей також є контейнером для обробника AppEngine. Об’єкти WinPlatform забезпечують виконання наступних функцій:

- розрахунок різноманітних статистичних даних для вузла, на якому цей об’єкт розгорнутий; статистична інформація розміщується в атрибутах;
- запуск та зупинку обробників, розгорнутих на платформі, залежно від їх типу запуску;

– моніторинг робочого стану обробників на платформі і у випадку, коли на платформі фіксується відмова обробника, об’єкт WinPlatform може зробити його перезапуск, залежно від значення атрибута перезапуску даного обробника [39].

Об’єкт AppEngine є обробником додатків. Цей об’єкт реалізує механізм, побудований на основі процесу сканування, виконання логіки, яка міститься в об'єктах автоматизації. Об’єкт AppEngine виконує такі функції:

– забезпечує розміщення та контроль об'єктів ApplicationObjects, об'єктів об'єднання пристроїв та зон;
– містить логіку для конфігурації та ініціалізації об'єктів при їх розгортанні;
– містить логіку для видалення об'єктів з оброблювача при їх згортанні;
– визначає час сканування для об’єктів [39].

Розміщення об’єктів типу ApplicationObjects відбувається у певній області або зоні. До зон можуть входити суб-зони. Об'єкти Area, головним чином, відіграють організаційну роль у групуванні екземплярів об’єктів. Зони умовно представляють місця географічного розміщення устаткування, представленого об’єктами типу ApplicationObjects. Зони також містять інформацію про аварійну сигналізацію та дозволяють надавати її користувачам, які користуються клієнтами аварійної сигналізації та здійснюють відстеження подій та моніторинг деякої зони відповідальності. Об'єкт Area забезпечує можливість запису до журналу значення трьох атрибутів:

– лічильника активних аварійних сигналів;
– лічильника непідтверджених аварійних сигналів;
– лічильника відключених або заглушених аварійних сигналів [50].

Об'єкт InTouchViewApp є відображенням програми InTouch у середовищі Aveva Application Server. Об'єкт InTouchViewApp забезпечує операції блокування, звільнення та розгортання програми InTouch.

Функціонування об'єкта InTouchViewApp забезпечується засобами об'єкту ViewEngine, який виконує такі задачі:

– забезпечення управління синхронізацією та доставкою файлів, потрібних для пов'язаного програми InTouch;
– забезпечення динамічного доступу до тегів у зв'язаному додатку InTouch;
– забезпечення запуск WindowMaker для програми InTouch під час редагування [50].

Висновок за розділом

Програмний продукт System Platform було створено як засіб для розробки програмного забезпечення для автоматизації, збору та аналізу даних, організації людино-машинного інтерфейсу, організації віддаленого доступу до систем управління. Необхідність створення програмного продукту такого

класу була обумовлення все більшим зростанням складності технологічних процесів, а з ними і систем управління цими процесами. У промислових компаній природньо виникла потреба у більш ефективних та гнучких засобах для розробки, впровадження та обслуговування програмних компонентів, які експлуатуються в ході роботи підприємства. Такими програмними продуктами стали SCADA системи, а пізніше — комплексні програмні платформи, які надають ширші функціональні можливості. Системна Платформа є продуктом, можливості застосування якого в розробці програмного забезпечення, зборі та аналізі даних, організації людино-машинного інтерфейсу є одними із найбільш просунутих на сьогоднішній день на ринку промислового ПЗ. Основною перевагою даної системи є те, що для розробки додатків застосовується об'єктно-орієнтований підхід, що в свою чергу дозволяє спростити розробку та скоротити час розробки, а також спрощує процес модифікації програм.

3 ДЖЕРЕЛО ДАНИХ

3.1 Загальні відомості про джерело даних

В якості джерела даних для розроблюваної системи збору даних використовується метеостанція, створена на базі апаратної платформи Arduino.

Дана метеостанція виконує функції вимірювання та відображення показників навколишнього середовища, а також виконує передачу виміряних параметрів на комп'ютер для їх подальшої обробки. Також серед функцій пристрою наявна функція годинника, із можливістю відображення поточного часу та дати у різних форматах, які налаштовуються програмно.

До складу пристрою входять:

- датчик температури;
- датчик атмосферного тиску;
- датчик рівня вологості повітря;
- датчик вмісту вуглекислого газу та летких органічних речовин;
- годинник реального часу.

На рисунку 3.1 наведено фото метеостанції.

Рисунок 3.1 — Метеостанція

Показання датчиків температури, тиску та вологості виводяться на головному екрані (рисунок 3.1), також ці показники більш детально можна переглянути в окремому режимі відображення.

Сигналізування про стан повітря здійснюється за допомогою RGB-світлодіоду:

- зелений колір відповідає нормальному вмісту вуглекислого газу та ЛОР; жовтий колір сигналізує про незначне перевищення вмісту CO_2 або ЛОР у повітрі
- червоний колір означає, що рівень CO_2 або ЛОР у повітрі значно перевищує нормальні значення.

Вміст вуглекислого газу у повітрі відображається у мільйонних долях (ppm, parts per million); вміст летчих органічних речовин відображається у мільярдних долях (ppb, parts rep billion).

Переглянути показники рівня вуглекислого газу та летких органічних речовин можна увімкнувши відповідний режим відображення (рисунок 3.2). Перемикання між режимами здійснюється тактовою кнопкою, вмонтованою в корпус пристрою.

Рисунок 3.2 — Режим відображення показників стану повітря

Загалом, розроблена метеостанція є простим пристроєм моніторингу за станом навколишнього середовища, із можливістю відображення вимірюваних даних на символьному LCD дисплеї. Даний пристрій придатний лише для домашнього використання.

3.2 Компоненти метеостанції

3.2.1 Arduino Nano

Основним компонентом метеостанції є платформа Arduino. Arduino – це екосистема, яка представлена великим різноманіттям апаратного забезпечення: різноманітні лінійки плат розробки на базі мікроконтролерів, плати розширення, датчики, актуатори тощо. Також до складу екосистеми Arduino можна віднести середовище розробки Arduino IDE та бібліотеки для мов програмування.

Традиційно, робота із мікроконтролерами потребувала значних затрат часу, так як для створення програм, їх компіляції та запису в

пам’ять (прошивки) необхідно було детально вивчати технічну документацію до конкретного мікроконтролера. Також, для прошивки мікроконтролера необхідно було використовувати спеціальний пристрій — програматор, що ще більше ускладнювало і без того складний процес програмування. Окрім програмного аспекту роботи із МК є ще й апаратний, який полягає у тому, що для коректної роботи будь-якого мікроконтролера необхідні ще ряд компонентів та електронних схем, які забезпечують його нормальне функціонування. Сукупність цих компонентів у колах радіолюбителів та професіональному середовищі називають обв’язкою.

Розробники Arduino мали на меті прибрати всі перераховані вище бар’єри та надати непрофесіоналам засіб для створення власних електронних пристроїв, який би був простим у освоєнні та не потребував глибоких знань.

Плати розробки Arduino містять всю необхідну обв’язку для роботи із мікроконтролером. Також мікроконтролери, які встановлені на платах Адруіно, містять спеціальну програму-завантажувач, яка дозволяє записувати користувацькі програми в пам’ять мікроконтролера без використання програматора [52].

Для створення проекту метеостанції було обрано модель Nano (рисунок 3.3), через її компактні розміри, що в свою чергу дозволяє створювати малогабаритні пристрої.

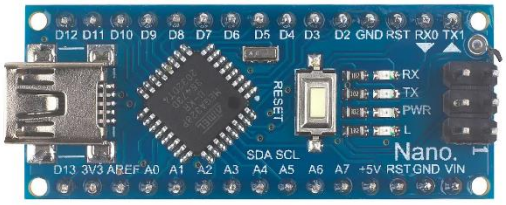

Рисунок 3.3 — Arduino Nano

Плата Arduino Nano побудована навколо мікроконтролера ATmega328P. ATmega328P це восьмирозрядний мікроконтролер із сімейства мікроконтролерів AVR в основі якого лежить процесор на базі модифікованої архітектури RISC [53].

ATmega328P містить 32 регістри загального призначення розрядністю вісім біт. Контролер має тридцять два виводи, із яких для взаємодії користувачу доступні двадцять три лінії введення-виведення загального призначення. Схема розташування виводів наведена на рисунку 3.4.

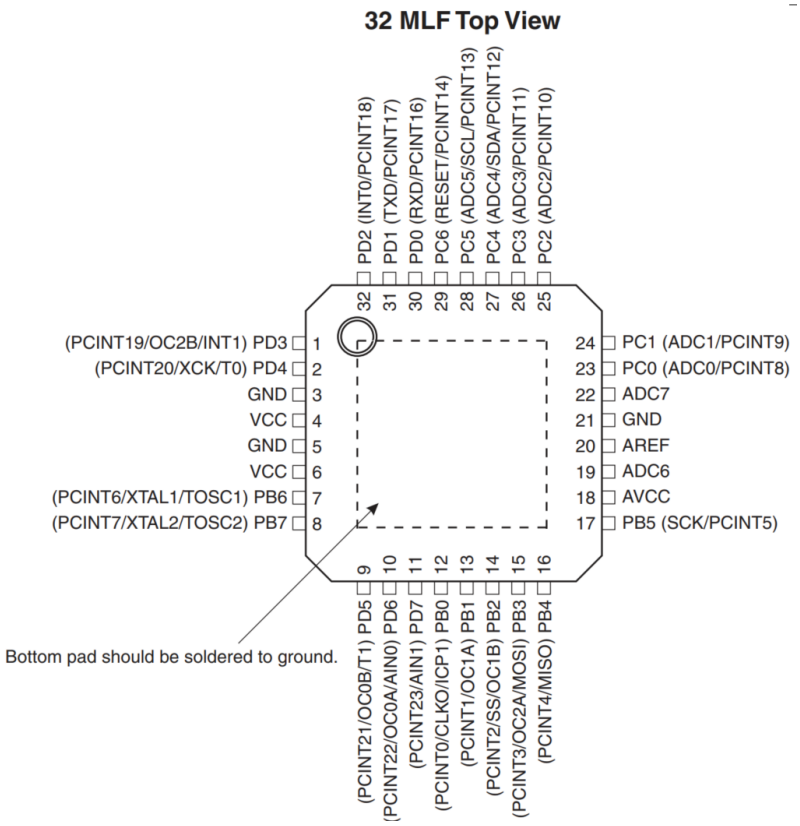

Рисунок 3.4 — Схема розташування виводів мікроконтролера ATmega328P

Щодо характеристик, які стосуються пам’яті, то даний мікроконтролер містить тридцять два кілобайти Flash пам’яті для зберігання програм, один кілобайт EEPROM пам’яті та два кілобайти основної (оперативної) пам’яті SRAM. Також, до складу мікроконтролера входить десятирозрядний аналого-цифровий перетворювач. Для обміну даними із зовнішніми пристроями в цьому МК реалізовано I2C, SPI, USART інтерфейси [53].

На рисунку 3.5 наведено структурну схему мікроконтролера ATmega328P.

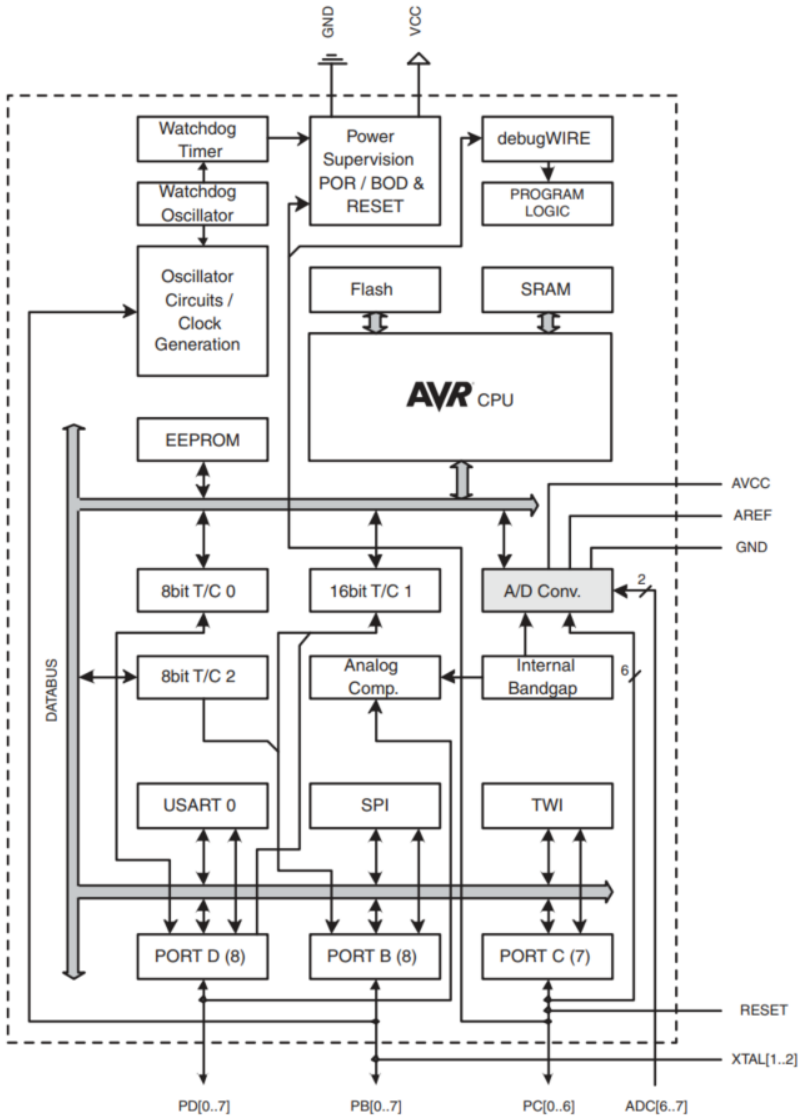

Рисунок 3.5 — Структурна схема ATmega328P

Всі виводи мікроконтролера ATmega328P доступні на платі Arduino (рисунок 3.6), а доступ до них є більш зручним ніж при безпосередній роботі із мікроконтролером.

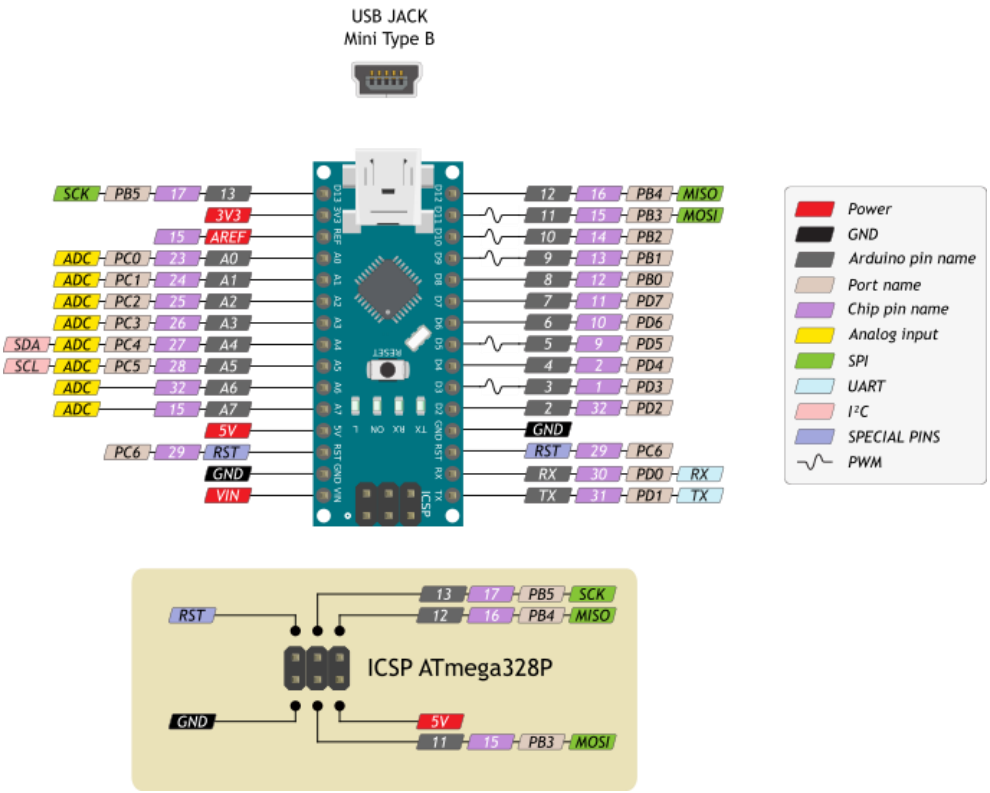

Рисунок 3.6 — Схема призначення виводів плати Arduino Nano

Програмування плат Arduino здійснюється із середовища розробки Arduino IDE (рисунок 3.7).

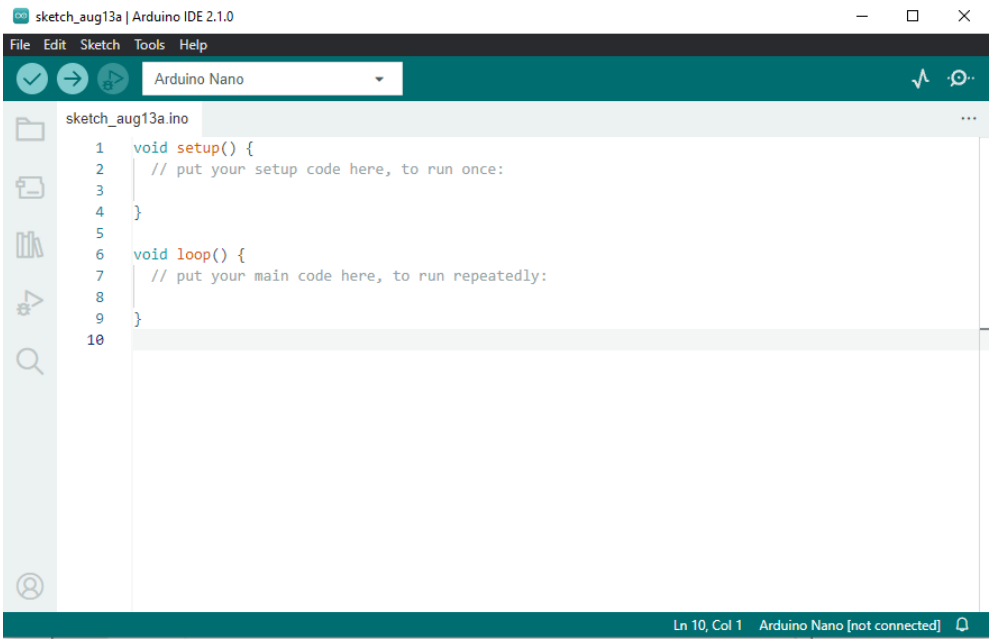

Рисунок 3.7 — Arduino IDE

Перш за все, середовище розробки містить редактор коду для написання програм. Програми для Arduino називаються скетчами. Редактор має всі необхідні функціональні можливості, які притаманні будь-якому редактору коду, такі як підсвічування синтаксису, авто-форматування або авто-

завершення коду. Незважаючи на те, що редактор коду Arduino є досить простим, його можливостей достатньо для комфортного написання скетчів [54].

Arduino IDE містить компілятор, дебагер — засіб для пошуку помилок у програмному коді, програматор для передачі згенерованого компілятором машинного коду в пам’ять мікроконтролера [55]

До складу IDE входять такі засоби як Serial Monitor та Serial Plotter. Serial Monitor дозволяє відображати текст, який передається від мікроконтролера до ПК, або передавати текст від ПК до мікроконтролера. Даний засіб часто застосовується для відлагодження. Serial Plotter це засіб для відображення числових даних у вигляді двомірних графіків [54].

До складу IDE також входить менеджер бібліотек, який дозволяє швидко та просто завантажувати бібліотеки.

Загалом, Arduino IDE це зручне середовище розробки, яке надає всі необхідні функціональні можливості для створення та налагодження програм для мікроконтролерів, їх прошивки.

3.2.2 Датчик BME280

З метою отримання значень температури, вологості та атмосферного тиску в метеостанції використовується комбінований цифровий датчик вологості, тиску та температури BME280 (рисунок 3.8).

Рисунок 3.8 — Датчик BME280

Датчик BME280 має надкомпактні розміри: всього 2.5 мм у ширину та довжину із висотою 0,93 мм. Також, цей датчик характеризується дуже низьким енергоспоживанням, що робить його підходящим для пристрої, живлення яких відбувається від батарей [56].

Цикл вимірювання датчика складається із послідовних вимірювань температури, атмосферного тиску та рівня вологості. Після вимірювання дані про температуру та тиск можуть опціонально бути піддані фільтрації через фільтр з нескінченною імпульсною характеристикою (IIR), який відсікає короткотривалі флуктуації тиску, наприклад при різкому зачиненні дверей).

Розрядність значення вологості прив’язане до розрядності АЦП та складає шістнадцять біт. Розрядність значення тиску при використанні IIR фільтру складає 20 біт, а якщо така фільтрація не використовується, то

розрядність становить 18 біт. Розрядність значення температури така ж як і розрядність тиску.

У датчику BME280 реалізовано два інтерфейси: I²C та SPI. На рисунку 3.9 наведено структурну схему даного датчика.

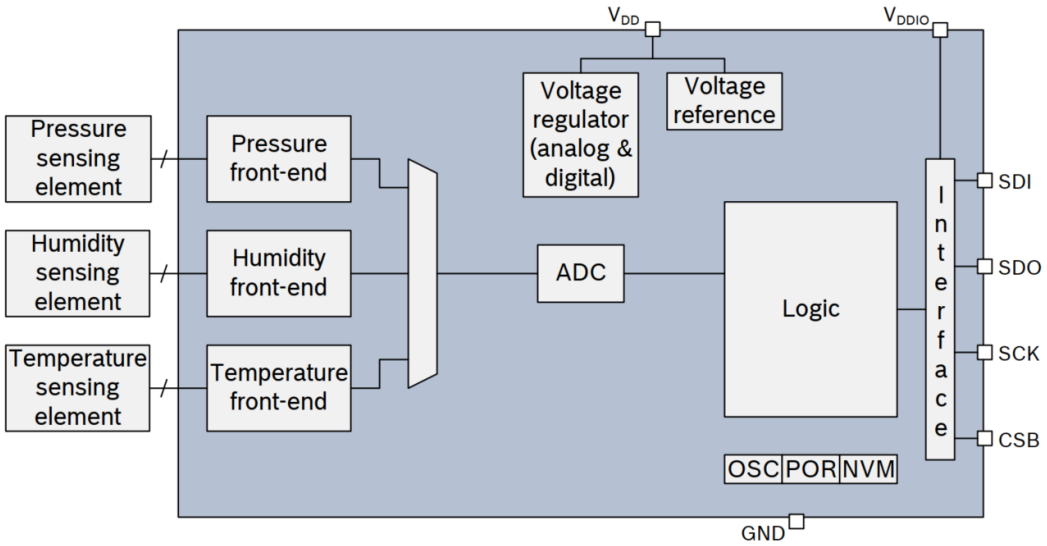

Рисунок 3.9 — Структурна схема BME280

Датчик може працювати в одному із трьох наступних режимів [56]:

1. Режим сну. В цьому режимі вимірювання не відбуваються, всі регістри доступні, енергоспоживання найменше.
2. Нормальний режим. Нескінченний цикл, який складається із періоду вимірювання та періоду очікування. Час очікування може бути встановлений в діапазоні між 0,5 мс та 1000 мс.
3. Примусовий (forced) режим. Відбувається одиночне вимірювання відповідно до встановлених параметрів вимірювання після чого датчик переходить в режим сну.

В проекті метеостанції був застосований модуль із датчиком BME280 та необхідною для роботи датчика обв'язкою (рисунок 3.10). Даний модуль підключається до мікроконтролера через інтерфейс I²C.

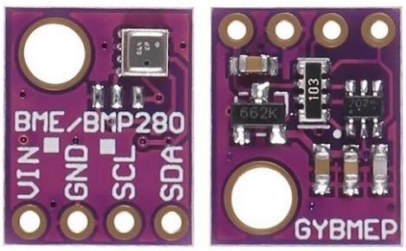

Рисунок 3.10 — Електронний модуль з датчиком BME280

3.2.3 Датчик CCS811

CCS811 — це цифрового датчика газу побудований на основі метал-оксидного сенсору. Він здатен виявляти широкий спектр летких органічних сполук (ЛОС) та може використовуватись для моніторингу за станом якості повітря. Окрім сенсора до складу датчика входить за мікроконтролер (MCU), який містить аналого-цифровий перетворювач (АЦП) і I²C інтерфейс [57].

Вбудований мікроконтролер виконує управління режимами роботи датчика та виконує обробку виміряних даних. Цифровий I²C інтерфейс істотно спрощує інтеграцію датчика з іншими пристроями, а також дозволяє без значних зусиль здійснювати передачу даних.

CCS811 підтримує інтелектуальні алгоритми для обробки виміряних значень і отримання готового вихідного значення TVOC або еквівалентного CO2 (eCO2).

Діапазон вихідного значення для летких органічних сполук (TVOC) становить від 0ppb до 1187ppb. Датчик відкалібрований відповідно до типових TVOC у приміщенні. Якщо співвідношення сполук у середовищі суттєво відрізняються від тих сполук, на яких проводилось калібрування датчика, вплив даних сполук на датчик буде відрізнятись від очікуваного. Діапазон значень eCO2 становить від 540 ppm до 8192 ppm [57].

Для температурної компенсації можуть застосовуватись показання від зовнішнього датчика. Дані про температуру записуються у відповідний регістр мікроконтролера.

На рисунку 3.11 наведено діаграму, яка відображає структуру датчика CCS811.

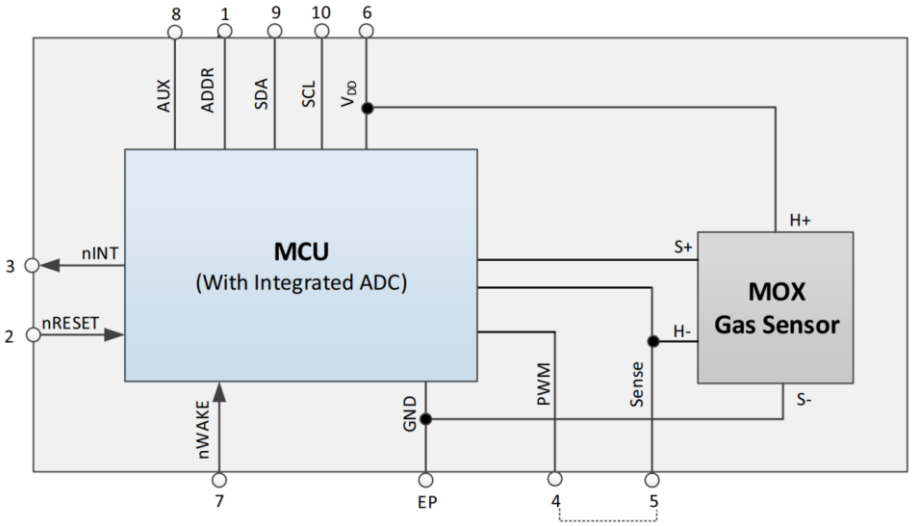

Рисунок 3.11 — Структурна схема датчика CCS811

Для створення метеостанції було застосовано плату (рисунок 3.11), на якій виконана рекомендована виробником схема включення даного датчика.

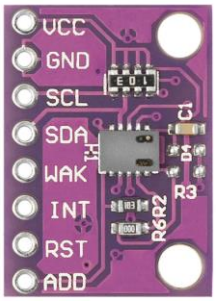

Рисунок 3.12 — Модуль із датчиком CCS811

Використання такої плати-модуля значно спрощує спряження датчика та Arduino.

3.2.4 Годинник реального часу DS3221

Однією із функцій погодної станції є функція годинника. Для визначення поточного часу в метеостанції застосовано мікросхему DS3221 (рисунок 3.13).

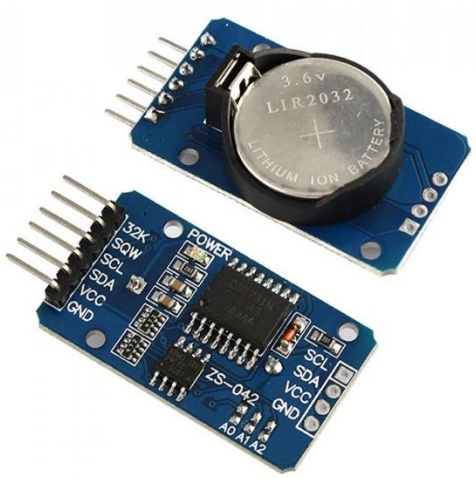

Рисунок 3.13— Модуль DS3221

DS3221 це годинник реального часу (RTC) із точністю ± 0,572 секунди на день. Даний годинник має відлік секунд, хвилин, днів, місяців та років. Дата в кінці місяця автоматично коригується для місяців, які містять менше 31 дня, включаючи поправки на високосний рік. Годинник може працювати в 24-годинному або 12-годинному форматі. Внутрішні регістри доступні через інтерфейс шини I2C. Схема опорної напруги та компаратора з температурною компенсацією контролює напругу живлення для виявлення збоїв живлення та автоматичного перемикання на резервне джерело живлення (батарейку), метою забезпечення неперервного відліку часу [58].

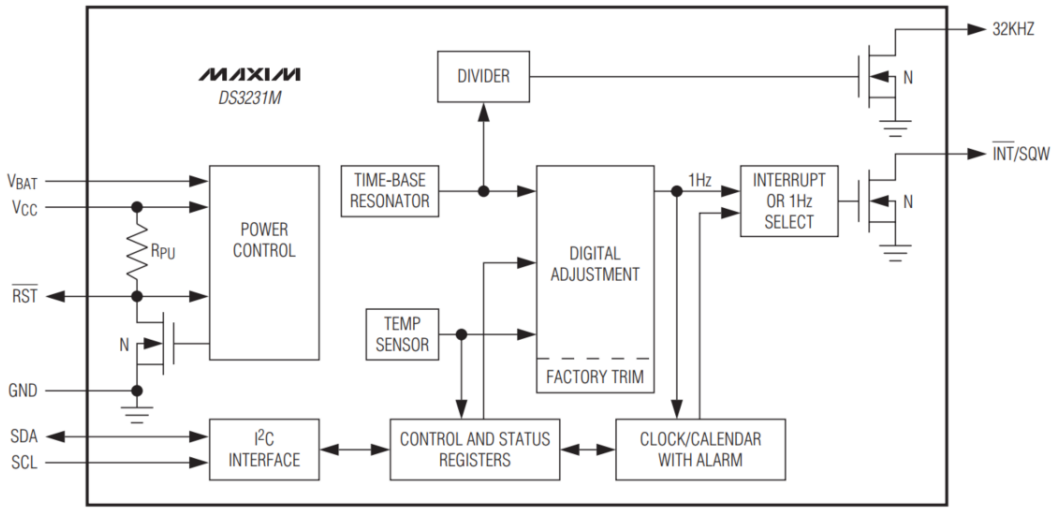

Рисунок 3.14 — Структурна діаграма DS3221

Для підвищення точності годинника застосовується температурна компенсація, для здійснення якої в мікросхемі DS3221 наявний датчик температури (рисунок 3.14).

3.2.5 Символьний дисплей

Для виведення показань датчиків застосовується символьний LCD дисплей (рисунок 3.15). У проекті метеостанції застосовано дисплей, який має чотири символьні рядки із кількість символів на рядок – двадцять.

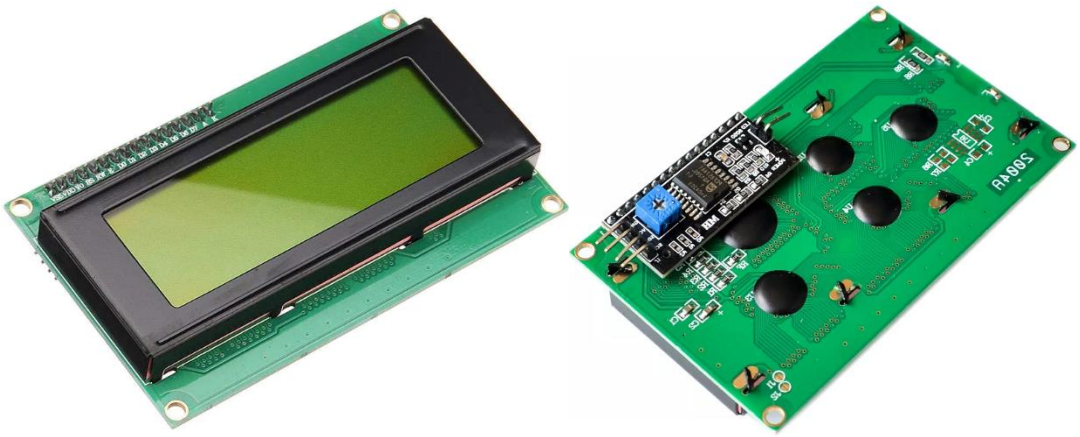

Рисунок 3.15 — Символьний LCD дисплей

Даний дисплей обладнаний адаптером для підключення по шині I2C, завдяки чому кількість провідників (дротів), необхідних для підключення дисплею до мікроконтролера, знижується до двох.

Символи, які може відображати дисплейний модуль, зберігаються в пам’яті у вигляді таблиці, поділеної на дві сторінки. В залежності від вибраної сторінки відображаються ті чи інші символи. Серед символів наявні літери латинського алфавіту, цифри, спеціальні символи.

3.3 Принципова схема пристрою

Для того, щоб із компонентів, огляд яких було проведено у попередніх підрозділах, зібрати метеостанції, необхідно дані компоненти відповідним чином під'єднати. Центральною частиною, до якої приєднуються інші частини, є Arduino Nano. Так як всі пристрої використовують для передачі даних шину I2C, то їх необхідно під'єднати до відповідних пінів на Arduino, а саме піни номер 23(A4) та 25(A5). Також, необхідно забезпечити живлення всіх пристроїв. На рисунку 3.16 наведено принципову схему метеостанції.

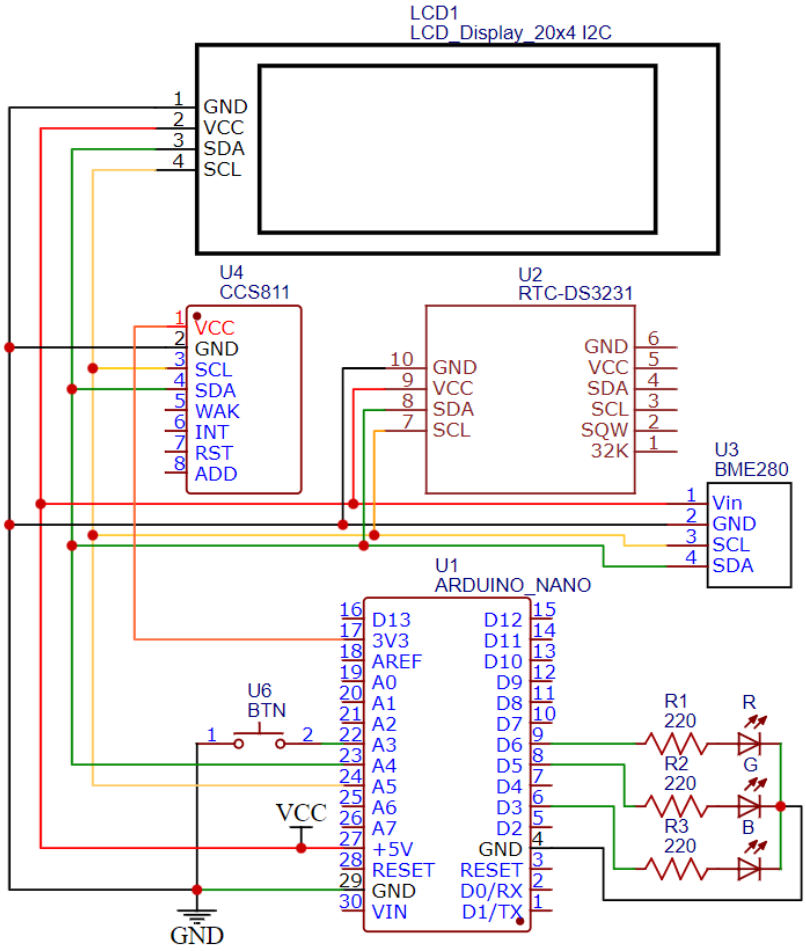

Рисунок 3.16 — Принципова схема метеостанції

Всі компоненти було з'єднано шляхом паяння згідно із наведеної вище принциповою схемою, та поміщено у пластиковий корпус. В результаті отримано готовий пристрій (рисунок 3.17).

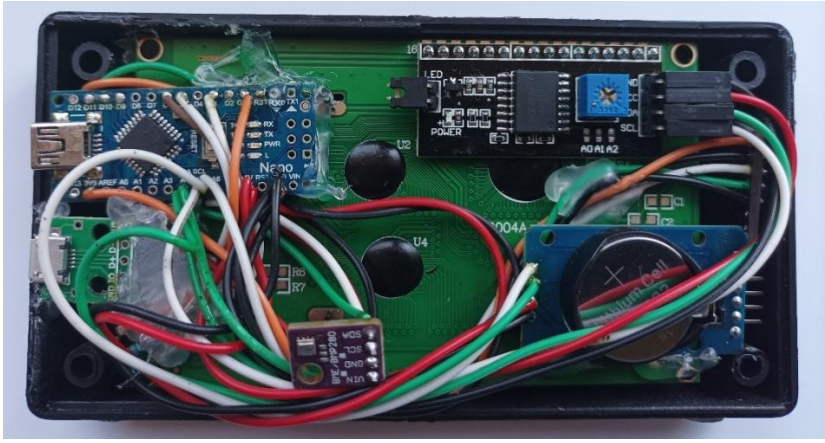

Рисунок 3.17 — Метеостанція зсередини

3.4 Програмне забезпечення метеостанції

В даній роботі не приводиться повний текст програми прошивки. Нижче коротко буде розглянуто основні функції, які стосуються отримання даних від датчиків та їх передачі.

Для отримання даних від датчика BME280 створено функції *readSensors* (лістинг 3.1). Всередині функції здійснюється виклик бібліотечних функцій читання даних із регістрів датчика.

Лістинг 3.1 — Функція отримання показань від датчика BME280

```
void readSensors(){
  bme.takeForcedMeasurement();
  temp = bme.readTemperature();
  pressure = bme.readPressure() * 0.0075; // Pa => mmHg
  hum = bme.readHumidity();
  if(mode == 0) displayShortSensors();
  if(mode == 1) displaySensors();
}
```

Отримання показань датчика CCS811 реалізовано у функції *checkAirQuality* (лістинг 3.2). Читання даних здійснюється з використанням функцій, наявних у відповідній бібліотеці.

Лістинг 3.2 — Функція отримання показань від датчика CCS811

```
void checkAirQuality(){
if(ccs.available()){
    if(!ccs.readData()){
      co2 = ccs.geteCO2();
      tvoc = ccs.getTVOC();
```

Для передачі виміряних значень у додаток Archestra застосовується технологія OPC. З метою передачі даних від ардуіно до OPC сервера було застосовано бібліотеку Arduino/Genuino OPC Library (https://github.com/ildemartinez/OPC).

Дана бібліотека передбачає використання об’єкта типу OPCSerial для передачі даних через послідовний інтерфейс (лістинг 3.3).

Лістинг 3.3 — Декларування об’єкта типу OPCSerial

```
75   Adafruit_CCS811 ccs;
76   Adafruit_BME280 bme;
77   OPCSerial aOPCSerial;
78
79   uint16_t pressure;
```

OPC-дані створюються шляхом застосування функції *addItem* (лістинг 3.4).

Лістинг 3.4 — Створення OPC-даних

```
aOPCSerial.setup();
aOPCSerial.addItem("temperature",opc_read, opc_float, callback_temp);
aOPCSerial.addItem("pressure",opc_read, opc_int, callback_press);
aOPCSerial.addItem("humidity",opc_read, opc_int, callback_hum);
aOPCSerial.addItem("CO2",opc_read, opc_int, callback_co2);
aOPCSerial.addItem("TVOC",opc_read, opc_int, callback_tvoc);
```

Перший параметр функції визначає ім’я OPC-item’а, другий параметр визначає режим доступу до даних, третій параметр визначає тип даних, четвертий параметр визначає callback функцію. Callback функцію буде виконано, коли зовнішній клієнт захоче прочитати/записати елемент даних, і виконує читання або запис значення даного OPC-item’а.

Нижче приведено всі Callback-функції, які використовуються в програмі.

Лістинг 3.5 — Callback-функції

```
float callback_temp(const char *itemID, const opcOperation opcOP, const float value){
  return temp;
}

int callback_press(const char *itemID, const opcOperation opcOP, const int value){
  return pressure;
}

int callback_hum(const char *itemID, const opcOperation opcOP, const int value){
  return hum;
}

int callback_co2(const char *itemID, const opcOperation opcOP, const int value){
  return co2;
}

int callback_tvoc(const char *itemID, const opcOperation opcOP, const int value){
  return tvoc;
}
```

Обробка запитів OPC-сервера виконується функцією *processOPCCommands()*.

Лістинг 3.6— Функція обробки запитів

```
void loop() {

  if(mode == 0 && millis() - clockTimer >= 1000){
    clockTimer = millis();
    displayDateTime();
    if(isMidnight()) displayShortSensors();
  }

  if(millis() - sensorsTimer >(long)SENSORS_PERIOD){
    sensorsTimer = millis();
    readSensors();
  }

  if(millis() - airTimer >= (long)AIR_PERIOD){
    airTimer = millis();
    checkAirQuality();
  }

  buttonTick();
  modeChange();
  lcdChangeBacklightMode();

  aOPCSerial.processOPCCommands();
}
```

Висновок за розділом

Розроблений електронний пристрій виконує функції моніторингу за станом навколишнього середовища із можливістю відображення вимірюваних даних на символьному LCD дисплеї. Основною структурною та функціональною одиницею даного пристрою є Arduino Nano — плата розробки на базі мікроконтролера ATmega328P. Проект Arduino було створено для того, щоб спростити та зробити більш зручною роботу із мікроконтролерами, а також для того, щоб знизити поріг входження до світу розробки електронних пристроїв. Застосування Arduino Nano в проекті метеостанції обумовлено її форм-фактором, наявністю достатньої кількості виводів загального призначення та низькою вартістю.

Окрім Arduino Nano до складу метеостанції входять датчики та дисплей. В розділі було виконано огляд цих компонентів, розглянуто їх основні характеристики та функціональні можливості.

Проект метеостанції розміщений в репозиторії GitHub та є доступним за посиланням: https://github.com/romanenko74/Arduino-Weather-Station-OPC.git

4 КОНФІГУРАЦІЯ СИСТЕМНОЇ ПЛАТФОРМИ

4.1 Створення Галактики

Для початку роботи із середовищем розробки ArchestrA необхідно підключитись до Galaxy. На рисунку 4.1 показано діалогове вікно з'єднання з Галактикою.

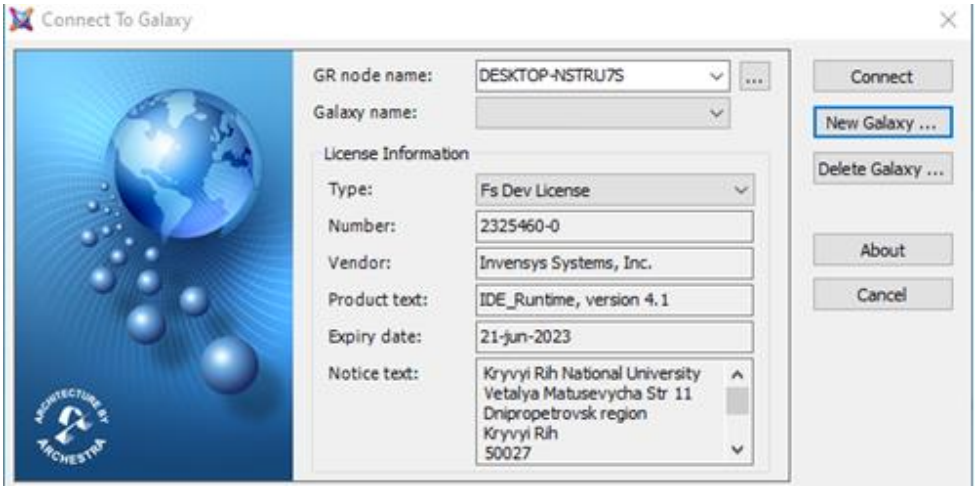

Рисунок 4.1 — Вікно з'єднання з Galaxy

Вікно з'єднання з Galaxy містить три групи елементів:

- вибір підключення, що складається із полів «GR Node» та «Galaxy name»;
- кнопки підключення до існуючої Galaxy, створення нової Galaxy, видалення існуючої Galaxy;
- ліцензійна інформація.

Для розробки додатку буде створено нову Галактику. Для створення нової Галактики необхідно у вікні з'єднання з Galaxy (рисунок 4.1) натиснути на кнопку «New Galaxy». У вікні створення Галактики (рисунок 4.2) необхідно вказати ім'я вузла, на якому буде створено нову Галактику, ім'я Галактики та тип створюваної Галактики.

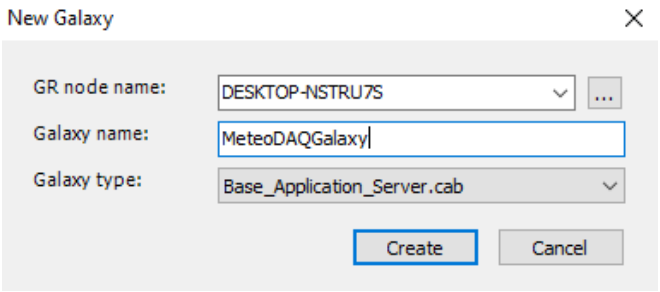

Рисунок 4.2 — Вікно створення Galaxy

Після завершення створення нової Галактики з'являється вікно із повідомленням (рисунок 4.3).

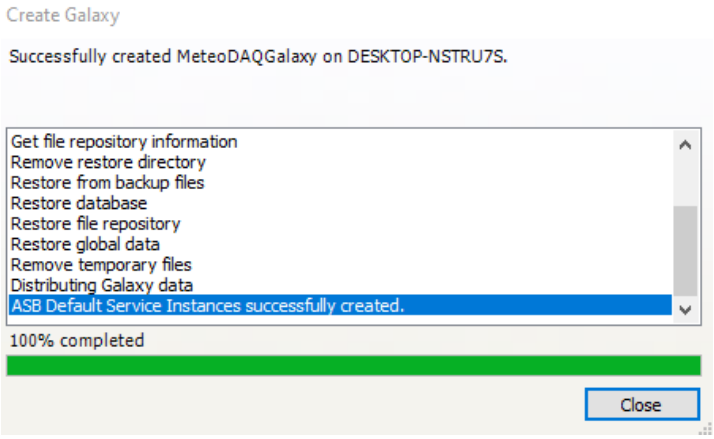

Рисунок 4.3 — Повідомлення про успішне створення Галактики

Після підключення до щойно створеної галактики відкривається головне вікно ArchestrA IDE (рисунок 4.4).

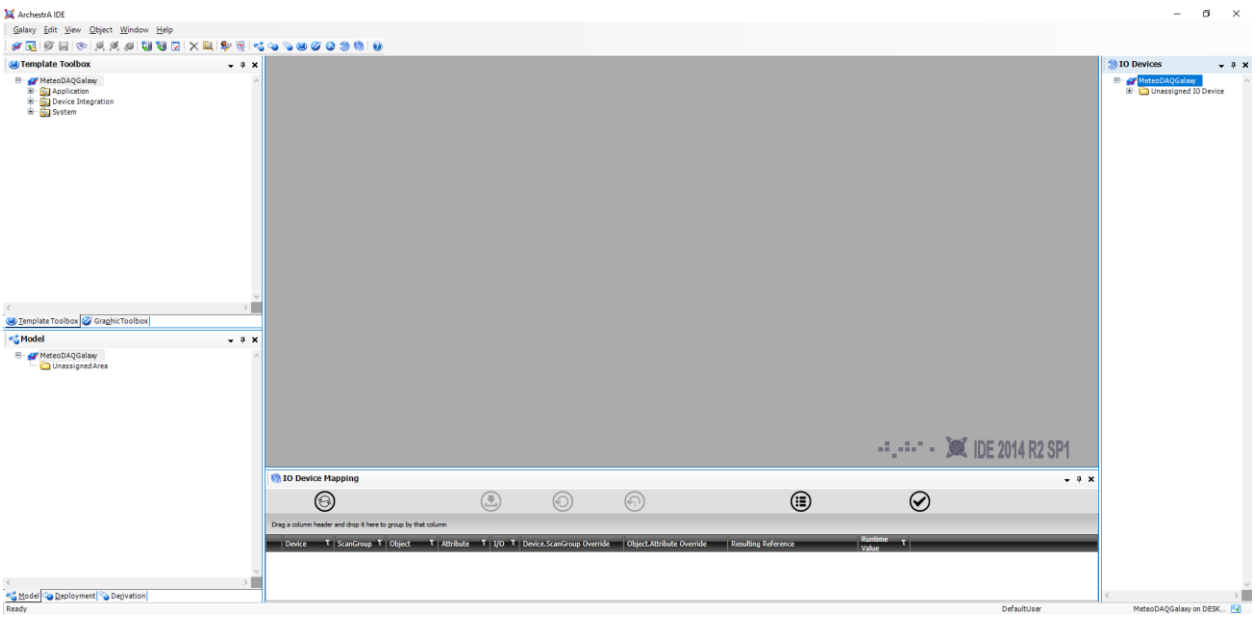

Рисунок 4.4 — Головне вікно ArchestrA IDE

4.2 Створення та налаштування об’єктів

На панелі шаблонів «Template Toolbox» створено новий набір шаблонів (toolset) MeteoToolset (рисунок 4.5) з метою логічної організації шаблонів, які будуть використовуватись в ході розробки додатку. Для відокремлення системних шаблонів та шаблонів додатку створено вкладені набори шаблонів ObjectsTemplates та SystemTemplates для зберігання шаблонів об’єктів та системних шаблонів.

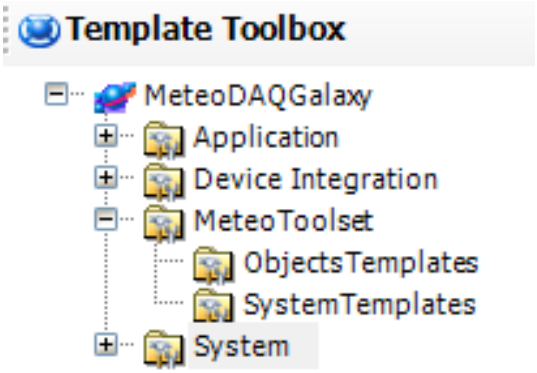

Рисунок 4.5 — Набір шаблонів MeteoToolset

Для початку розробки додатку необхідно створити шаблони, похідні від системних (рисунок 4.6).

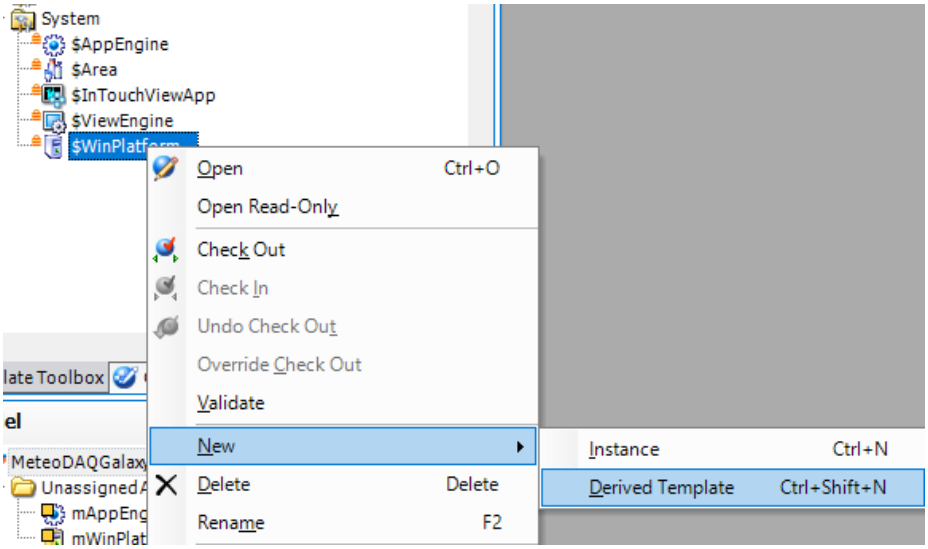

Рисунок 4.6 — Створення похідного шаблону

Таким чином створено похідні шаблони mWinPlatfor, mAppEngine та mArea (рисунок 4.7).

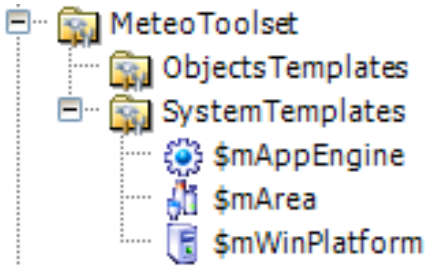

Рисунок 4.7 — Похідні шаблони системних об’єктів

Для подальшої роботи необхідно створити екземпляри системних об’єктів. Екземпляри створюються на основі шаблонів, які були створені на попередньому кроці (рисунок 4.8).

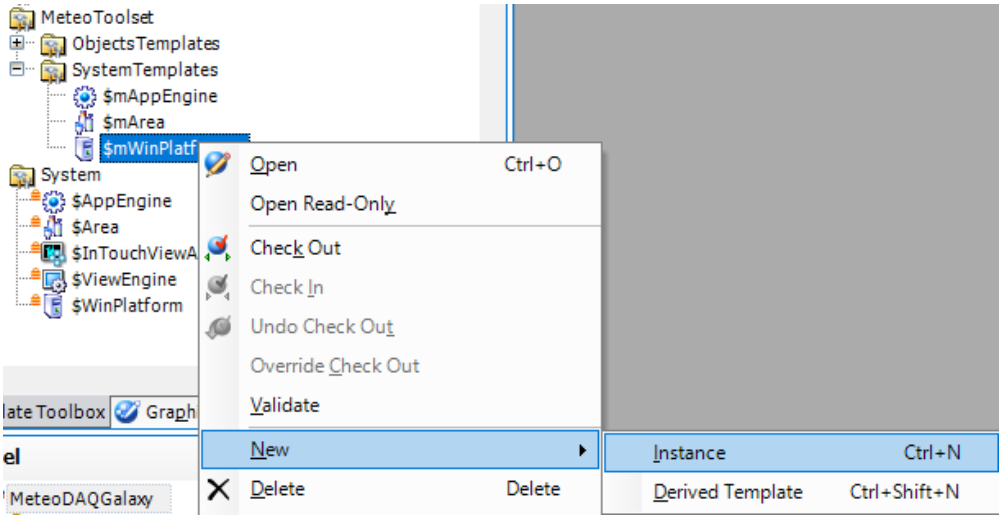

Рисунок 4.8 — Створення екземпляру об’єкту mWinPlatform

Аналогічним чином було створено екземпляри всіх системних об’єктів. Створені об’єкти можна переглянути на панелі структури додатку.

Панель структури додатку містить три вкладки:

– Model view (Загальна структура). Тут відображаються ієрархічні відносини між об’єктами у вигляді деревовидної структури. На панелі загальної структури об’єкти згруповано за зонами і за відношеннями вкладеності (рисунок 4.9).

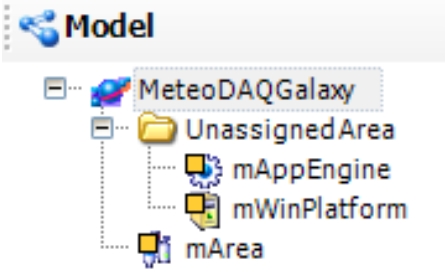

Рисунок 4.10 — Model view

– Deployment view (Структура розгортання). На цій вкладці відображено фізичне розміщення компонентів на апаратних ресурсах (комп'ютерах). Deployment view дає представлення того, як компоненти додатку ArchestrA розгортаються в системі (рисунок 4.11).

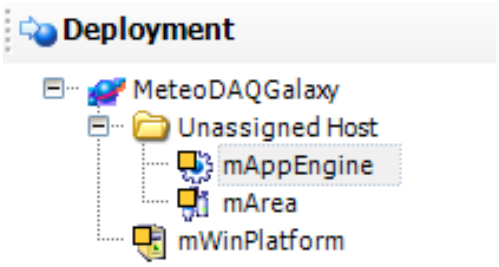

Рисунок 4.11 — Deployment view

– Derivation view (Структура наслідування). Derivation view відображає шаблони та екземпляри відповідно до їх походження, тобто відношення «батько\нащадок» між об'єктами (рисунок 4.12).

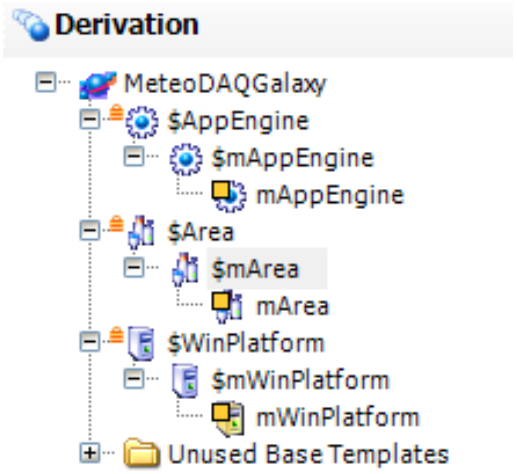

Рисунок 4.12 — Derivation view

Після створення системні об'єкти необхідно відповідним чином розмістити.

Основним є об'єкт WinPlatform. Цей об'єкт представляє одиничний комп'ютер в системі та використовується для розміщення всіх інших об'єктів. Об'єкт AppEngine розміщується безпосередньо на платформі (WinPlatform). Об'єкт Area розміщується в AppEngine. На рисунку 4.13 наведено ієрархію розміщення перелічених вище об'єктів.

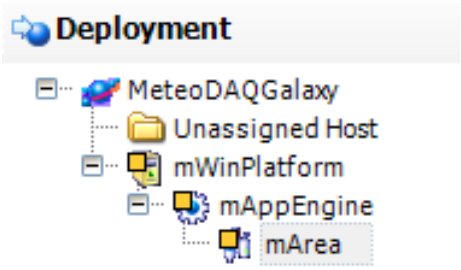

Рисунок 4.13 — Розміщення системних об’єктів

Для представлення метеостанції в Галактиці створено похідний від базового шаблону «UserDefined» шаблон «Sensor», який представляє деякий пристрій вимірювання (рисунок 4.14).

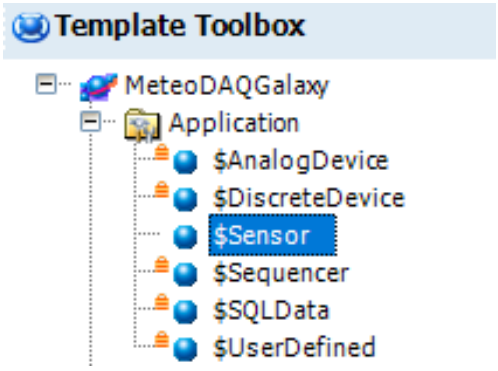

Рисунок 4.14 — Шаблон «Sensor»

Створений шаблон «Sensor» перенесено у відповідний набір шаблонів «ObjectsTemplates» (рисунок 4.15).

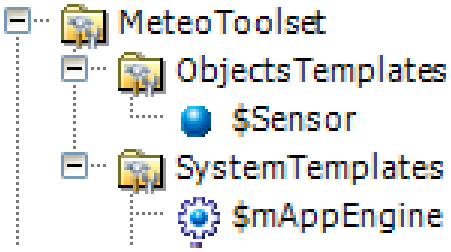

Рисунок 4.15 — Шаблон «Sensor» у наборі «ObjectsTemplates»

На основі шаблону Sensor створено екземпляр Meteostation (рисунок 4.16).

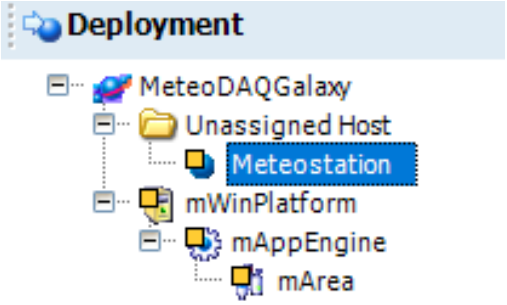

Рисунок 4.16 — Екземпляр Meteostation у Deployment view

Всі об’єкти додатку (Application Objects) повинні належати до якоїсь зони (Area), тому необхідно створений екземпляр Meteostation перенести до існуючої зони mArea (рисунок 4.17).

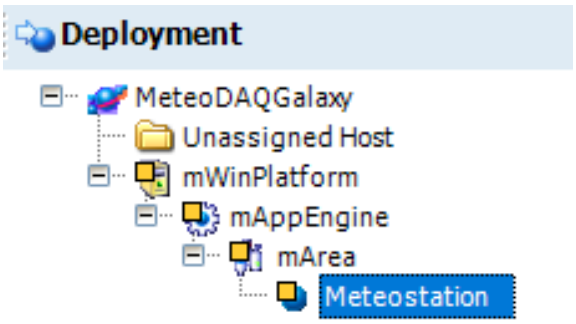

Рисунок 4.17 — Розміщення екземпляру Meteostation

4.3 Створення атрибутів об’єкта Meteostation

Для отримання даних від джерела, яке в даній роботі представлене Arduino-метеостанцією, необхідно виконати налаштування раніше створеного об’єкта Meteostation. Перш за все необхідно створити та налаштувати атрибути цього об’єкта, які б відображали ті параметри, вимірювання яких здійснює реальний пристрій.

На рисунку 4.18 зображено вікно конфігурації об’єкта Meteostation. Наразі, об’єкт ніяк не налаштований. Далі наведений опис створення атрибутів даного об’єкта.

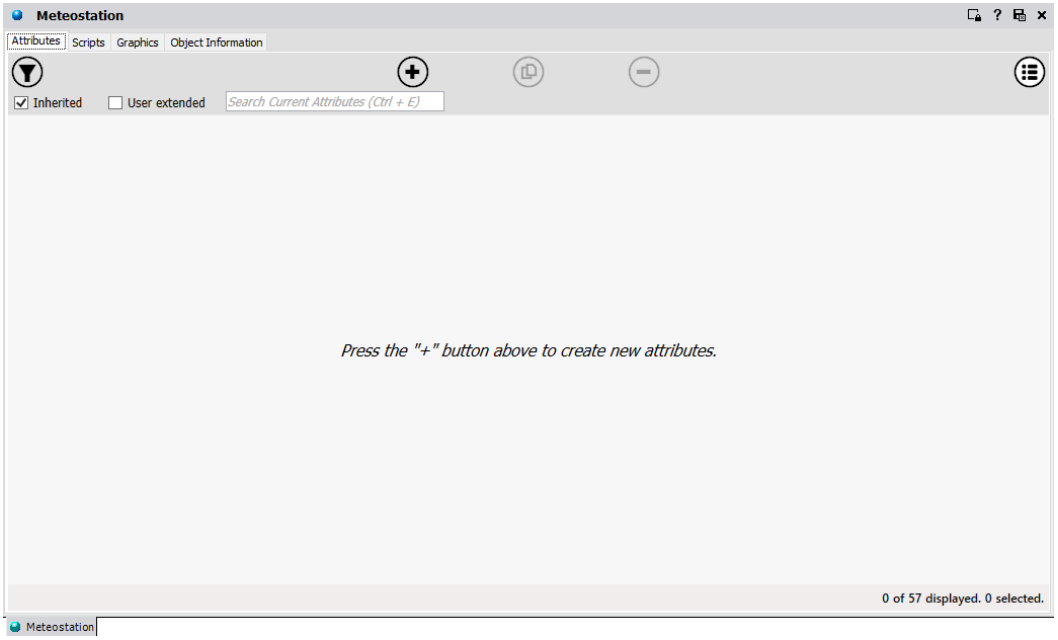

Рисунок 4.18 — Вікно конфігурації об’єкта

Метеостанція вимірює такі параметри навколишнього середовища: температура, атмосферний тиск, вологість повітря, рівень вуглекислого газу та

концентрація летких органічних сполук у повітрі, отже об’єкт Meteostation повинен мати такі ж атрибути.

Атрибут «Temperature» представляє температуру, виміряну засобами метеостанції (рисунок 4.19). Налаштування даного атрибуту виконується із такими параметрами:

- Тип даних атрибуту — число з плаваючої комою (float);
- Можливість запису — object writeable, що означає, що значення цього атрибуту може встановлювати інший об’єкт;
- Інженерні одиниці — градуси Цельсія;

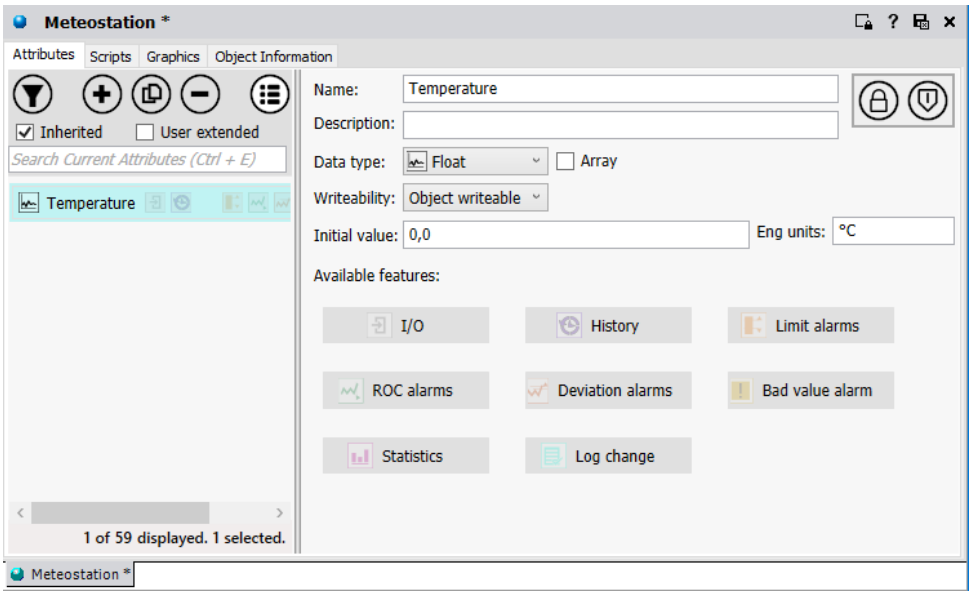

Рисунок 4.19 — Створення атрибуту «Temperature»

Наступним кроком є створення та конфігурація атрибута «Humidity», який представляє вологість повітря (рисунок 4.20). Даний атрибут має наступні характеристик:

- Тип даних атрибуту — ціле число (integer);
- Можливість запису — object writeable;
- Інженерні одиниці — відсотки;

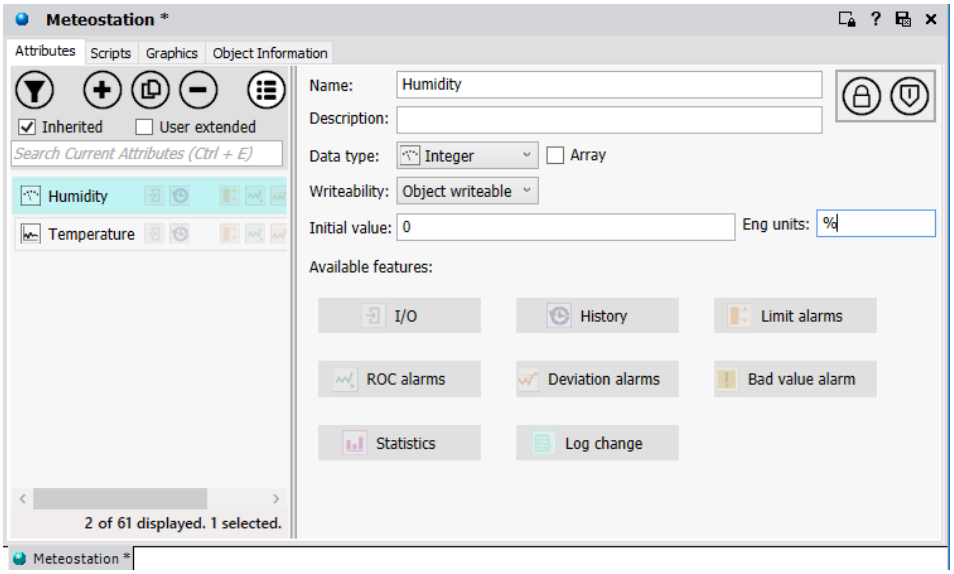

Рисунок 4.20 — Створення атрибуту «Humidity»

Атрибут «Pressure» представляє значення атмосферного тиску (рисунок 4.21).

- Тип даних атрибуту — ціле число (integer).
- Можливість запису — object writeable.
- Інженерні одиниці — міліметри ртутного стовпчика (mmHg).

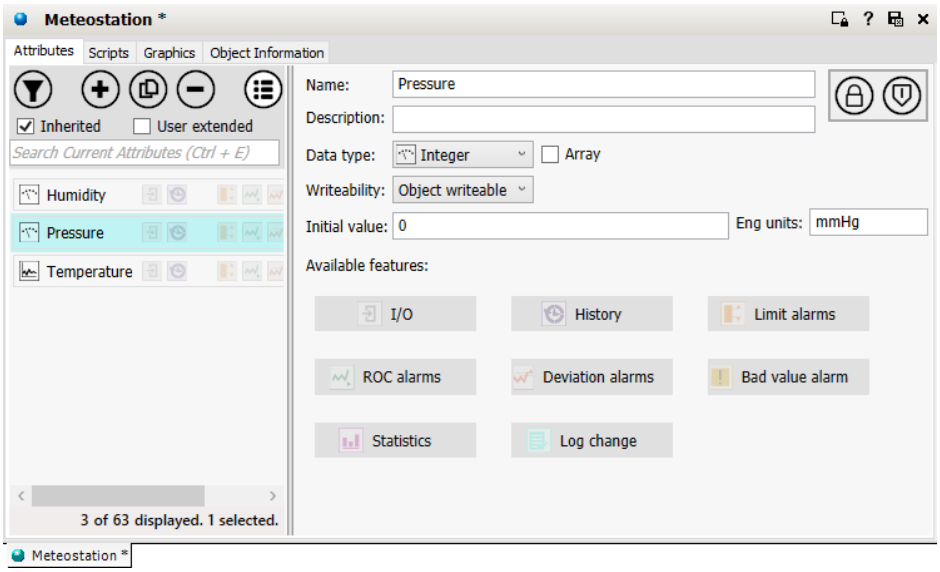

Рисунок 4.21 — Створення атрибуту «Pressure»

Атрибут «CO2» представляє значення вмісту вуглекислого газу в повітрі (рисунок 4.22) і має такі налаштування:

- Тип даних атрибуту — ціле число (integer);
- Можливість запису — object writeable;
- Інженерні одиниці — мільйонна частка (ppm).

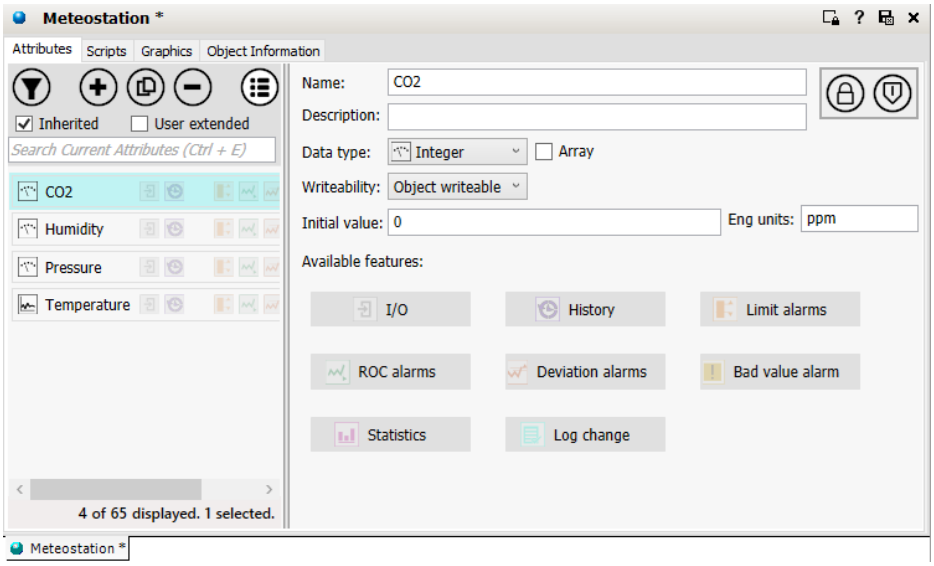

Рисунок 4.22 — Створення атрибуту «CO2»

Для отримання значення вмісту летких органічних речовин в повітрі від джерела даних створено атрибут «TVOC». На рисунку 4.23 наведено вікно конфігурації даного атрибуту. Цей атрибут налаштовано наступним чином:

- Тип даних атрибуту — ціле число (integer);
- Можливість запису — object writeable;
- Інженерні одиниці — мільярдна частка (ppb).

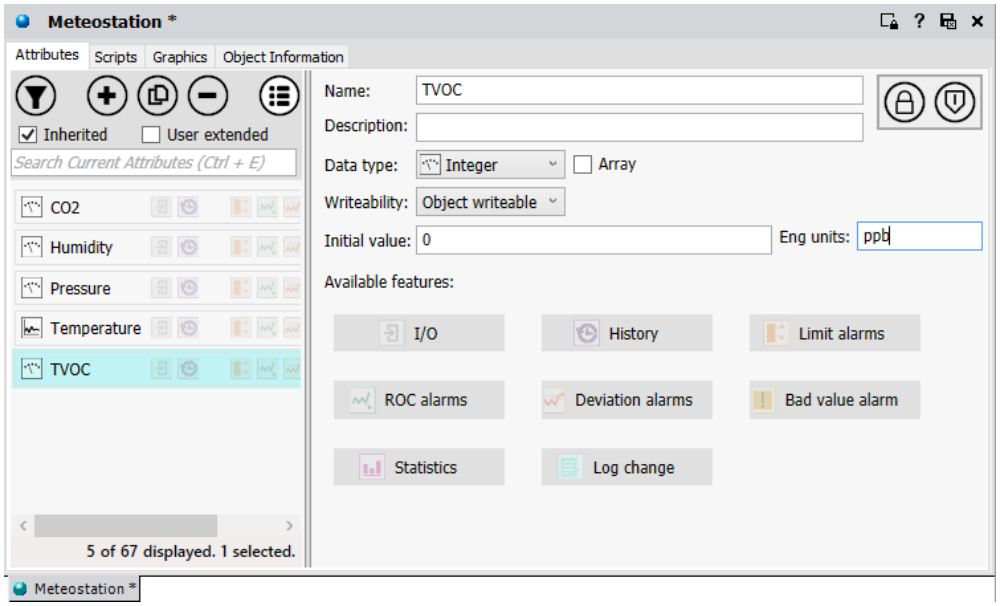

Рисунок 4.23 — Створення атрибуту «TVOC»

4.4 Конфігурація з’єднання Arduino та System Platform

Для встановлення з’єднання між додатком та зовнішнім пристроєм, в даному випадку — метеостанцією на базі Ардуіно, призначені об’єкти типу Device Integration Object (DIObject).

Device Integration Object це об’єкт, який дозволяє встановлювати з’єднання із зовнішніми пристроями. Контейнером для DIObject є AppEngine. В рамках Системної Платформи наявні два типи об’єкта DIObject: DlNetwork Objects та DlDevice Objects:

- DIDevice Object представляє реальний пристрій на кшталт ПЛК або мікроконтролера.
- DINetwork Object представляє мережеве з’єднання між додатком та реальним об’єктом або з’єднання між двома додатками.

Для організації комунікації між Arduino та Galaxy було обрано технологію OPC.

Технологія обміну даними OPC була створена як відкритий стандарт кросплатформенного програмного інтерфейсу для обміну даними. Даний стандарт визначає універсальний механізм обміну даними між клієнтами та серверами, серверами та серверами. OPC-інтерфейс виконує трансляцію вендор-специфічних команд (запитів, відповідей) у загальні OPC-запити та навпаки. Впровадження технології OPC дозволило пов’язувати між собою пристрою, сумісність яких ніколи не перевірялась, що в свою чергу уможливило використання виробів різних вендорів в одній системі [59].

Технологія OPC передбачає використання клієнт-серверного підходу до організації взаємодії між пристроями. OPC-сервер це програма, яка отримує дані від деякого пристрою (наприклад ПКЛ) по специфічному для даного пристрою протоколу і виконує трансляцію в стандартний формат, передбачений стандартом. OPC-клієнтом є програма (програмний модуль), який здійснює запити на отримання даних від OPC-сервера [60].

Найбільш широкого застосування в сфері автоматизації набув стандарт OPC Data Access (OPC DA), який визначає інтерфейс доступу до даних реального часу, які генеруються пристроями вимірювання, та передачу управляючих команд від системи управляння до виконавчих пристроїв.

Як було зазначено вище, для організації комунікації із зовнішніми пристроями в System Platform передбачені об’єкти інтеграції пристроїв (Device Integration Object). Для отримання даних від джерела скористаємось вбудованим Device Integration шаблоном — OPCClient. Як можна зрозуміти із назви, даний об’єкт є OPC-клієнтом та дозволяє отримувати дані від OPC-сервера.

На рисунку 4.24 наведено схему того, як дані від джерела передаються в додаток Galaxy по OPC.

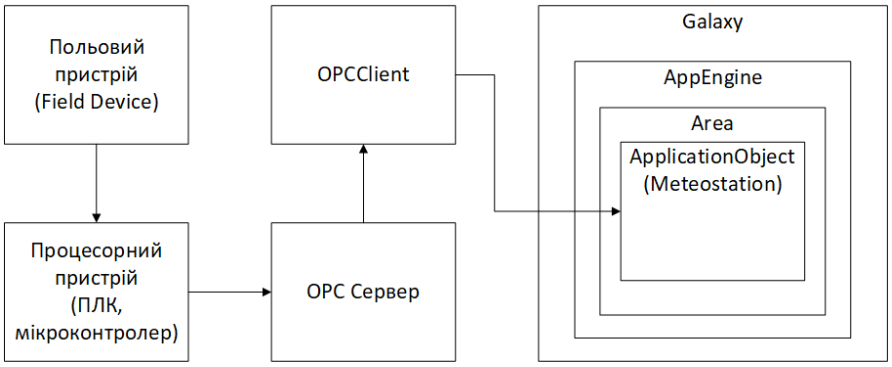

Рисунок 4.24 — Тракт даних

Для реалізації комунікації метеостанції та додатку Galaxy перш за все, необхідно створити новий похідний шаблон від базового шаблону OPCClient. Похідний шаблон названо — mOPCClient (рисунок 4.25)

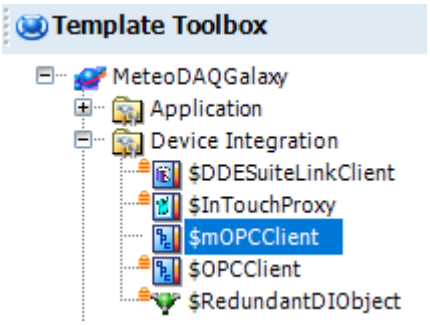

Рисунок 4.25 — Шаблон mOPCClient

Далі створено екземпляр об’єкта mOPCClient (рисунок 4.26).

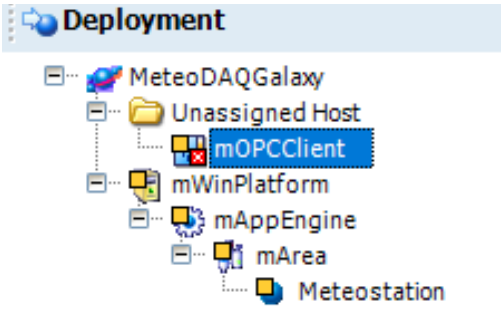

Рисунок 4.26 — Екземпляр об’єкта mOPCClient

Наступним кроком є присвоєння об’єкта OPCClient існуючому об’єкту WinPlatform (рисунок 4.27).

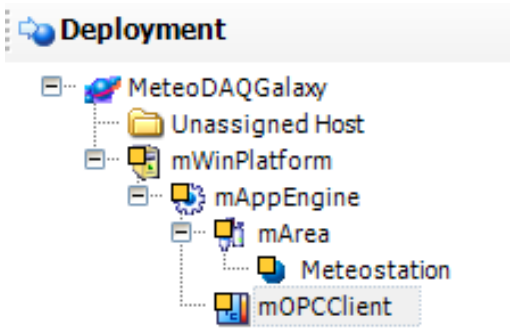

Рисунок 4.27 — Об’єкт mOPCClient

При конфігурації об’єкта типу OPCClient необхідно вказати ім’я вузла на якому запущено OPC-сервер та ім’я OPC-сервера. На рисунку 4.28 наведено параметри конфігурації об’єкта mOPCClient. Параметр «Server node» визначає комп’ютер, на якому запущено цільовий OPC сервер. Так як OPC сервер та додаток Galaxy розташовані на одній машині, то значення параметр «Server node» можна вказати як localhost. Параметр «Server name» визначає ім’я цільового OPC сервера. Galaxy автоматично визначає запущені на локальній машині сервери, та відображає їх у випадаючому списку.

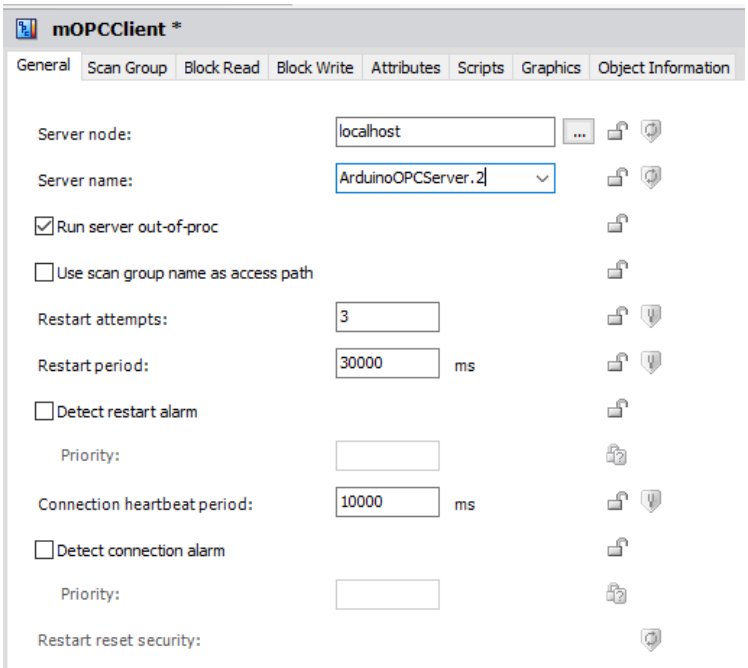

Рисунок 4.28 — Конфігурація об’єкта mOPCClient

Зв’язування реального пристрою та об’єкта System Platform здійснюється шляхом розміщення цього об’єкта у скан-групі.

Скан-група це певний набір тегів (атрибутів), значення яких надходять від одного джерела даних. Так як значення атрибутів об’єкта Meteostation будуть в подальшому отримуватись від одного джерела — ардуіно - метеостанції, то в проекті застосовано лише одну скан-групу MeteoScan (рисунок 4.29).

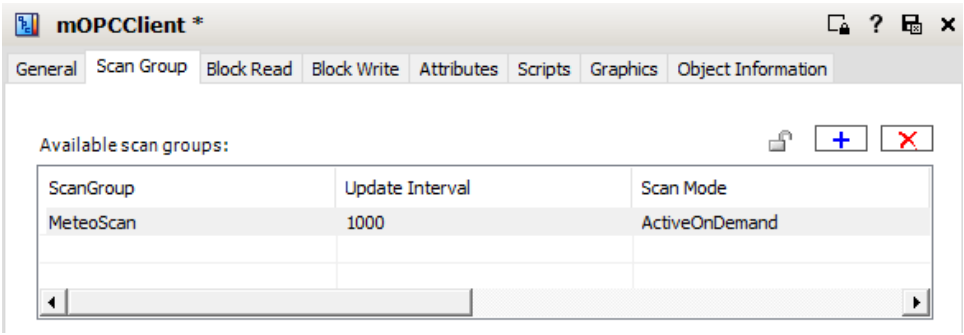

Рисунок 4.29 — Скан-група MeteoScan

На панелі IO Devices (Пристрої введення-виведення) відображаються всі системні об’єкти (system objects) та об’єкти додатку (application objects), а також Device Integration об’єкти, із якими зв’язані користувацькі групи сканування. За відсутності попередніх налаштувань, на панелі пристроїв введення-виведення всі об’єкти знаходяться в каталозі «Unassigned IO Devices» (рисунок 4.30).

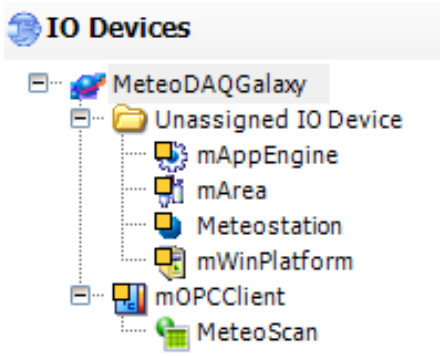

Рисунок 4.30 — Пристрої введення-виведення

Для того, щоб закріпити об’єкт за групою сканування необхідно перетягти його до відповідної групи. Об’єкт Meteostation було вкладено до групи сканування MeteoScan (рисунок 4.31).

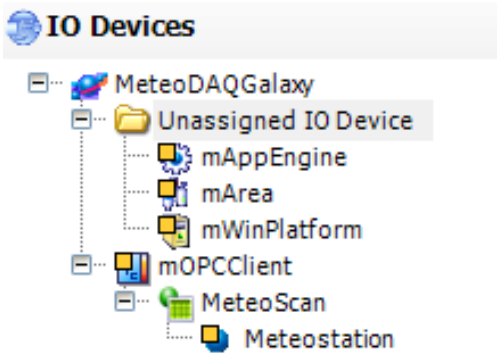

Рисунок 4.31 — Присвоєння об’єкта скан-групі

Для отримання даних від метеостанції через OPC було застосовано вільно розповсюджуваний OPC-сервер для Ардуіно. Інтерфейс програми-сервера зображено на рисунку 4.32.

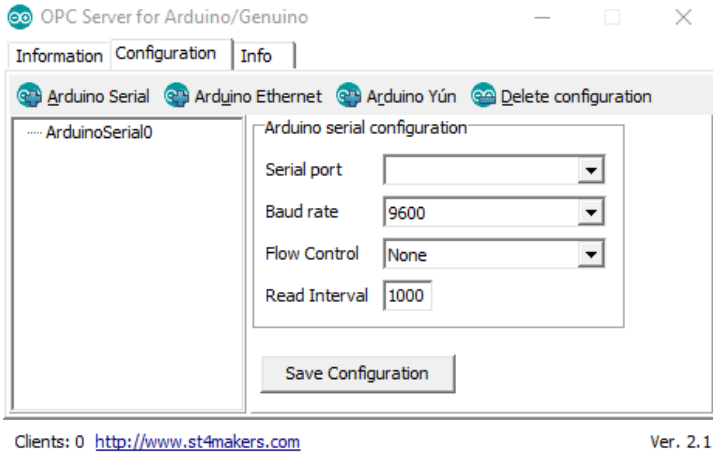

Рисунок 4.32 — Інтерфейс OPC-сервера

При налаштуванні Arduino OPC сервера необхідно вказати порт, до якого підключено Arduino, символьну швидкість та інтервал читання даних від джерела (в мілісекундах). На рисунку 4.33 наведено значення, задані для перелічених вище параметрів.

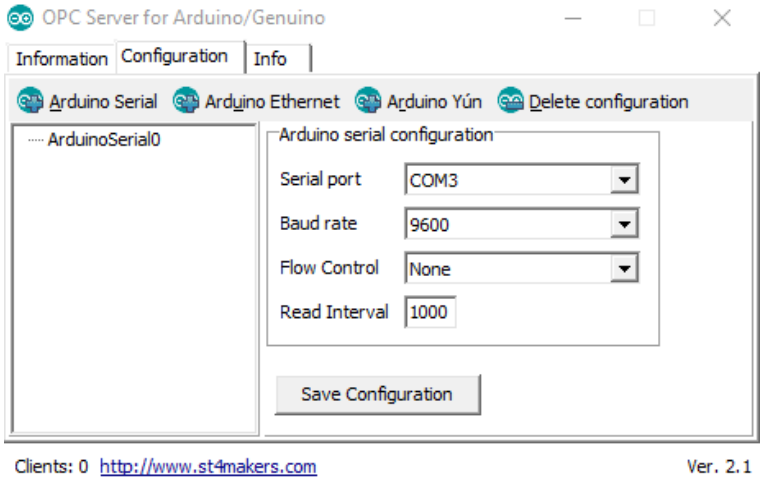

Рисунок 4.33 — Параметри OPC-сервера

4.6 Налаштування параметрів атрибутів об’єкта Meteostation

Для того, щоб значення атрибутів об’єкта отримувались від OPC сервера необхідно налаштувати відповідний параметр атрибутів – I/O (введення-виведення). Так як дані будуть лише зчитуватись, то даний параметр налаштовано в режим «лише читання». Далі необхідно вказати джерело даних. В полі «Read from» необхідно вказати посилання на джерело, яке включає ім’я DI об’єкта (у даному випадку це mOPCClient), ім’я скан-групи, ім’я з’єднання із Arduino (ArduinoSerial0) та ім’я параметра значення якого буде присвоєно відповідному атрибуту. Отже, шаблон для створення посилання буде виглядати наступним чином: <DI Object.ScanGroup.Object.Attribute>. Важливо зауважити, що Attribute – це ім’я змінної у скетчі Ардуіно, значення якої буде присвоєно атрибуту, для якого задано посилання на цю змінну.

На рисунках 4.34 – 4.38 наведено налаштування параметру I/O для всіх атрибутів.

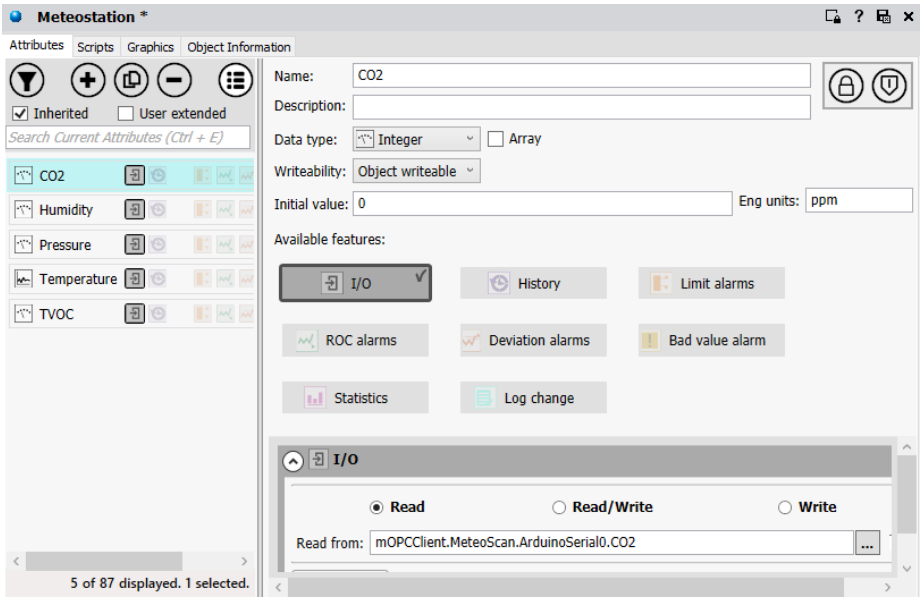

Рисунок 4.34 — Налаштування параметру I/O для атрибуту CO2

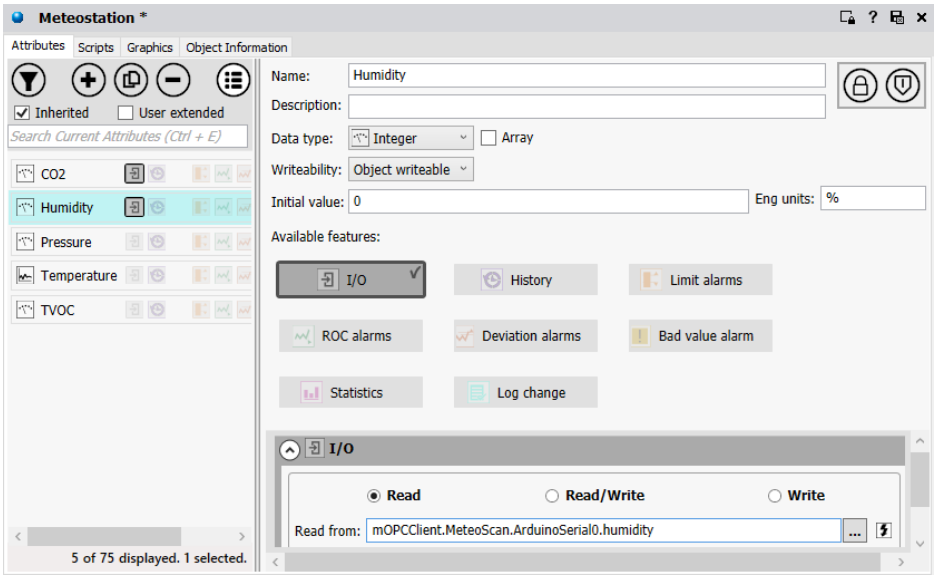

Рисунок 4.35 — Налаштування параметру I/O для атрибуту Humidity

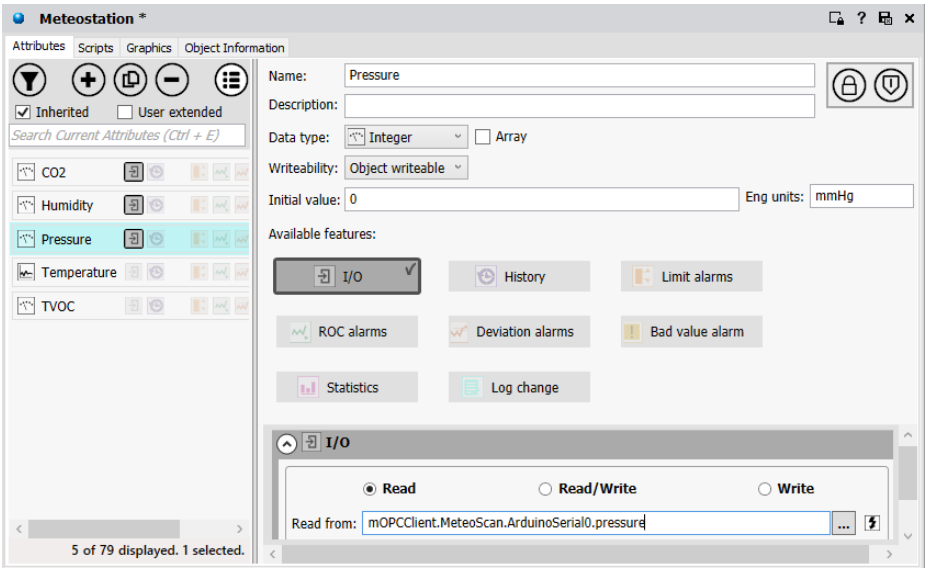

Рисунок 4.36 — Налаштування параметру I/O для атрибуту Pressure

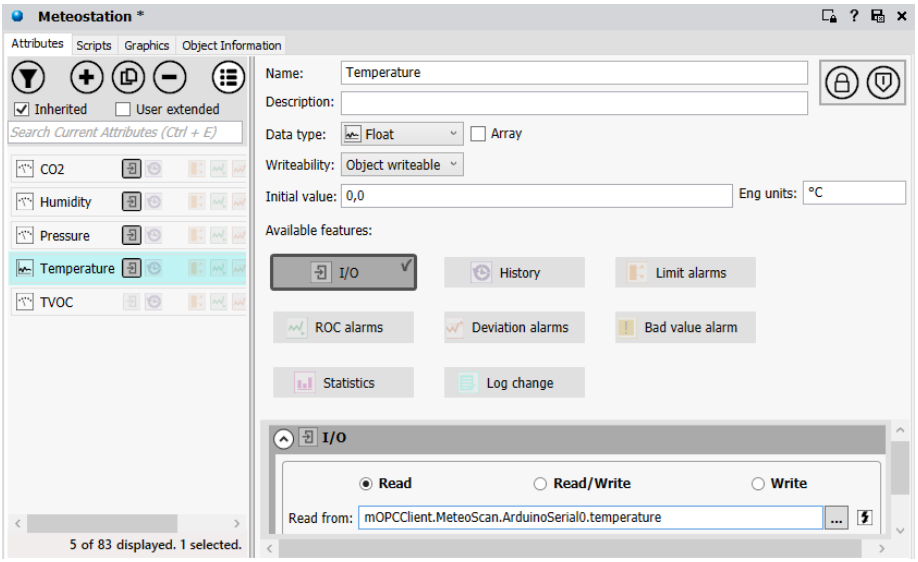

Рисунок 4.37 — Налаштування параметру I/O для атрибуту Temperature

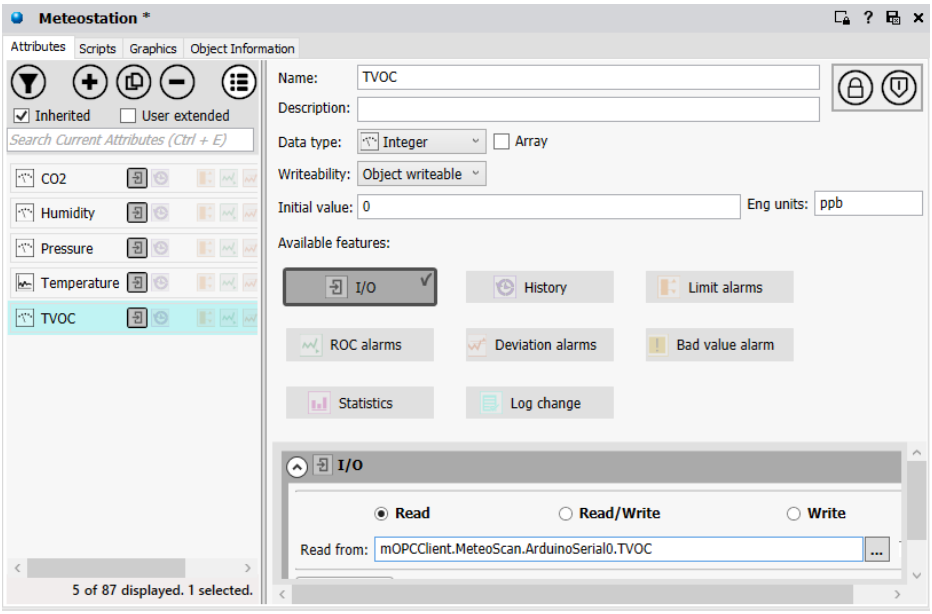

Рисунок 4.38 — Налаштування параметру I/O для атрибуту TVOC

Після конфігурації всіх об’єктів необхідно виконати їх розміщення у середовищі виконання, тобто виконати «deploy» (рисунок 4.39). Процес деплою полягає у копіюванні об’єктів із середовища розробки до середовища виконання, де виконується логіку, закладена в об’єктах.

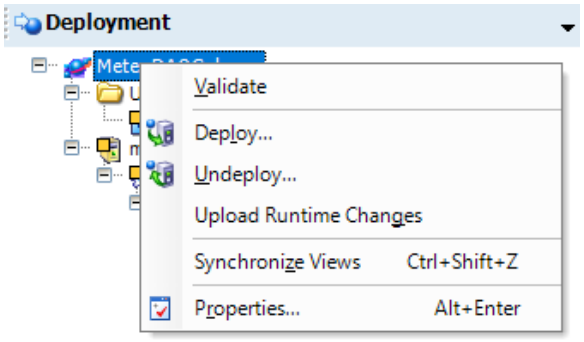

Рисунок 4.39 — Deploy

У меню деплою активовано опцію каскадного розміщення. Каскадне розміщення об’єктів дозволяє виконати деплой відразу усіх об’єктів в правильному порядку (рисунок 4.40).

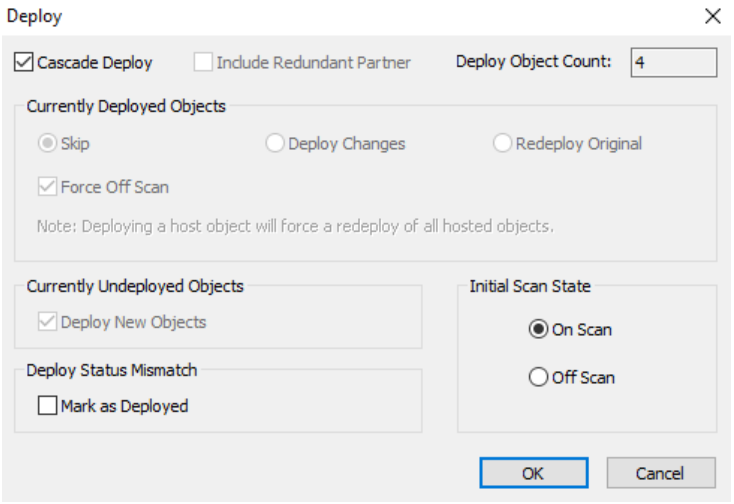

Рисунок 4.40 — Параметри деплою

На рисунку 4.41 показано хід виконання розміщення об’єктів у середовищі виконання ArchestrA.

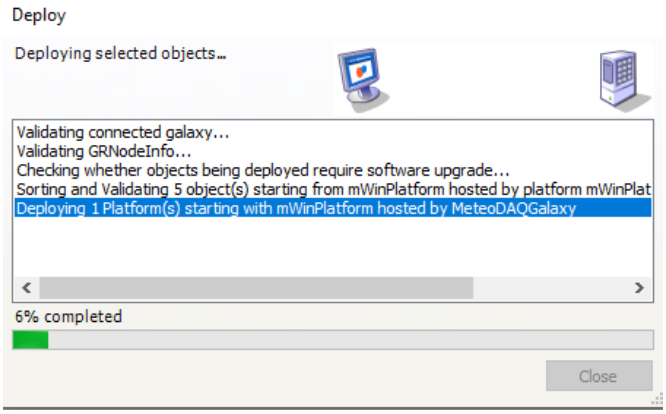

Рисунок 4.42 — Процес деплою

Після того як процес деплою завершено, з'являється відповідне повідомлення про успішне завершення розміщення (рисунок 4.43), отже всі попередні дії було виконано без помилок.

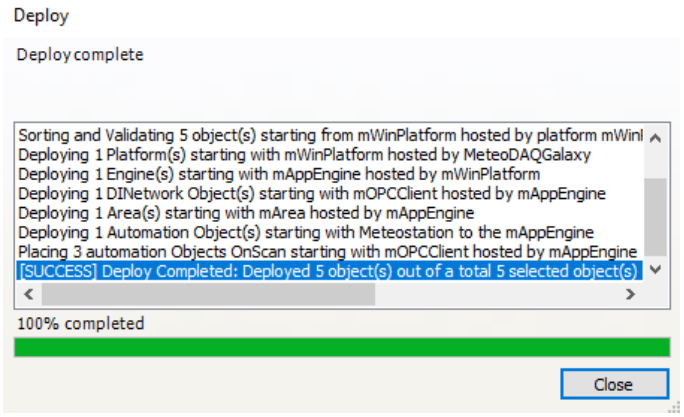

Рисунок 4.43 — Повідомлення про успішний деплой

Після того як об'єкти було перенесено до середовища виконання, можна спостерігати за параметрами об'єкта Meteostation, значення яких надходять від від OPC сервера. Спостерігати за параметрами об'єкта Meteostation можна через Object Viewer (рисунок 4.44).

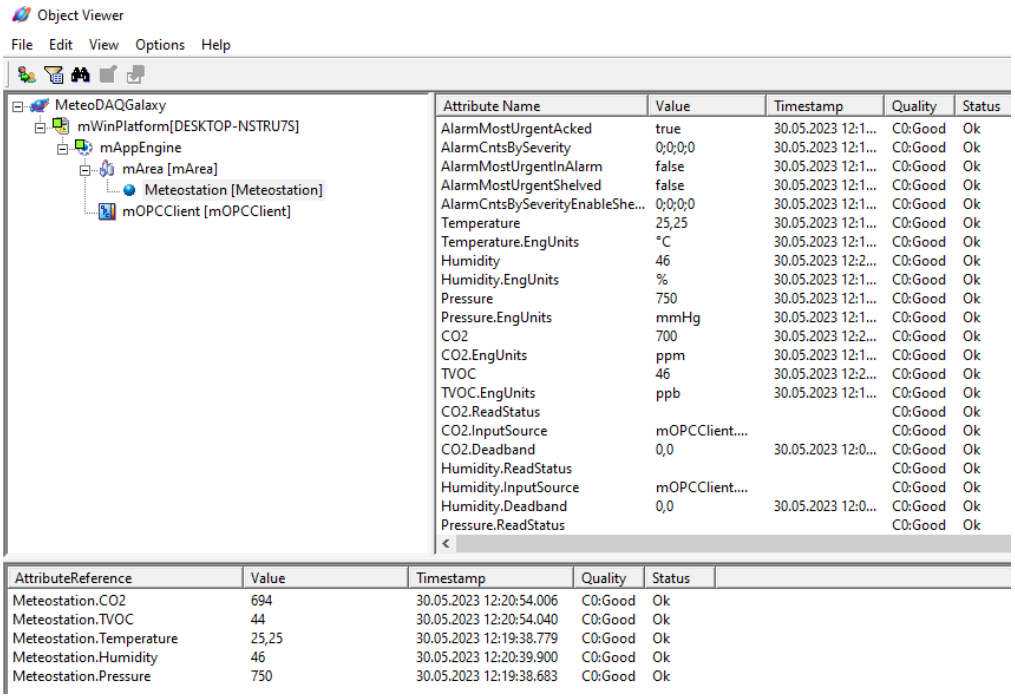

Рисунок 4.44— Object Viewer

4.6 Створення HMI

В рамках Системної Платформи засобом створення людино-машинного інтерфейсу (HMI) є програмне забезпечення InTouch.

Додаток InTouch може бути створений двома способами – як окремий додаток через InTouch HMI Application Manager або через середовище розробки ArchestrA IDE. В цьому проекті додаток візуалізації було створено

через ArchestrA IDE. На рисунку 4.45 показано структуру інтеграції ArchestrA IDE та InTouch HMI компонентів.

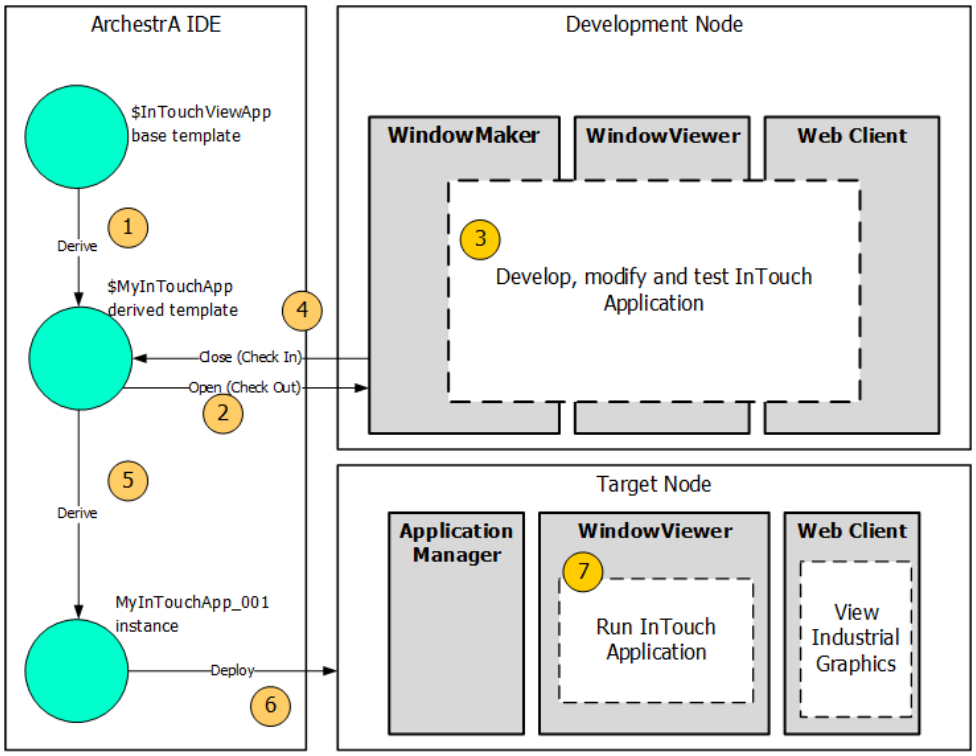

Рисунок 4.45 — Структуру інтеграції ArchestrA IDE та InTouch HMI

Сервер додатків здійснює керування додатками InTouch за допомогою спеціального об’єкта, який називається InTouchViewApp. $InTouchViewApp є базовим системним шаблоном (рисунок 4.46)

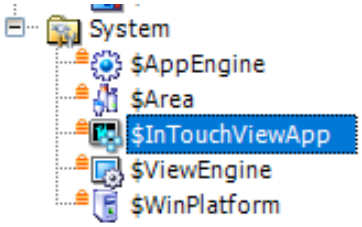

Рисунок 4.46 — Шаблон $InTouchViewApp на панелі шаблонів

Першим кроком при створенні InTouch HMI додатку із середовища розробки ArchestrA є створення похідного шаблону від $InTouchViewApp. Похідний шаблон названо MeteoView (рисунок 4.47).

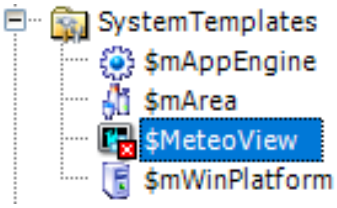

Рисунок 4.47 — Шаблон MeteoView

При першому відкритті об’єкта MeteoView з’являється вікно, де пропонується створити новий додаток InTouch або виконати експорт вже існуючого додатку (рисунок 4.48). Обрано опцію створення нового додатку.

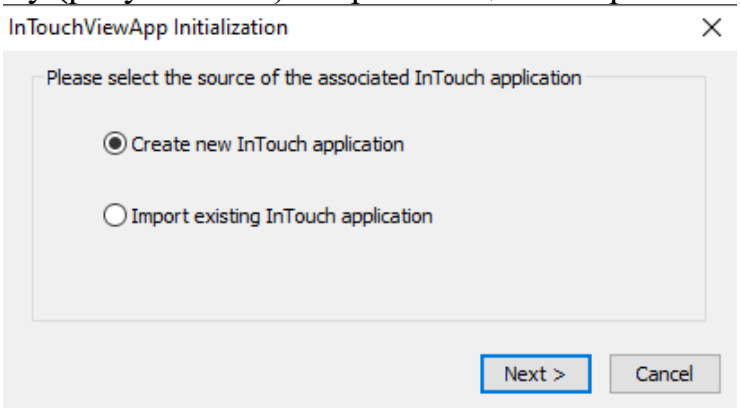

Рисунок 4.48 — Створення нового додатку

Далі необхідно вказати ім’я додатку та підтвердити створення нового додатку (рисунок 4.49).

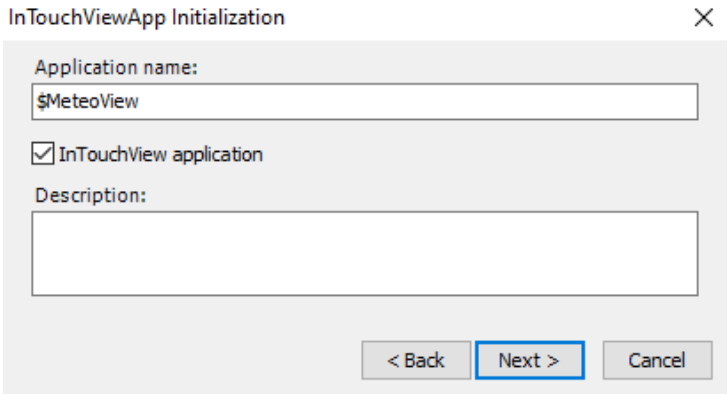

Рисунок 4.49 — Ім’я додатку візуалізації

Після створення нового додатку InTouch відкривається вікно редактору InTouch WindowMaker (рисунок 4.50).

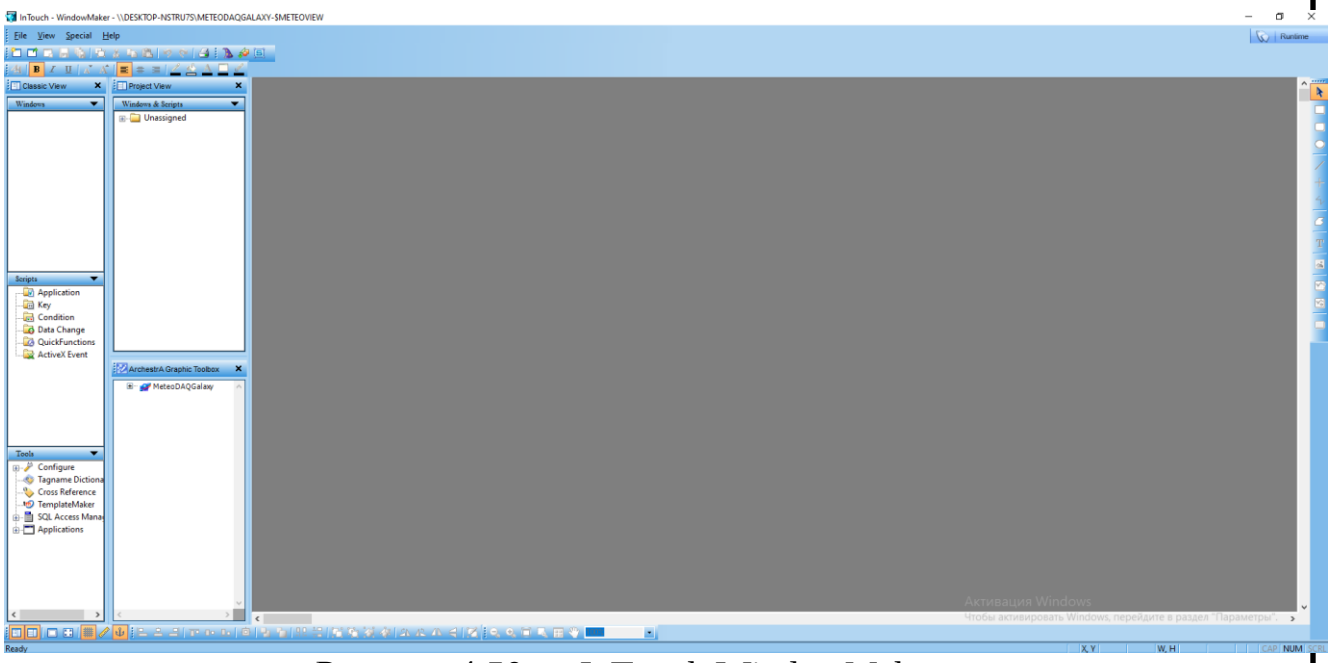

Рисунок 4.50 — InTouch WindowMaker

Першим кроком при розробці додатку візуалізаціє є створення нового вікна, де і будуть відображатись графічні об’єкти. На рисунку 4.51 наведено параметри створюваного вікна.

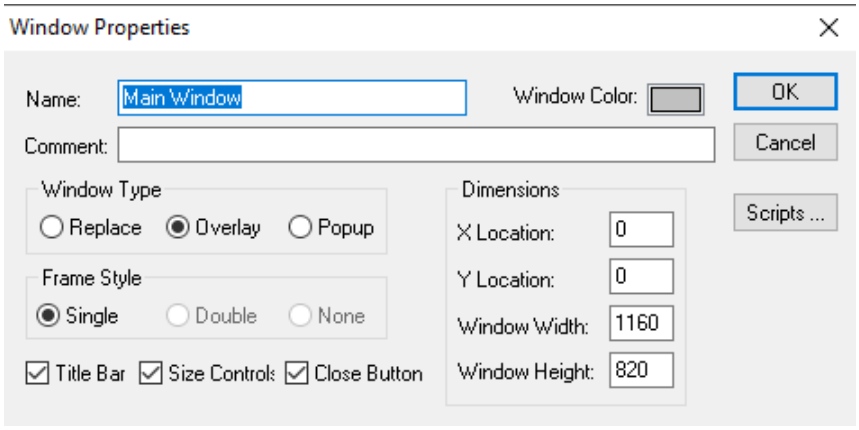

Рисунок 4.51 — Параметри створюваного вікна

Створене вікно тепер відображається в редакторі WindowMaker (рисунок 4.52)

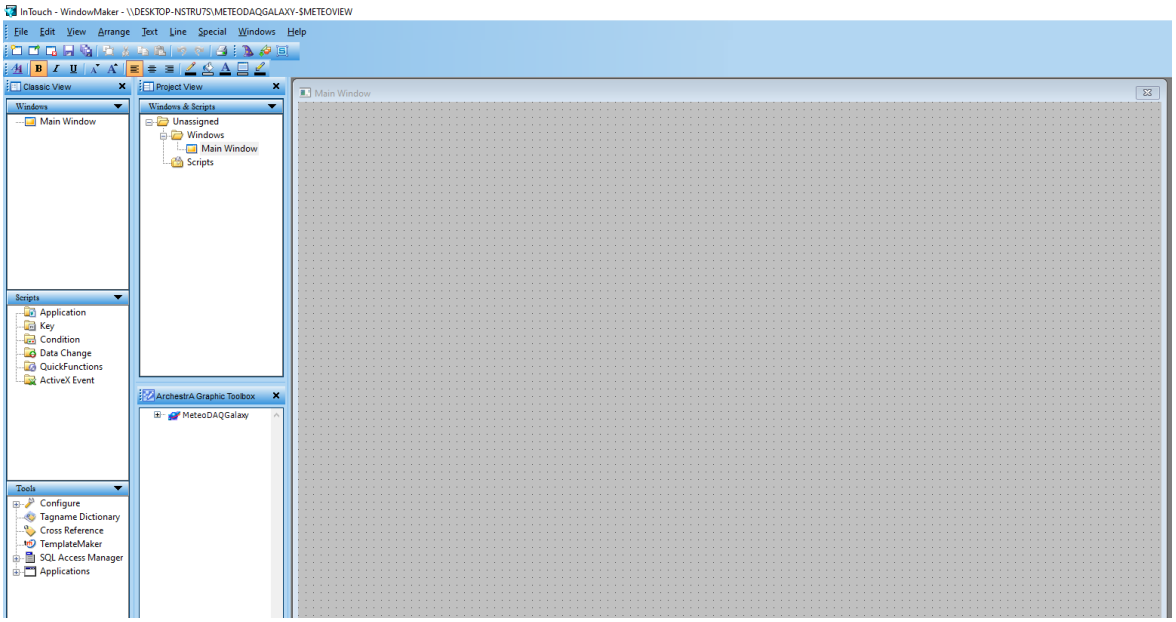

Рисунок 4.52 — Нове вікно

У вікні конфігурації об’єкта Meteostation обрано вкладку «Graphics» та створено новий символ «tempView», який буде відображати значення температури (рисунок 4.53). Такий спосіб створення графічного символу дозволяє автоматично зв’язати цей символ та даний екземпляр.

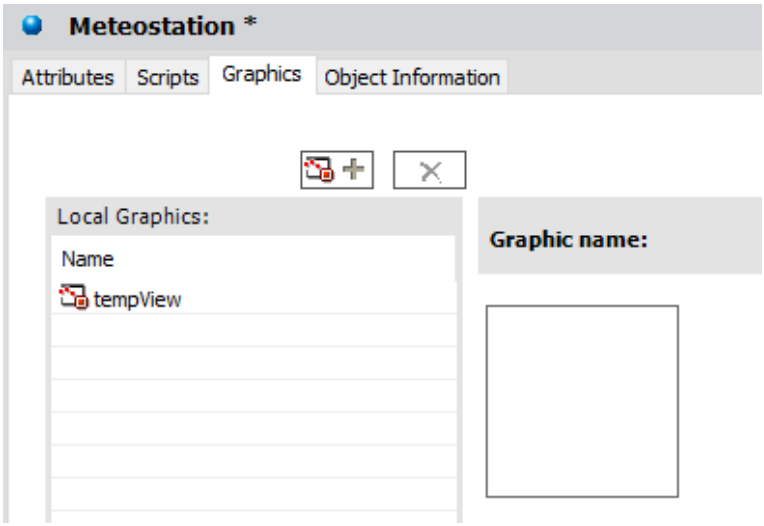

Рисунок 4.53 — Вкладка «Graphics»

Відкриття даного компоненту графіки призводить до запуску векторного графічного редактора Industrial Graphic Editor (рисунок 4.54).

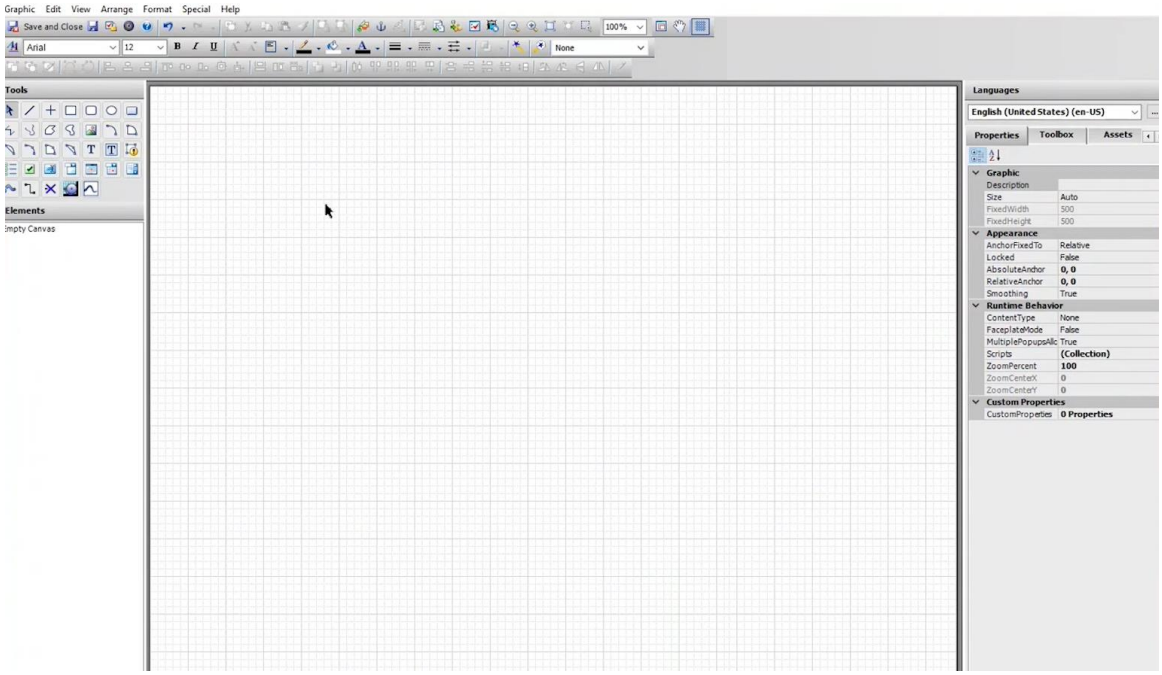

Рисунок 4.54 — Industrial Graphic Editor

Щоб додати новий символ необхідно на панелі інструментів обрати Embed ArchestrA Symbol, після чого відкривається вікно (Galaxy Browser) із вбудованими символами (рисунок 4.55).

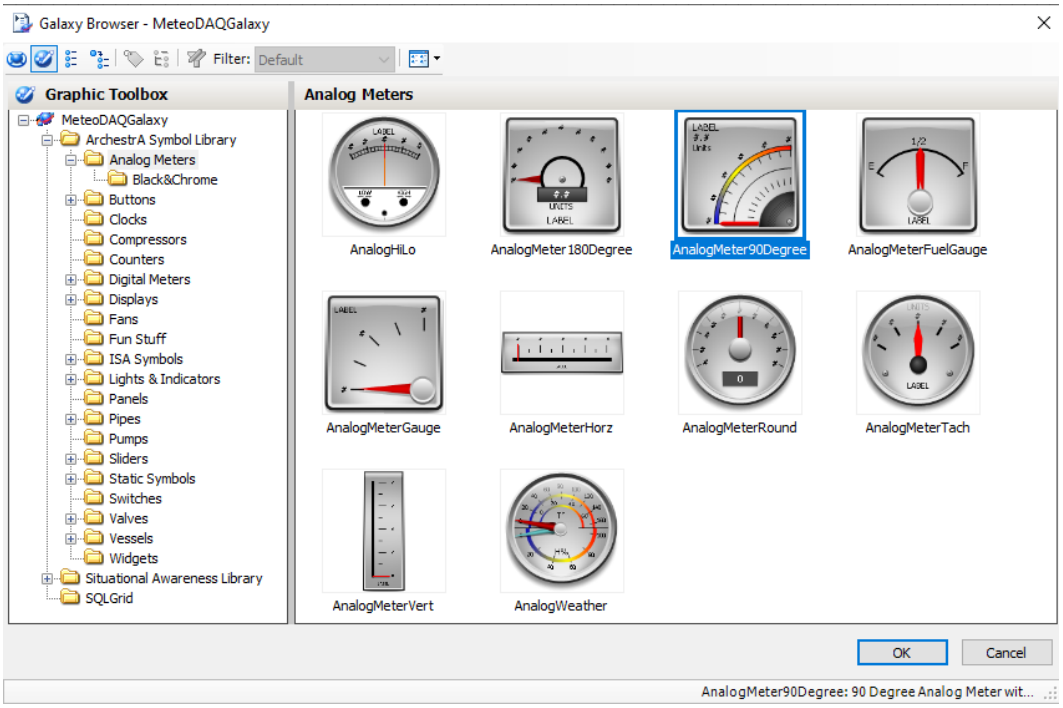

Рисунок 4.55 — Galaxy Browser

Для відображення температури було обрано символ, зображений на рисунку 4.56.

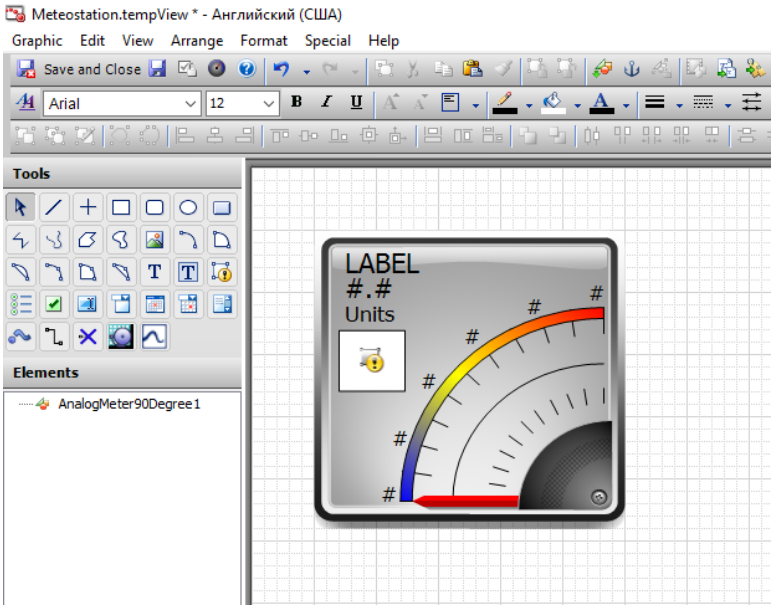

Рисунок 4.56 — Символ відображення температури

Текстові параметри даного графічного об’єкта було замінено на такі, які відповідають характеру відображуваних значень (рисунок 4.57).

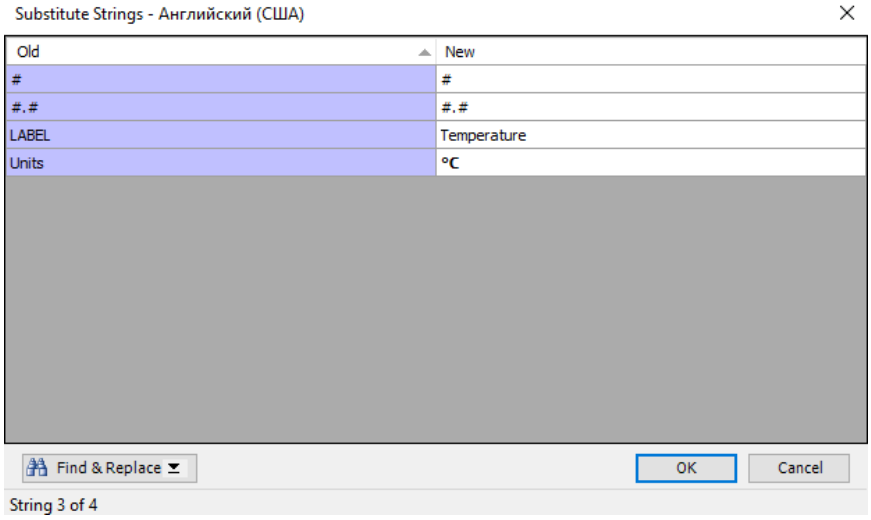

Рисунок 4.57 — Текстові параметри елемента

Задано найменування відображуваної величини — Temperature, одиниці вимірювання задані як градус Цельсія.

Для того, щоб графічний елемент відображував значення атрибуту об’єкта необхідно вказати який значення якого саме параметру повинен відображати даний символ. Для цього необхідно перейти до користувацьких налаштувань (рисунок 4.58). Тут необхідно задати граничні значення, які буде відображати цей графічний елемент, та значення для відображення (Default value).

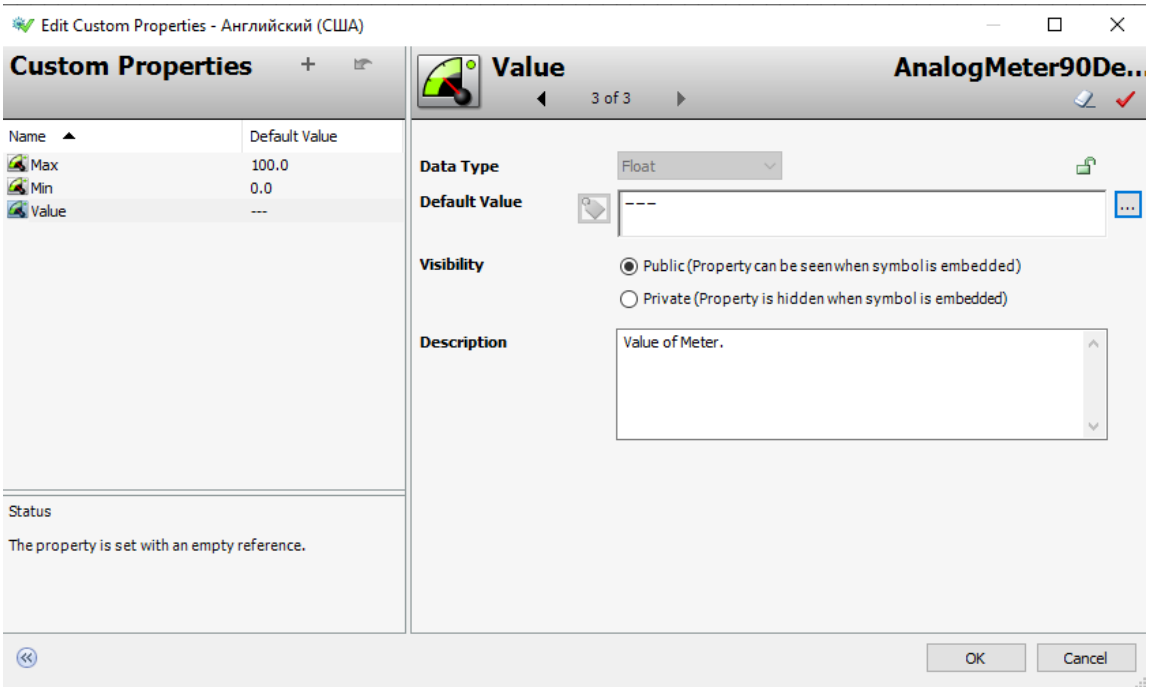

Рисунок 4.58 — Вікно користувацьких налаштувань

Значення для відображення задається через Galaxy Browser, де необхідно обрати той атрибут об’єкта, значення якого необхідно відобразити. На рисунку 4.59 показано вибір атрибута Temperature об’єкта Meteostation.

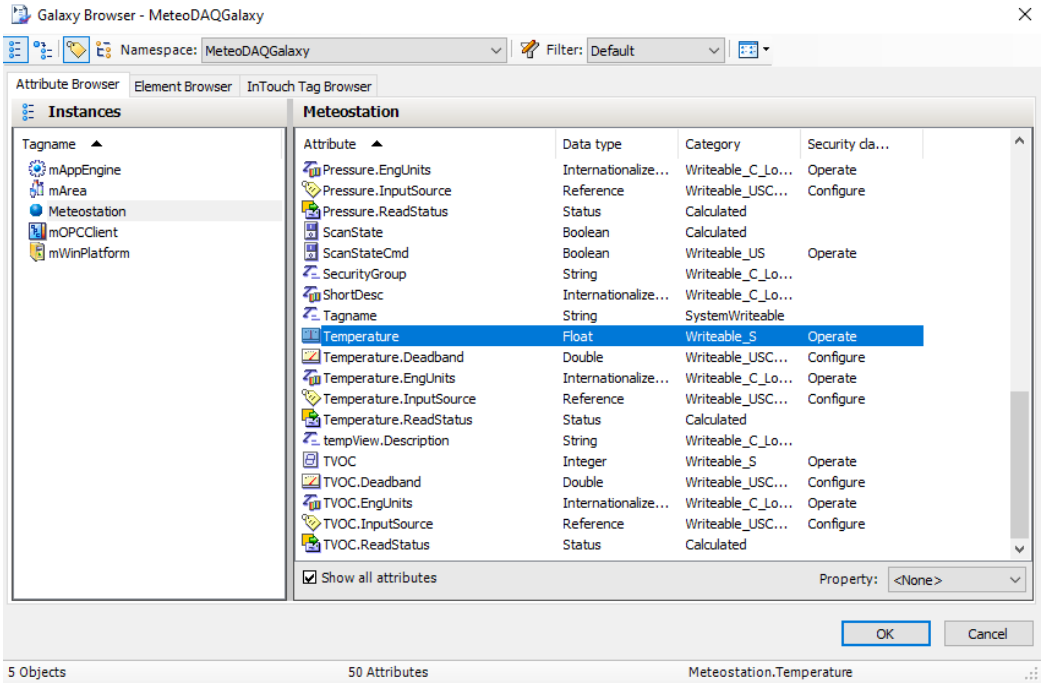

Рисунок 4.59 — Вибір атрибута для відображення

На рисунку 4.60 наведено параметри символу TempView.

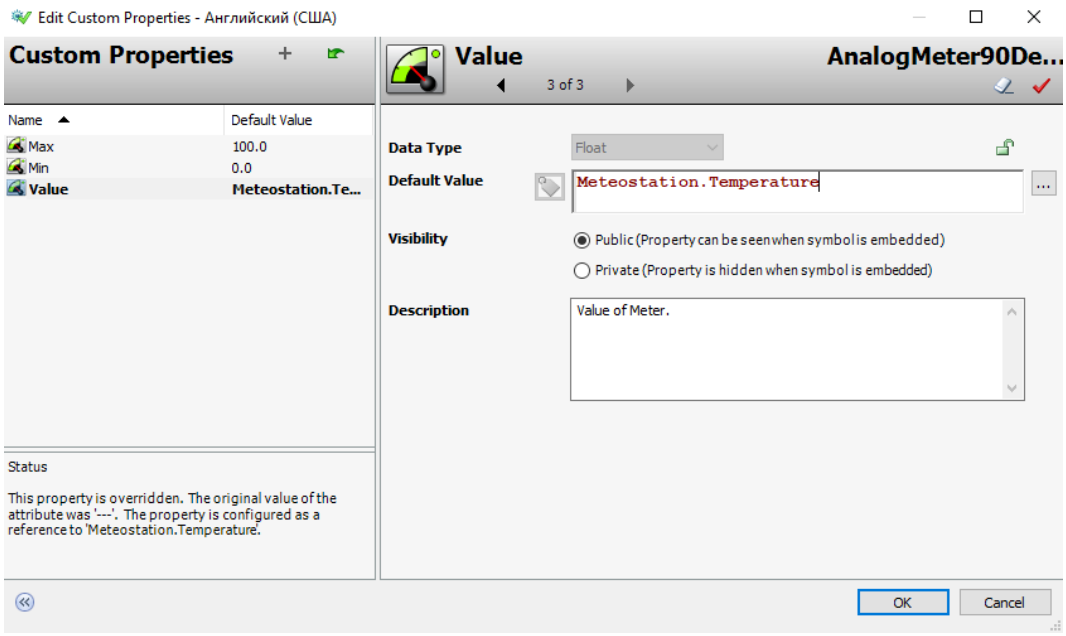

Рисунок 4.60 — Параметри графічного об’єкта

Після конфігурації графічного об’єкта в редакторі графіки можна повернутись у вікно редактору InTouch WindowMaker та додати створений щойно графічний символ у вікні. На рисунку 4.61 показано вікно із доступними для об’єкта Meteostation графічними об’єктами.

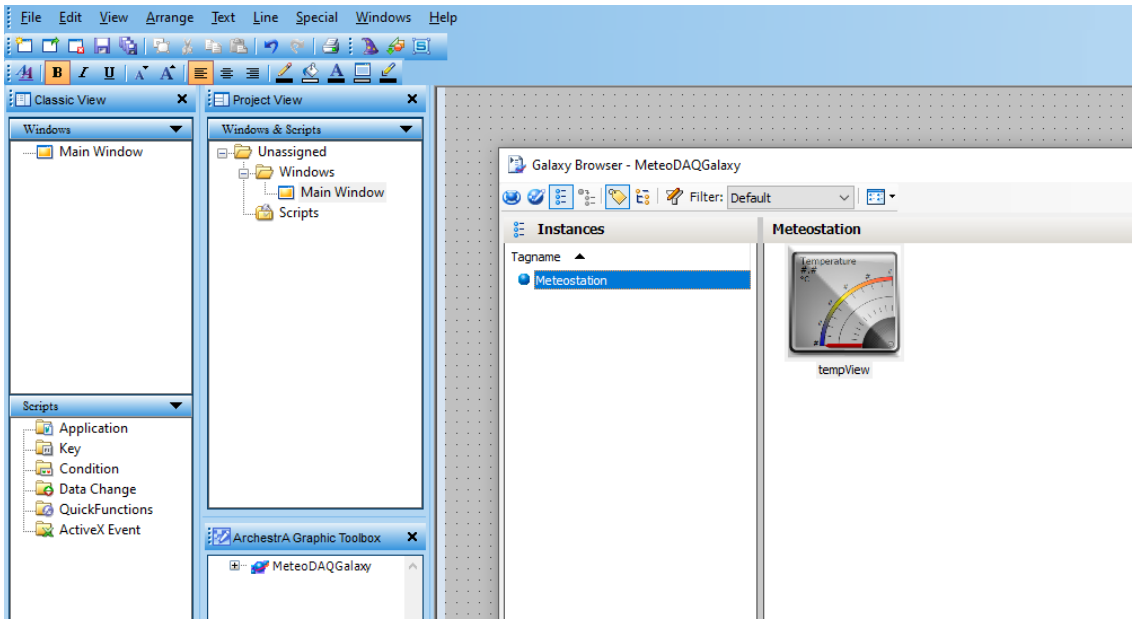

Рисунок 4.61 — Графічні символи для об’єкта Meteostation

На поле вікна було додано розроблений раніше графічний об’єкт tempView (рисунок 4.62).

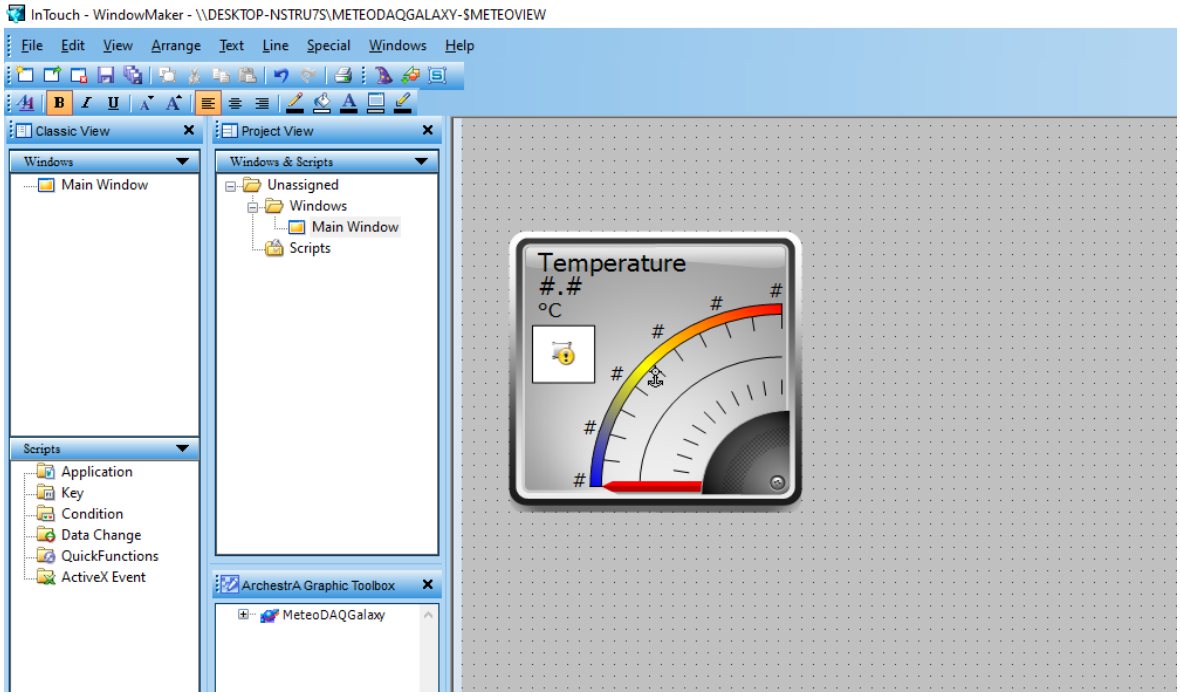

Рисунок 4.62 — tempView

На рисунку 4.63 зображено як виглядає відображення значення температури в режимі попереднього перегляду.

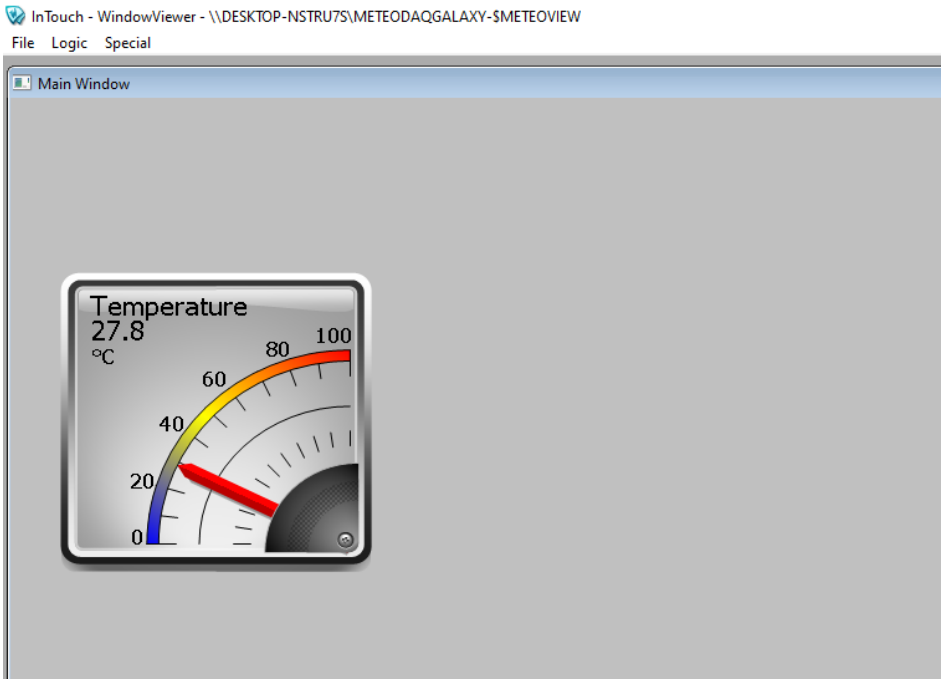

Рисунок 4.63 — Відображення температури

Наступним атрибутом для якого створюється графічне відображення є значення атмосферного тиску. У вікні конфігурації об’єкта Meteostation на вкладці Graphics створено новий символ – pressureView (рисунок 4.64).

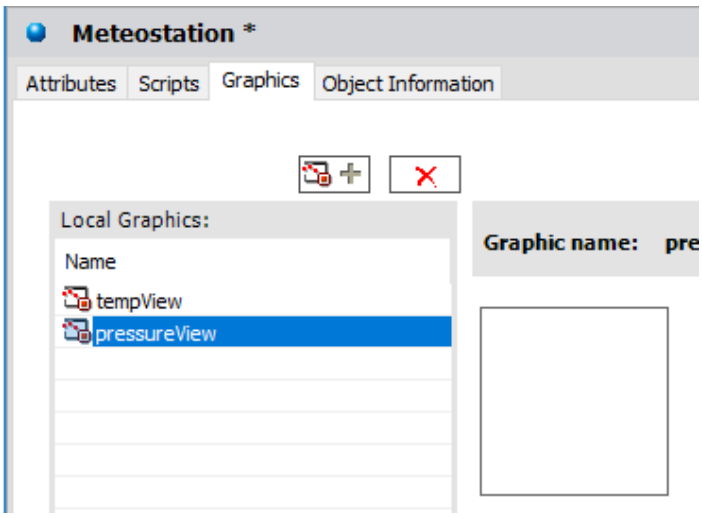

Рисунок 4.64 — pressureView

Із бібліотеки готових графічних об’єктів обрано об’єкт, який буде в подальшому відображати значення тиску (рисунок 4.65).

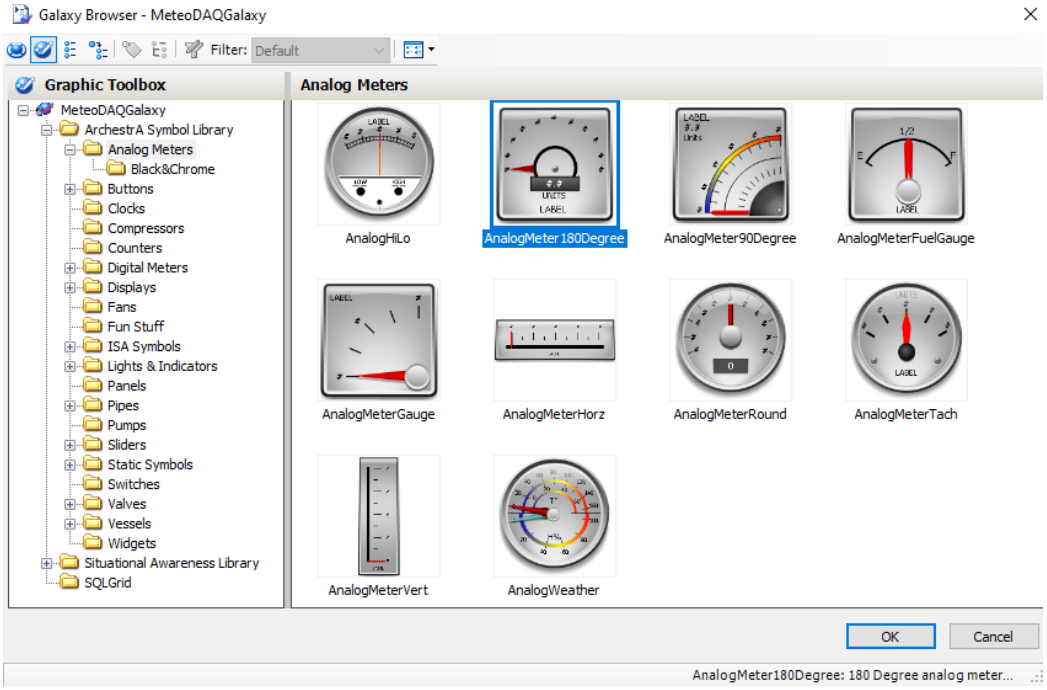

Рисунок 4.65 — Вибір графічного об’єкта для відображення тиску

Наразі, обраний об’єкт містить значення за умовчанням (рисунок 4.66), які необхідно замінити.

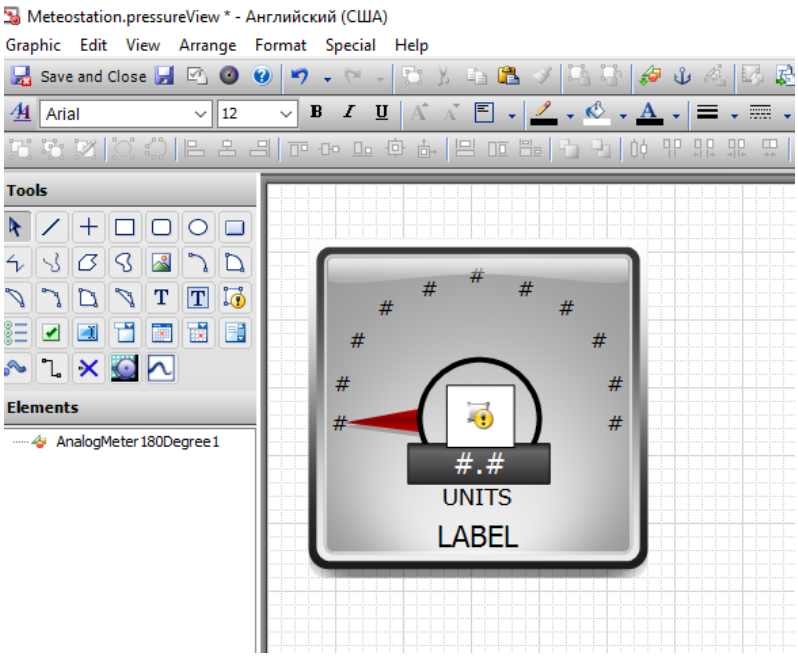

Рисунок 4.66 — Значення за умовчанням

Одиниці вимірювання було вказано як міліметри ртутного стовпчика і задано найменування відображуваної фізичної величини (рисунок 4.67).

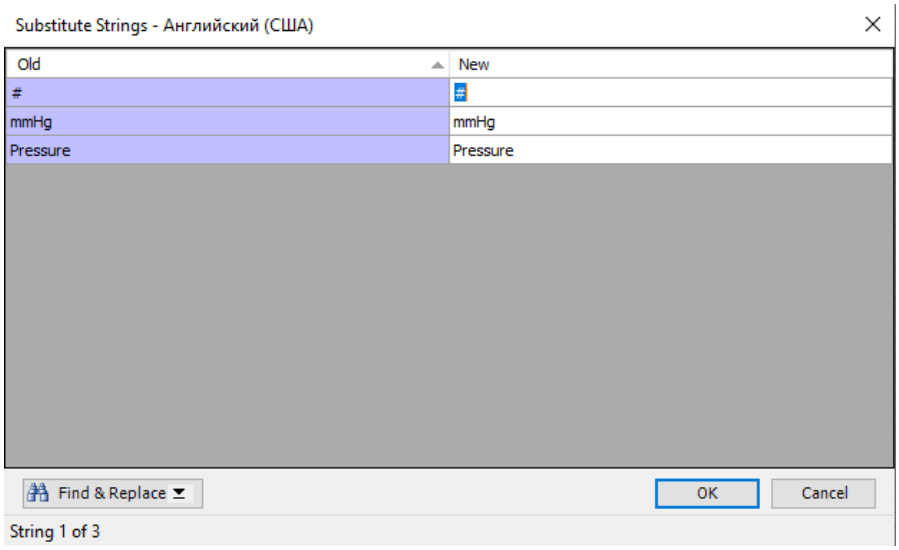

Рисунок 4.67 — Конфігурація статичних текстових параметрів

Після вказання текстових параметрів графічний елемент виглядає так, як показано на рисунку 4.68.

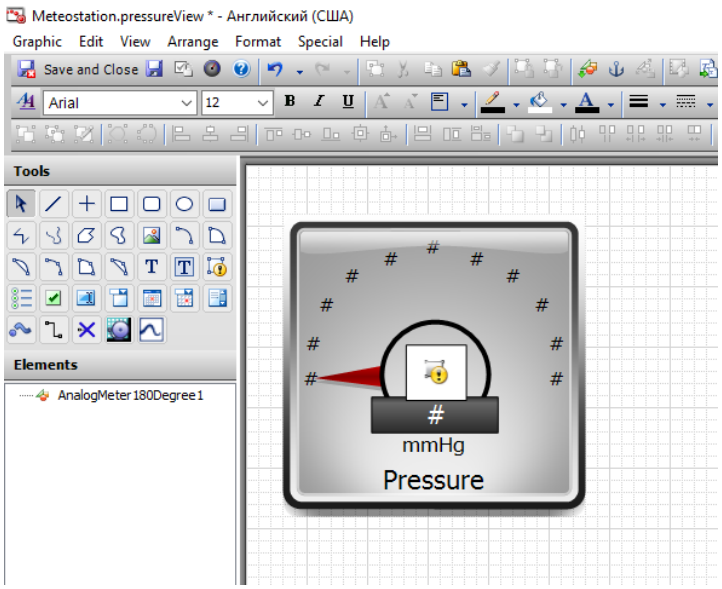

Рисунок 4.68— Графічний елемент pressureView

Наступним кроком є вказання граничних значень та значення для відображення. На рисунку 4.69 показано вибір атрибуту для відображення графічним об’єктом.

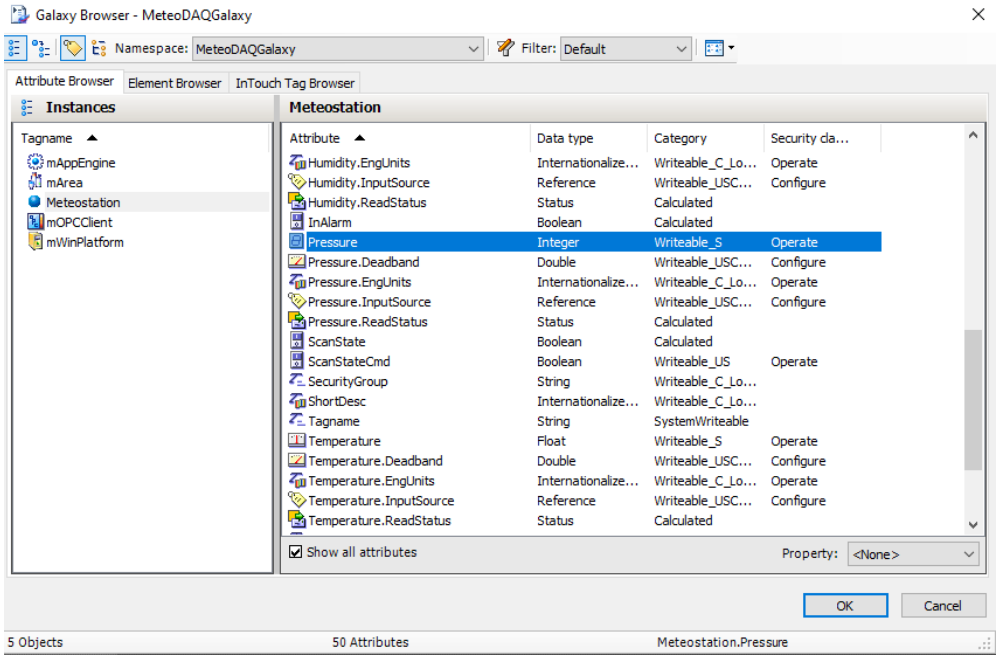

Рисунок 4.69 — Вибір атрибуту для відображення

Параметри конфігурації графічного символу pressureView наведено на рисунку 4.70.

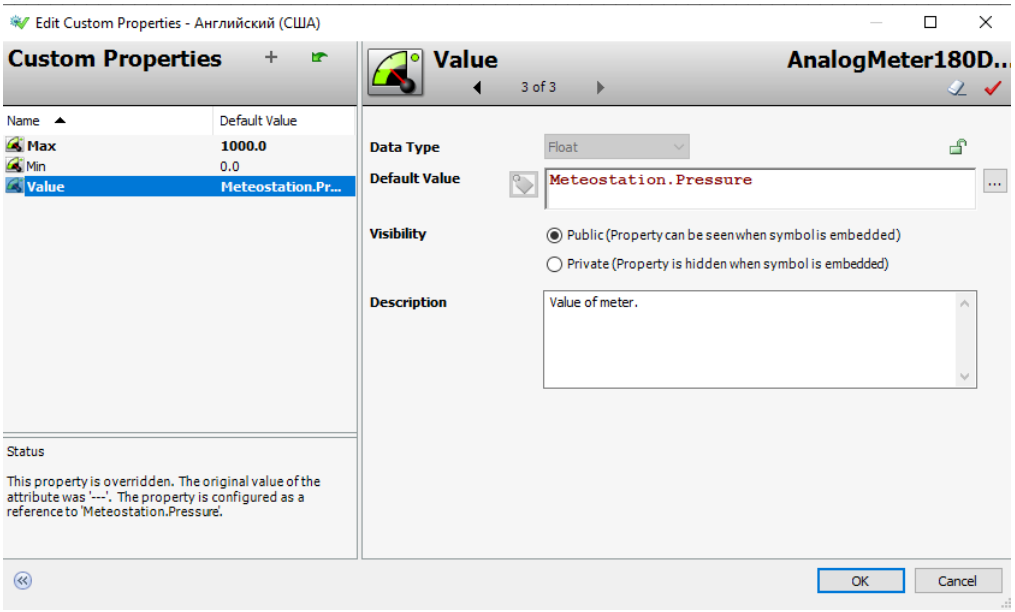

Рисунок 4.70 — Параметри конфігурації графічного символу pressureView

Після того, як робота із символом для відображення значення тиску у графічному редакторі завершена його можна розмістити у вікні візуального інтерфейсу. На рисунку 4.71 показано, що тепер для об’єкта Meteostation доступно два символи.

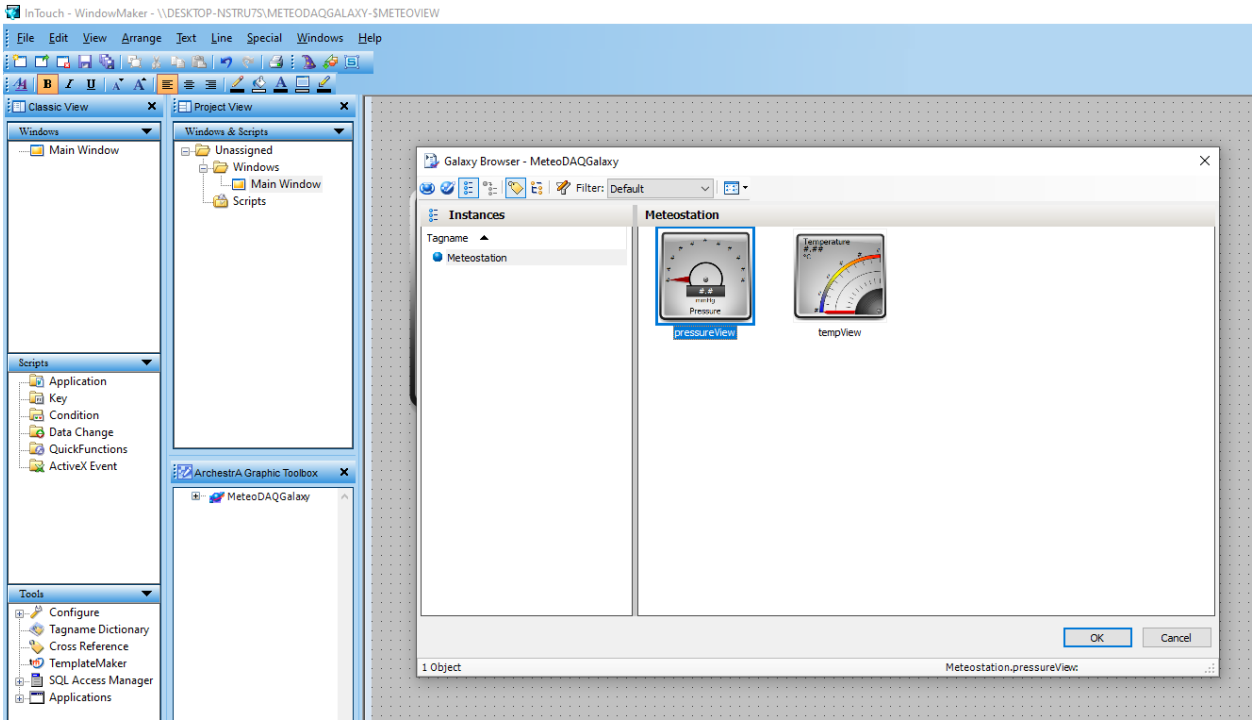

Рисунок 4.71 — Символи, доступні для об’єкта Meteostation

Щойно створений графічний об’єкт pressureView було розміщено у вікні візуалізації. В режимі попереднього перегляду тепер можна спостерігати за значенням тиску (рисунок 4.72).

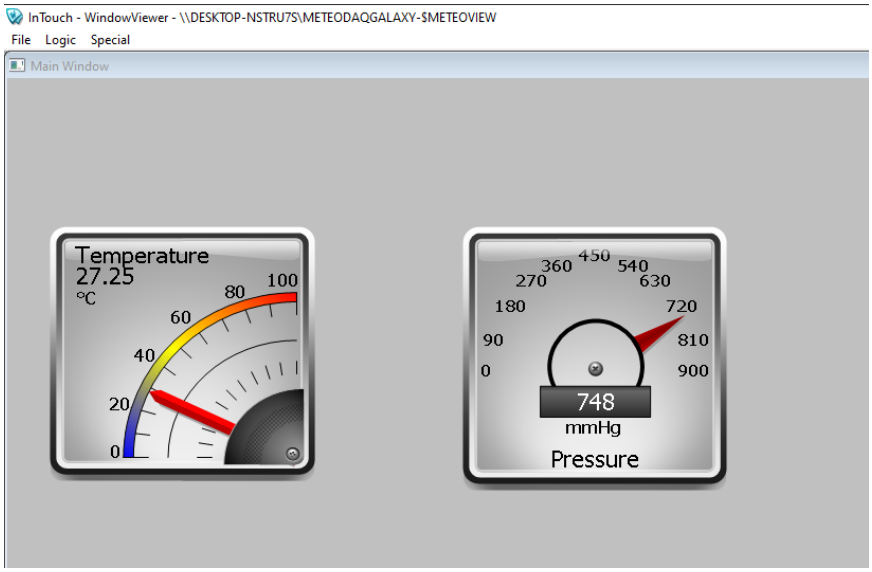

Рисунок 4.72 — Відображення значення тиску

Наступним параметром для відображення буде значення вологості повітря (рисунок 4.73).

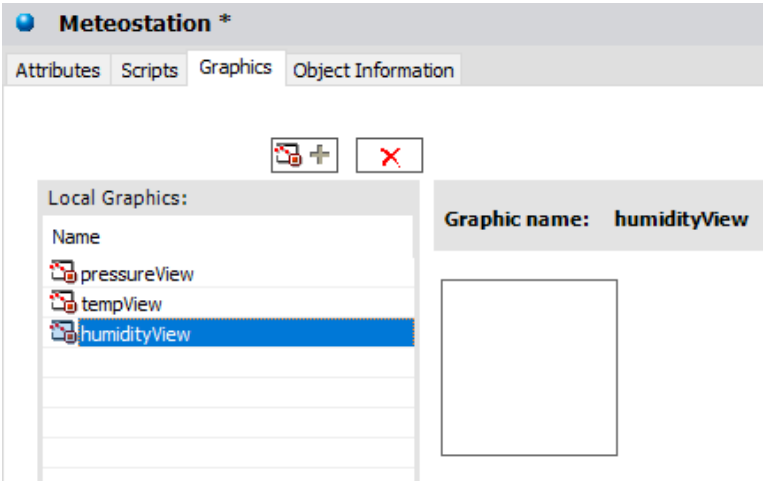

Рисунок 4.73 — humidityView

Для графічного об’єкта задано інженерні одиниці та найменування відображуваної величини (рисунок 4.74).

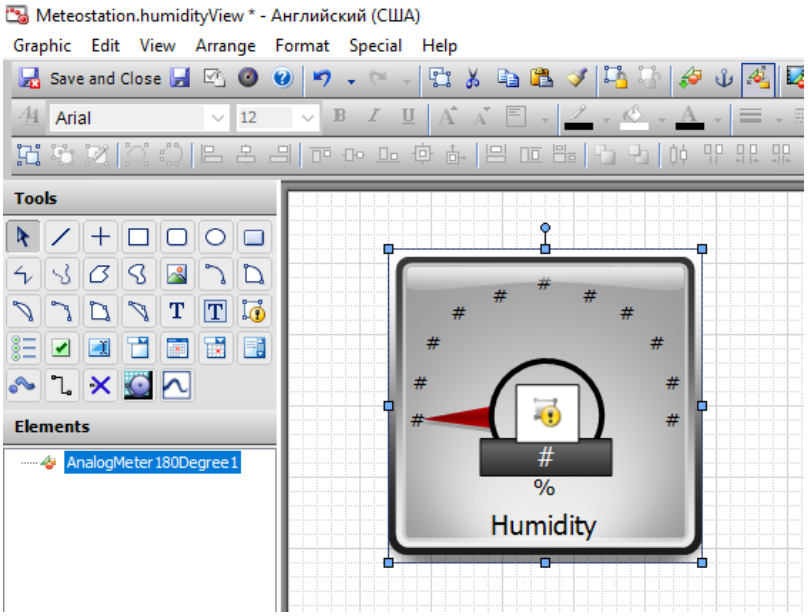

Рисунок 4.74 — Конфігурація текстових параметрів

Задано атрибут, значення якого буде відображатись даним графічним елементом (рисунок 4.75).

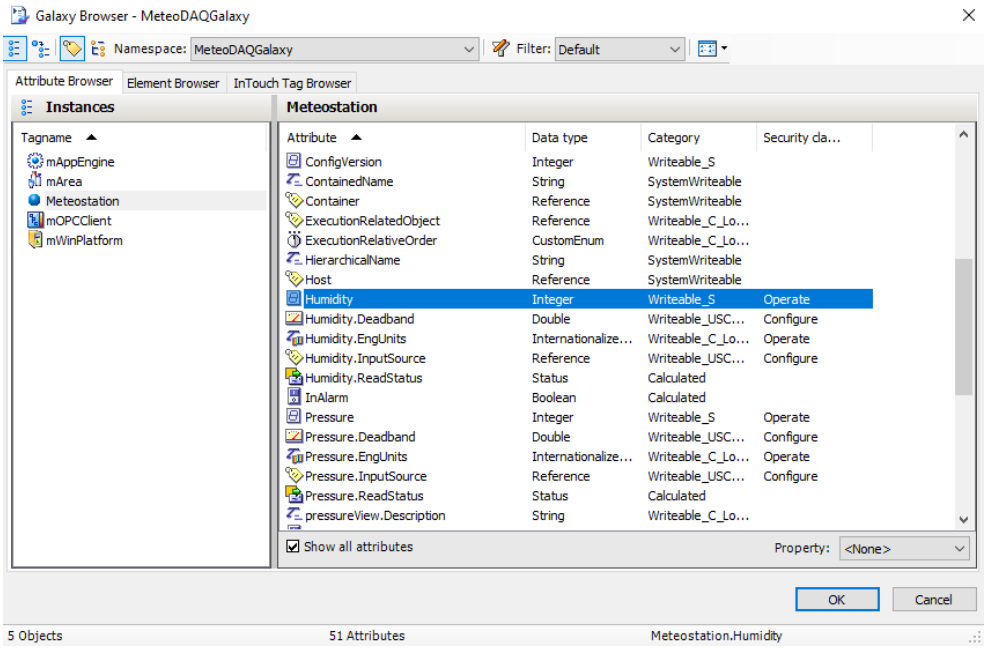

Рисунок 4.75 — Задання атрибуту для відображення

На рисунку 4.76 наведено параметри графічного об’єкта humidityView.

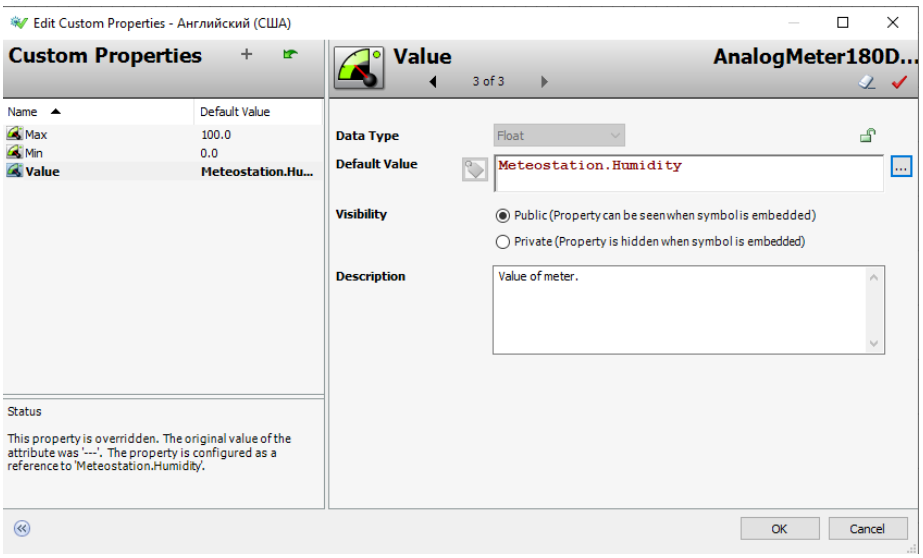

Рисунок 4.76 — Параметри графічного об’єкта humidityView

Тепер в Galaxy браузері є доступним елемент humidityView, який можна розмістити у вікні візуалізації (рисунок 4.77).

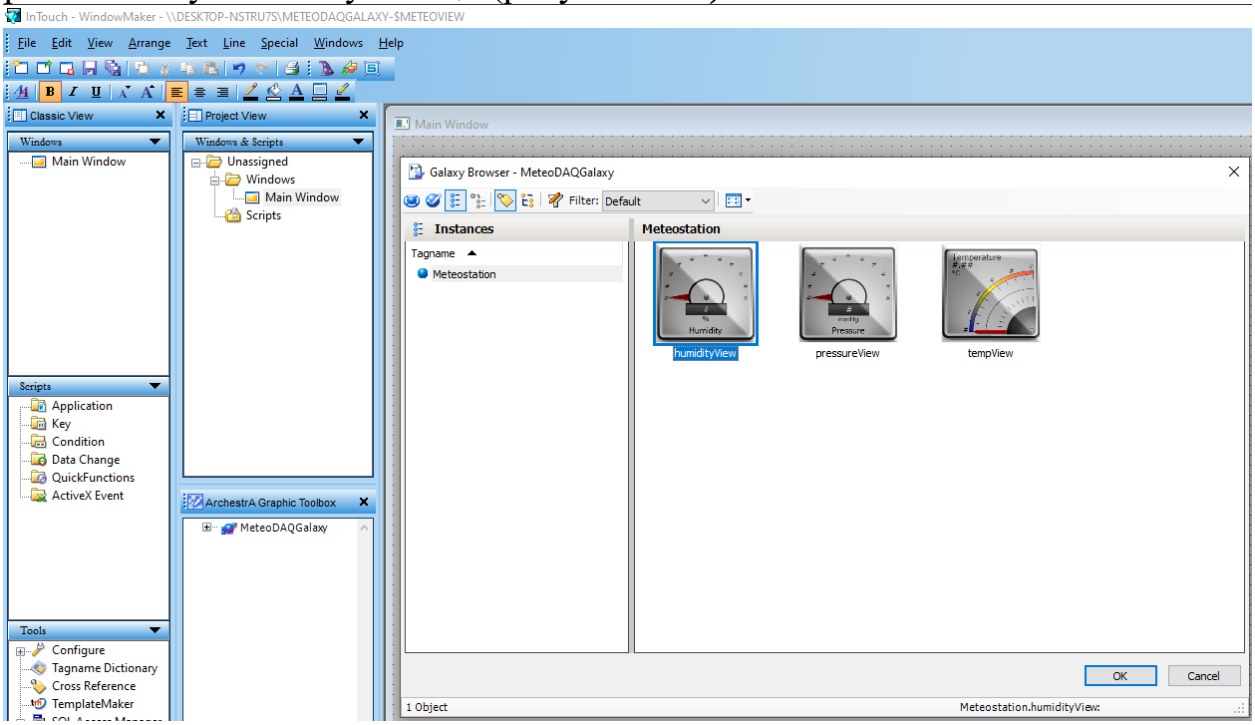

Рисунок 4.77 — Galaxy браузер

Елемент humidityView було розміщено на вікні людино-машинного інтерфейсу (рисунок 4.78).

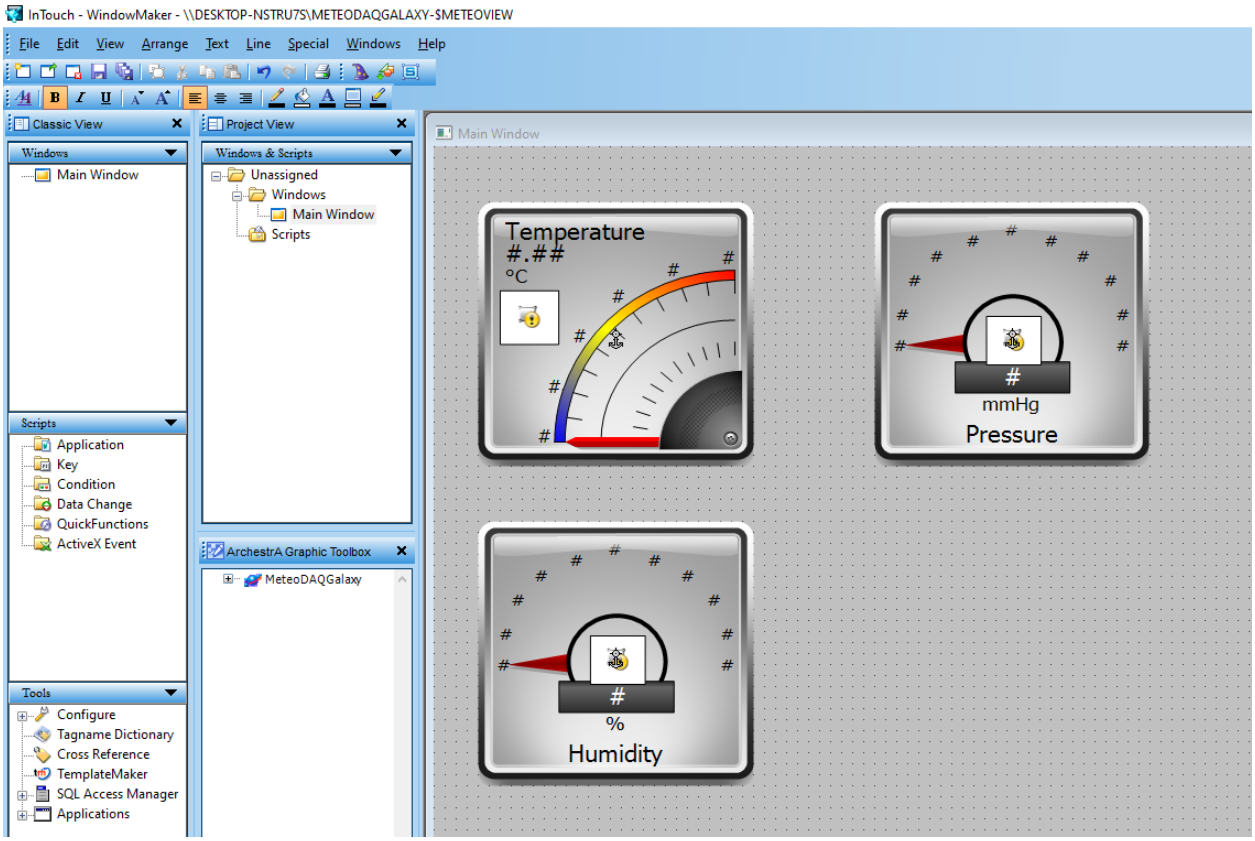

Рисунок 4.78 — Розміщення елемента humidityView

Після того як humidityView було розміщено, можна переглянути значення, які відображає даний елемент (рисунок 4.79).

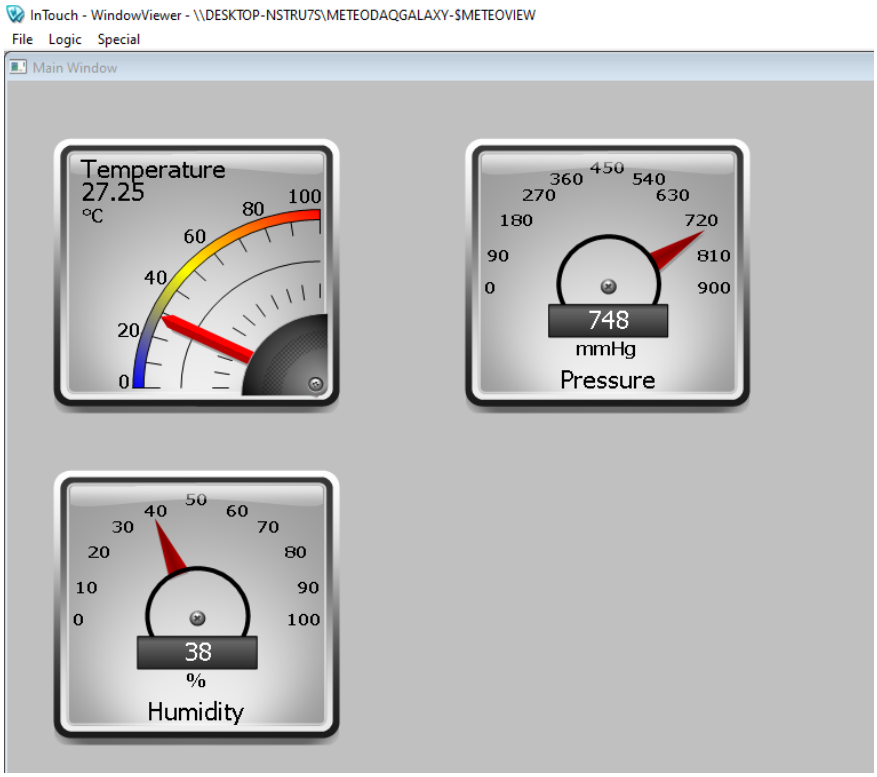

Рисунок 4.79 — Відображення значення вологості

Наступним атрибутом для якого буде створюватись графічне відображення, буде значення вмісту вуглекислого газу у повітрі (рисунок 4.80).

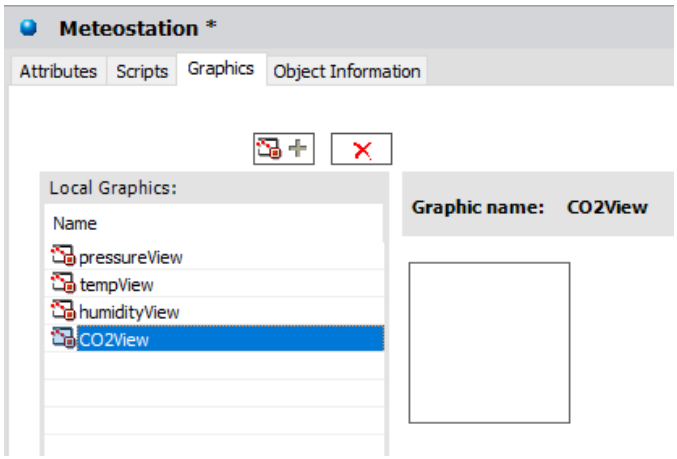

Рисунок 4.80 — CO2View

Було задано текстові параметри графічного об’єкта: одиниці виміру та найменування параметра (рисунок 4.81).

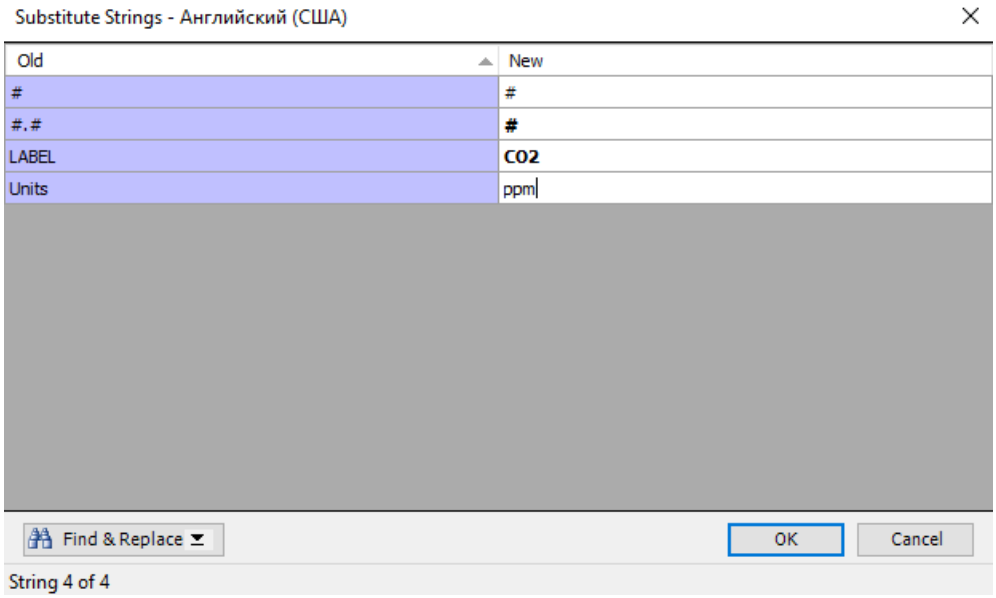

Рисунок 4.81 — Параметри символа

Задано граничні значення та вказано атрибут для візуалізації (рисунок 4.82).

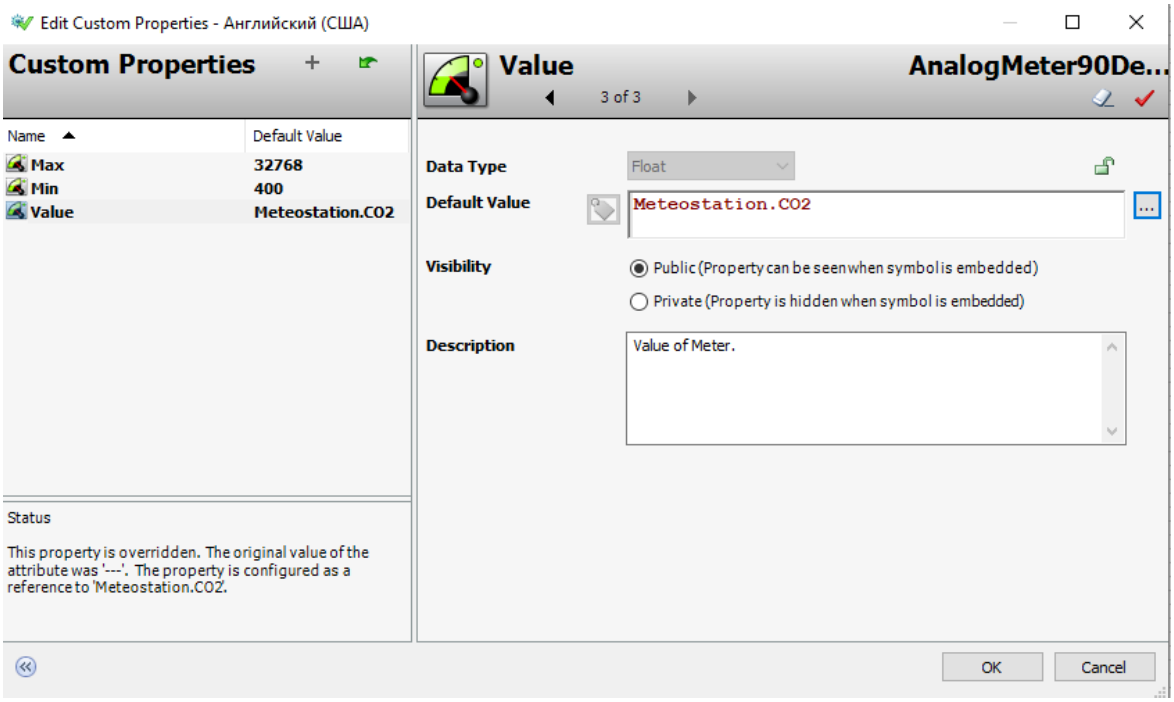

Рисунок 4.82 — Користувацькі налаштування

Для організації кольорової індикації стану повітря у налаштуваннях об’єкта Meteostation було створено три нові атрибути: air_green, air_yellow, air_red (рисунок 4.83). Тип даних створених атрибутів – логічний.

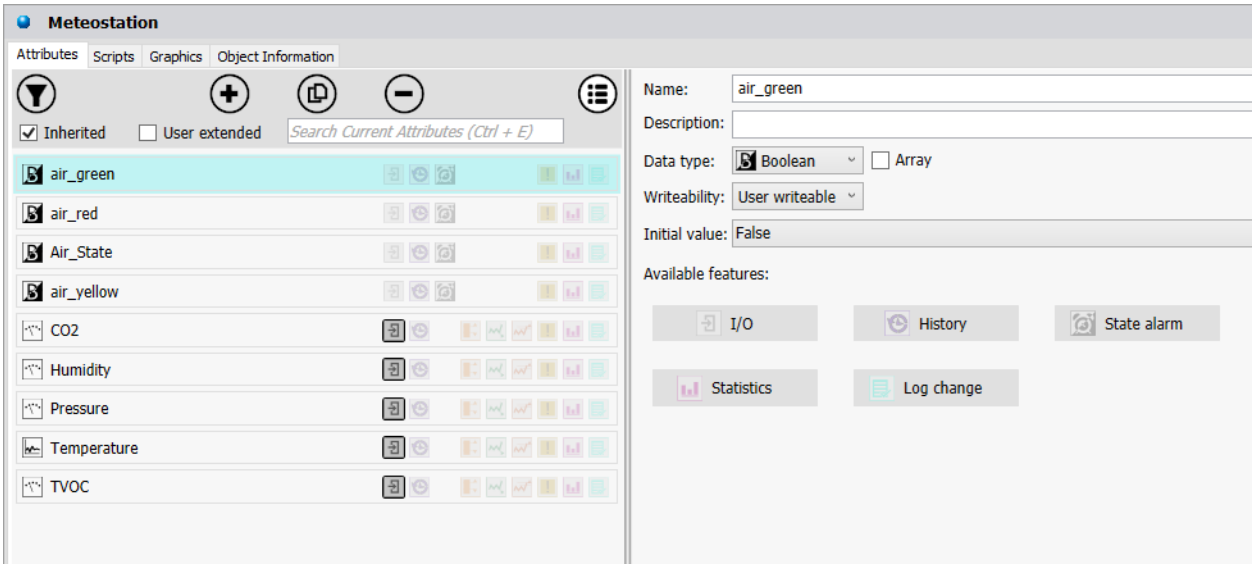

Рисунок 4.83 — Логічні атрибути

Дані атрибути приймають значення true у разі перевищення значення вмісту вуглекислого газу певного порогового значення. Істинним може бути лише одне із значень в один момент часу. Нижче (рисунок 4.84) наведено скрипт, задачею якого є встановлення значень логічних атрибутів у відповідності до рівня концентрації CO2.

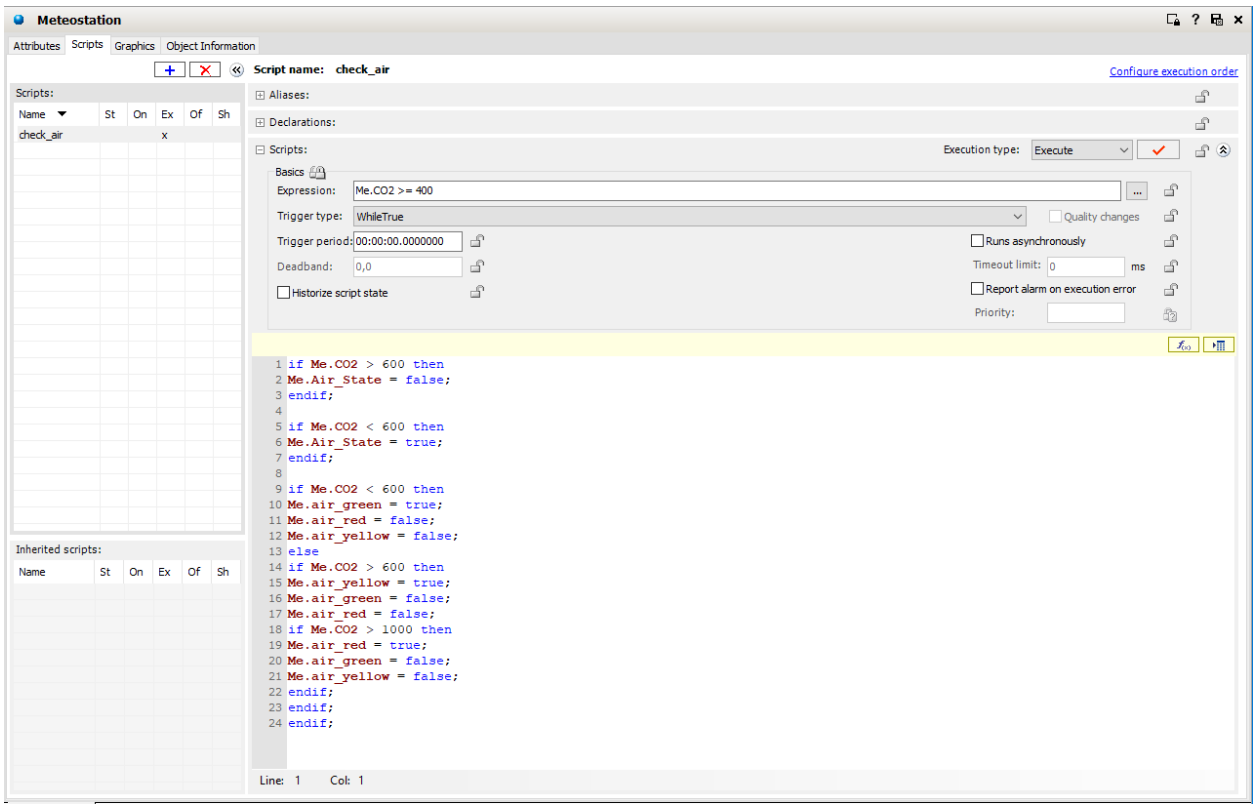

Рисунок 4.84 — Скрипт встановлення значень логічних атрибутів

Індикація здійснюється на основі створених логічних атрибутів. Для кожного кольору індикатора (рисунок 4.85) як значення за умовчанням задається відповідний логічний атрибут: для зеленого кольору – air_green; для жовтого — air_yellow; для червоного — air_red.

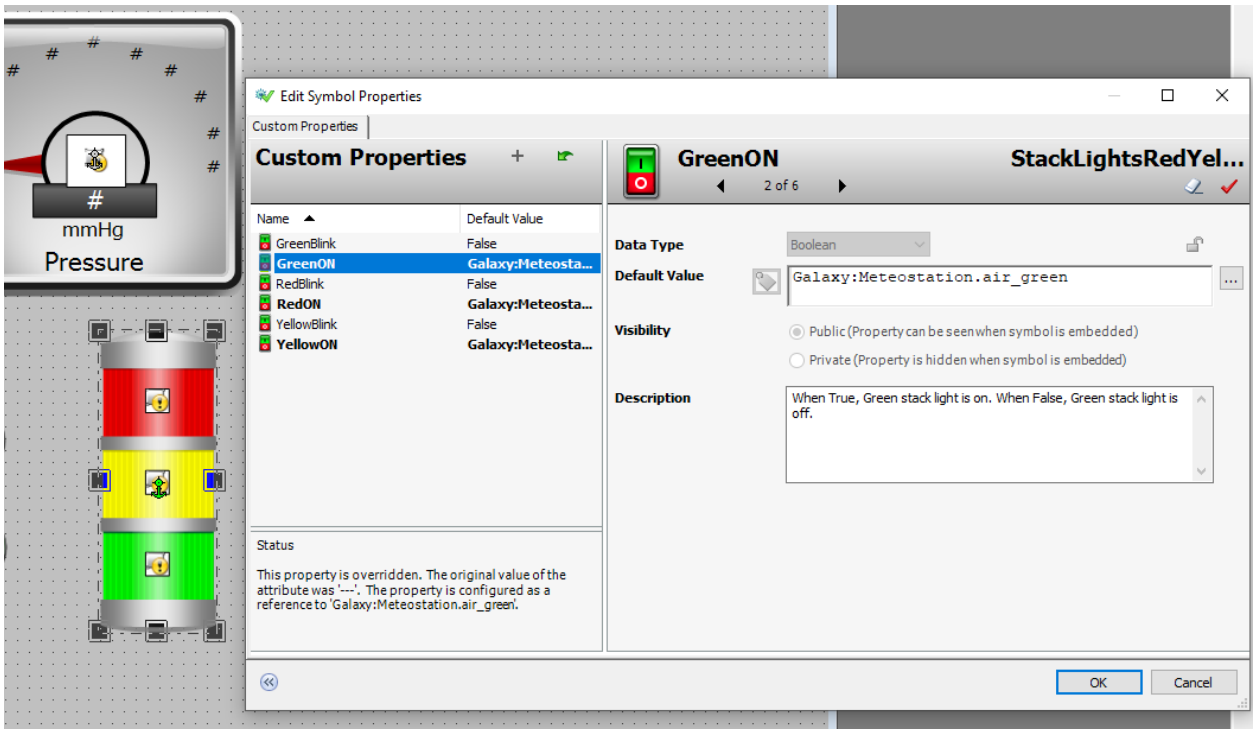

Рисунок 4.85 — Присвоєння атрибуту кольорам індикатора

Після конфігурації індикатора та розміщення його у вікні людино-машинного інтерфейсу, було виконано перевірку правильності роботи кольорової індикації. На рисунках 4.86-4.88 наведено зображення стану індикатора при різних значеннях CO2.

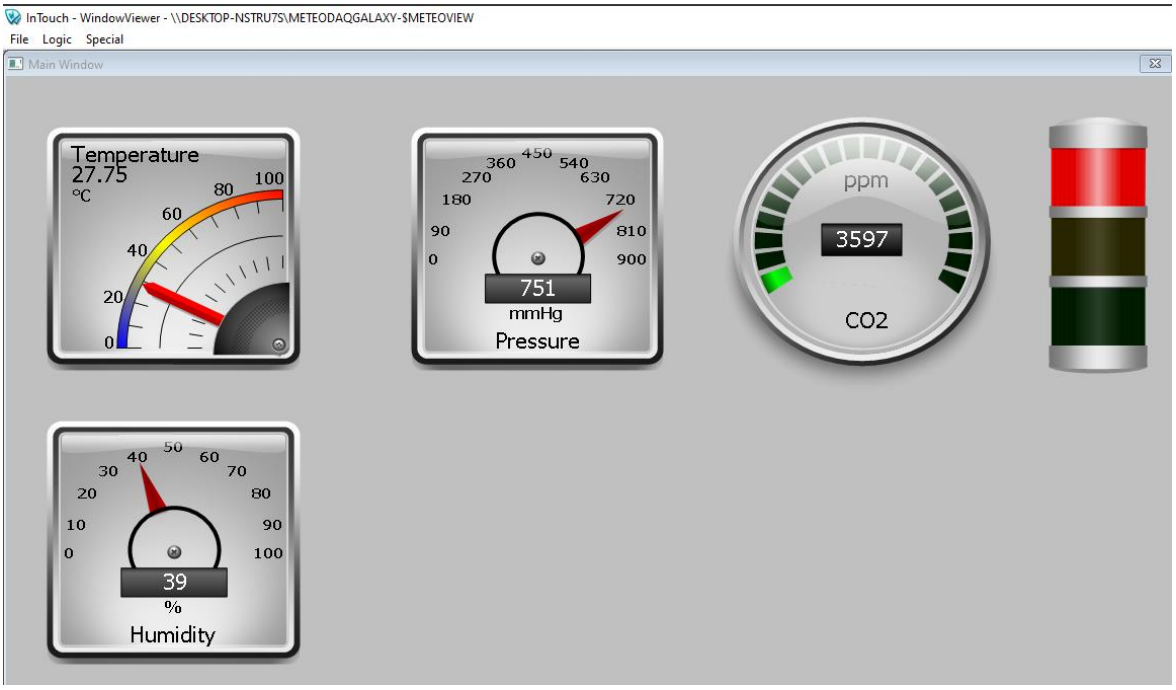

Рисунок 4.86 — Тестування кольорової сигналізації (критичний рівень)

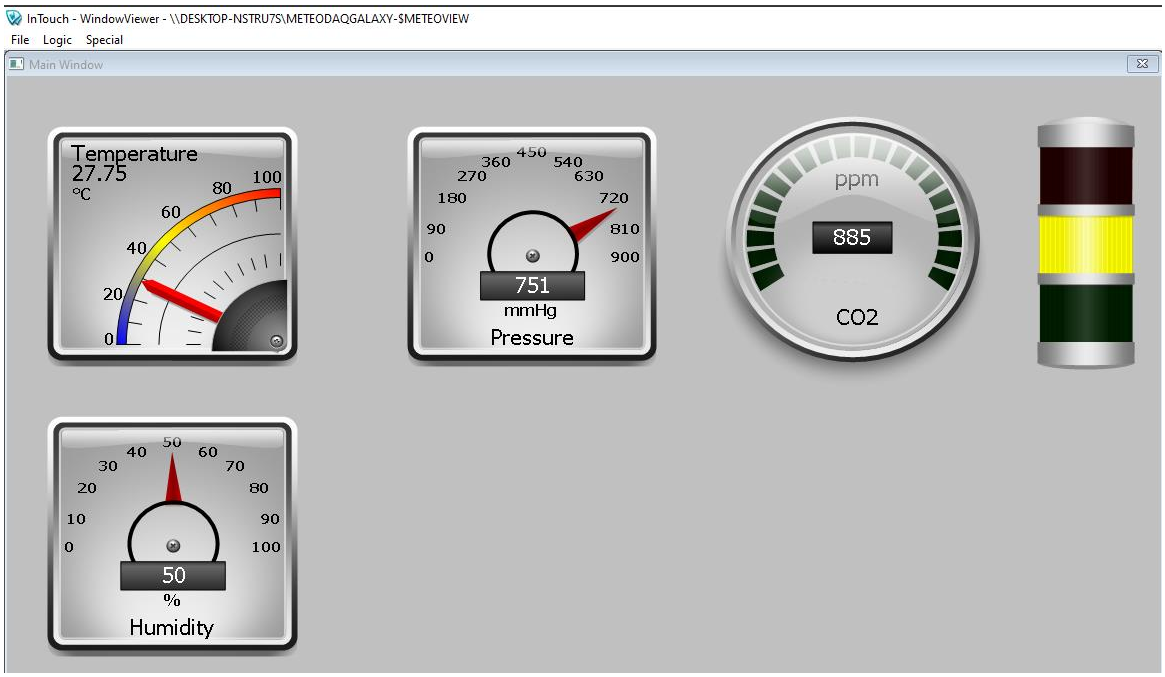

Рисунок 4.87 — Тестування кольорової сигналізації (підвищений рівень)

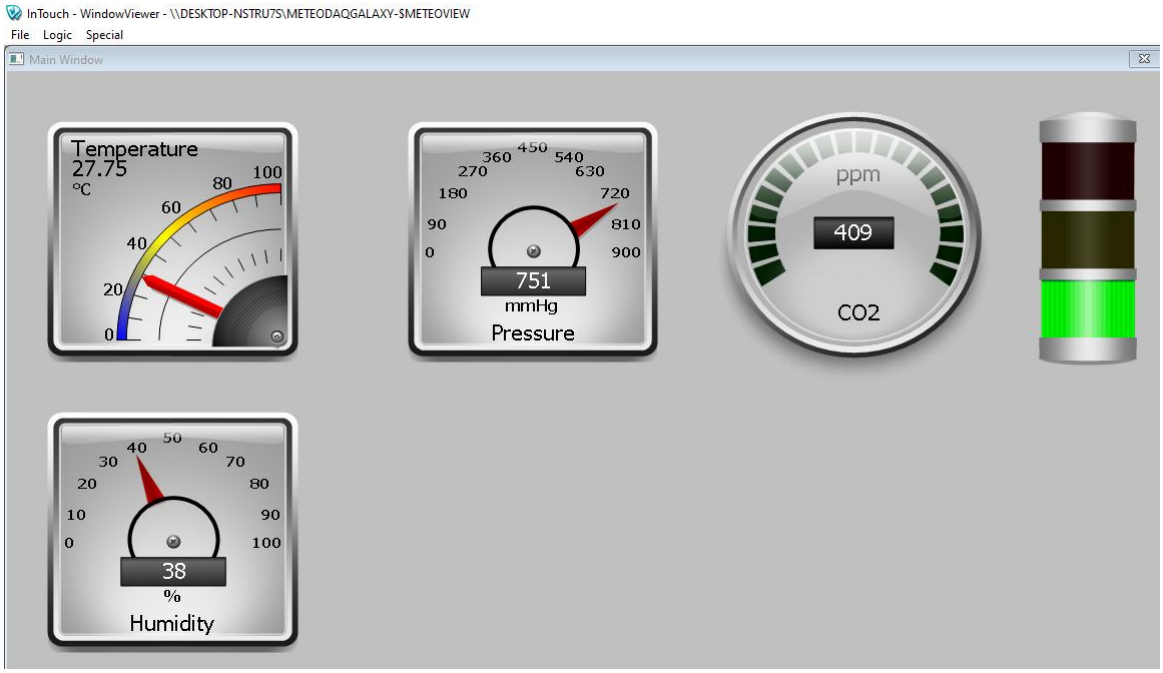

Рисунок 4.88 — Тестування кольорової сигналізації (нормальний рівень)

Останнім параметром, значення якого необхідно візуалізувати, є рівень вмісту летких органічних речовин (TVOC). Для об'єкта Meteostation створено новий графічний об'єкт (рисунок 4.89).

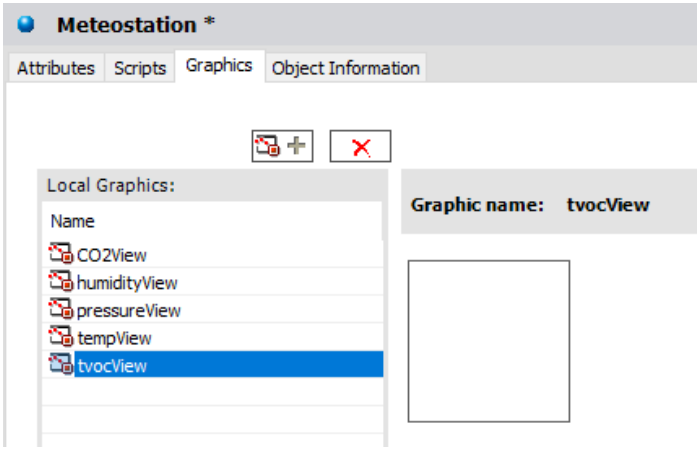

Рисунок 4.89 — Створення графічного об'єкта tvocView

Для даного об'єкта було обрано графічний символ із бібліотеки символів, виконано заміну текстових параметрів на такі, що відповідають атрибуту TVOC (рисунок 4.90).

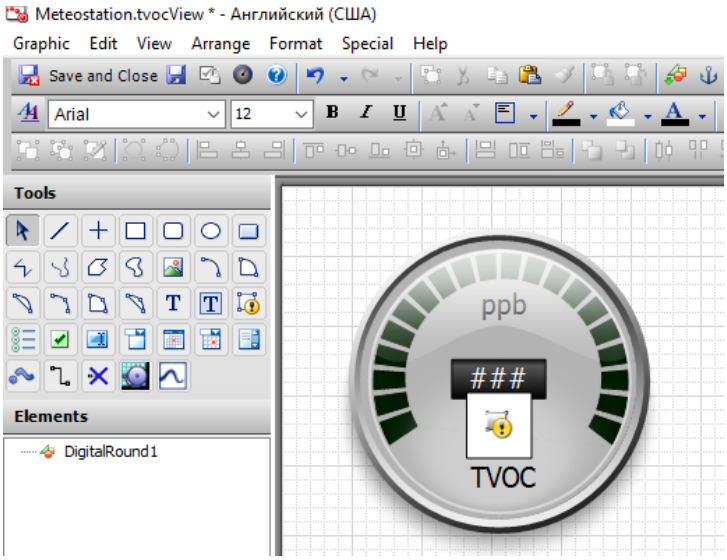

Рисунок 4.90 — Графічний символ TVOC

У користувацьких параметрах було задано граничні значення для відображуваного даних символом атрибута об’єкта Meteostation, задано атрибут, значення якого буде відображатись (рисунок 4.91).

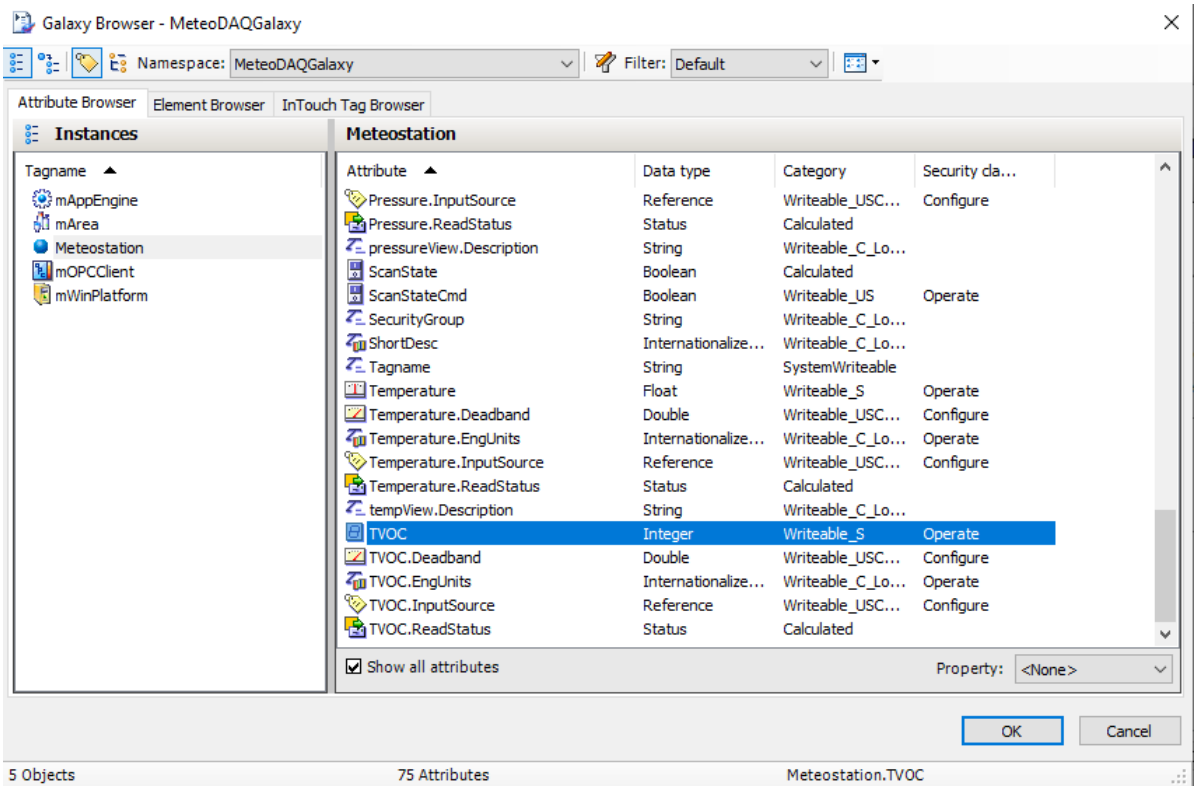

Рисунок 4.91— Атрибут TVOC

Символ, створений для відображення атрибуту TVOC, розміщено на полі вікна розроблюваного візуального інтерфейсу (рисунок 4.92).

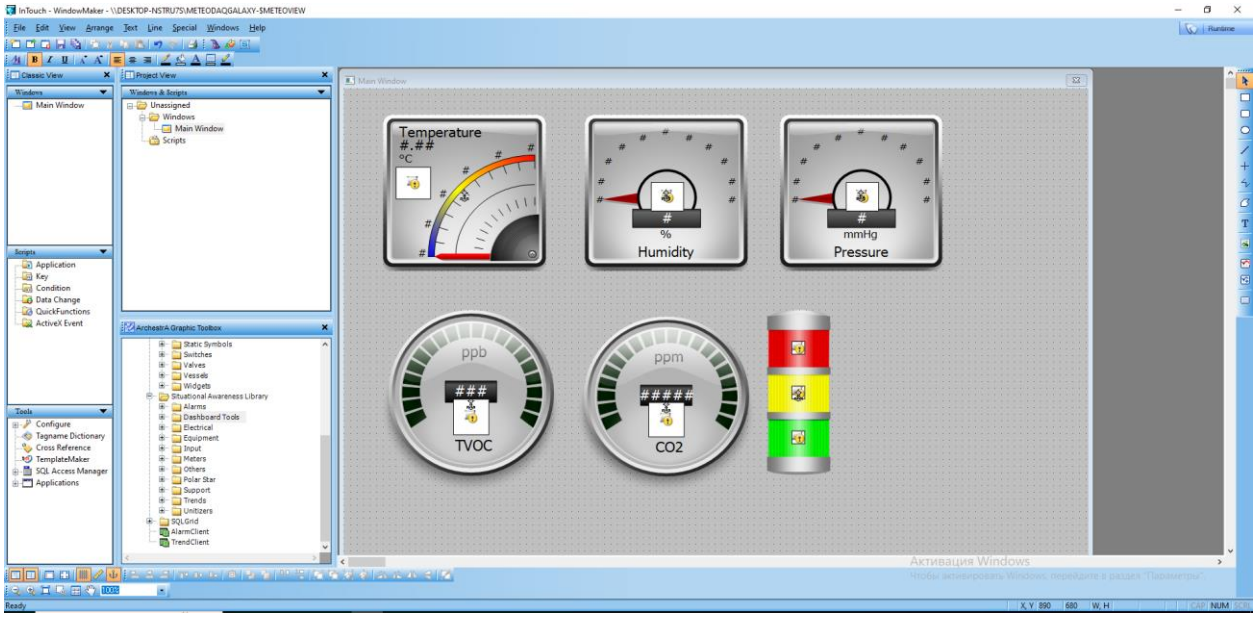

Рисунок 4.92 — Положення символів відображення

Після додавання символу відображення рівня вмісту TVOC людино-машинний інтерфейс завершено. На рисунку 4.92 наведено остаточний вигляд HMI.

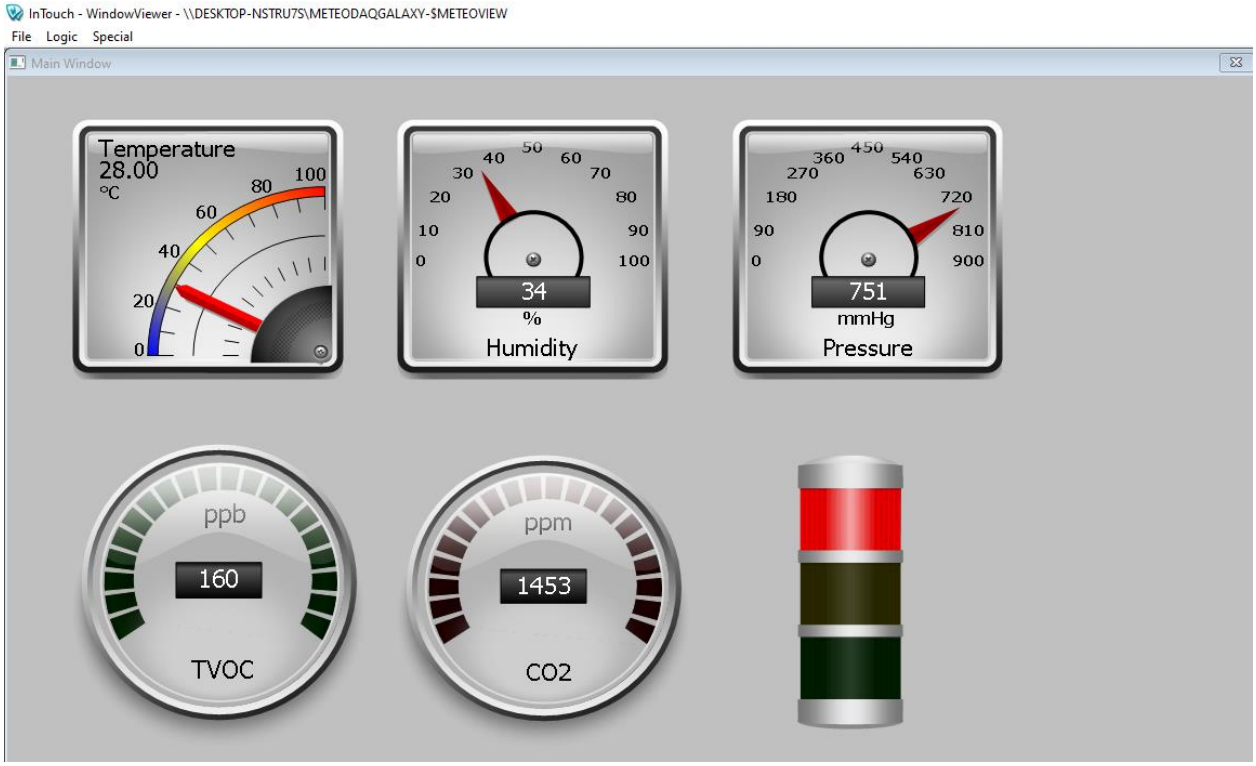

Рисунок 4.92 — Людино-машинний інтерфейс

Висновок за розділом

В даному розділі була виконана практична реалізація системи збору даних засобами Aveva System Platform, використовуючи можливості ArchestraIDE та InTouch. Після практичного ознайомлення та роботи із даним програмним забезпеченням можна відмітити інтуїтивно зрозумілий інтерфейс,

зручність користування, наявність довідкової інформації для кожного програмного компоненту, попереднього перегляду без необхідності деплою. Також, варто відзначити високий рівень інтеграції програмних компонентів Системної Платформи, що дозволяє створювати гнучкі додатки автматизації з можливістю колективної розробки.

Було виконано створення нового додатку MeteoDAQGalaxy. Створено та виконано конфігурацію необхідних об’єктів. Для представлення ардуіно-метеостанції у Системній Платформі на основі системного шаблону «UserDefined» було створено новий шаблон «Sensor», який представляє деякий пристрій вимірювання. На основі шаблону Sensor створено екземпляр Meteostation. Для екземпляру Meteostation задано атрибути, які відповідають вимірюваним метеостанцією параметрам. Було виконано конфігурацію OPC сервера та OPC клієнта для передачі даних від ардуіно до Galaxy. Розроблено людино-машинний інтерфейс.

ВИСНОВОК

Розвиток науки, удосконалення виробничих процесів, розвиток майже всіх без виключення галузей народного господарства, оборонної промисловості, в значній мірі залежить від своєчасного та якісного збору вимірювальної інформації. Встановлено, що системи збору даних зазвичай не виокремлюються як окрема структурна одиниця, а входять до складу інших систем, таких як системи управління виробництвом, системи життєзабезпечення, системи контролю тощо. В даній роботу було здійснено спробу виокремлення систем збору даних як окремого класу інформаційних систем з метою детального аналізу та узагальнення існуючих теоретичних відомостей про такі системи. В результаті проведеного дослідження було створено узагальнену функціональну модель для системи збору даних.

Визначено, що даний клас систем представляє собою програмно-апаратні системи, які застосовуються для вимірювання, обробки иа накопичення даних. Першочерговою задачею таких систем є отримання даних від досліджуваної системи. З цією метою застосовуються вимірювальні перетворювачі – датчики. Датчики оперують на нижньому рівні систем збору даних. Вони виконують функцію вимірювання кількісних показників, якими характеризується досліджувана система. Після одержання виміряного сигналу він проходить процес кондиціонування, тобто сигнал, отриманий від вимірювального пристрою, приводиться у вигляд придатний для подальшої обробки іншими пристроями. Кондиціонування може включати підсилення, фільтрацію, лінеаризацію тощо. Так як система обробки даних є цифровою, а виміряний сигнал, зазвичай, є аналоговим, то в процесі збору даних має місце етап аналого-цифрового перетворення, яке здійснюється за допомогою АЦП. Для передачі виміряних даних застосовуються системи зв’язку.

Для практичної реалізації системи збору даних було розроблено електронний пристрій, який виконує функції моніторингу за станом навколишнього середовища із можливістю відображення вимірюваних даних на символьному LCD дисплеї. Основною структурною та функціональною одиницею розробленого пристрою є Arduino Nano — плата розробки на базі мікроконтролера ATmega328P. Розроблений пристрій здійснює вимірювання температури, атмосферного тиску та вологості. Вимірювання здійснюють інтелектуальні датчики BME280 та CSS811.

В якості програмного забезпечення збору даних застосовано Aveva System Platform. Засобами Системної Платформи було організовано отримання показань датчиків із застосуванням технології OPC, реалізовано людино-машинний інтерфейс для візуалізації виміряних даних.

Після практичного ознайомлення та роботи із даним програмним забезпеченням можна відмітити інтуїтивно зрозумілий інтерфейс, зручність користування, наявність довідкової інформації для кожного програмного компоненту, попереднього перегляду без необхідності деплою. Також, варто відзначити високий рівень інтеграції програмних компонентів Системної Платформи, що дозволяє створювати гнучкі додатки автоматизації з можливістю колективної розробки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Паламар М. І. Контрольно-вимірювальні комплекси: Конспект лекцій. - Тернопіль: ТНТУ, 2014. – 157
2. Парахуда Р., Шевцов В. Автоматизация измерений и контроля. СПб. 75с.
3. Сусліков Л.М., Студеняк І.П. Первинні вимірювальні перетворювачі фізичних величин: Навчальний посібник. – Ужгород: Видавництво УжНУ, 2018. - 311 с.
4. Крюков В.В. Информационно-измерительные системы. Учебное пособие. – Владивосток: ВГУЭС, 2000. - 102 с.
5. Цапенко М. Измерительные информационные системы: Структуры и алгоритмы, системотехническое проектирование : учебное пособие. 2-ге вид. Энергоатомиздат.
6. Data Collection | Definition, Methods & Examples. Scribbr. URL: https://www.scribbr.com/methodology/data-collection/.
7. Austerlitz H. Data Acquisition Techniques Using PCs. Academic Press, 2003.
8. Data Acquisition Handbook. 3rd ed. Measurement Computing Corporation.
9. Park J., Mackay S. Practical Data Acquisition for Instrumentation and Control Systems. 2003.
10. Bolton W. Instrumentation and Control Systems. 3rd ed. 2021.
11. Коваль А. Вимірювальні перетворювачі : конспект лекцій. Харків, 2018.
12. Fraden J. Handbook of Modern Sensors Physics, Designs, and Applications. 5th ed.
13. The Importance of Signal Conditioning. URL: https://www.omega.com/en-us/resources/signal-conditioners.
14. James K. PC Interfacing and Data Acquisition. NEWNES, 2000. 448 p.
15. Physical Sensors: Thermal Sensors / T. Dinh et al. Encyclopedia of Sensors and Biosensors. 2023. Vol. 1. P. 20–33.
16. What is galvanically isolated?. Greentech Renewables. URL: https://www.greentechrenewables.com/question/what-galvanically-isolated.
17. Signal filtering, Signal suppression, Signal processing | Dewesoft. Dewesoft Training Portal |. URL: https://training.dewesoft.com/online/course/filters.
18. Bishop R. R. The Mechatronics Handbook. 2nd ed. 2008.
19. Панфілов І., Дирда В., Капацін А. Теорія електричного зв'язку. Київ, 1998. 328 с.
20. Журавський Ю., Полторак В. Теорія інформації та кодування : підручник. Київ : Вища шк., 2001. 255 с.
21. Клюев Л. Теория электрической связи. Минск : Дизайн ПРО, 1998. 336 с.
22. Биккенин Р., Чесноков М. Теория электрической связи. 336 с.

23. Кон Е. Л. Теория электрической связи. Помехоустойчивая передача данных в информационно-управляющих и телекоммуникационных системах: модели, алгоритмы, структуры / Е. Л. Кон, В. И. Фрейман., 2007. – 312 с.
24. Васюков В. Н. Теория электрической связи / В. Н. Васюков., 2005. – 392 с.
25. Теория электрической связи / А. Зюко та ін. 1999. 432 с.
26. Теорія електричного зв'язку: навч.посібник / О. Ю. Гусєв та ін.
27. Батаєв О.П., Ковтун І.В., Корольова Н.А. Теорія електричного зв'язку: Навч. посібник. – Харків: УкрДАЗТ, 2010. - 630 с.
28. Шварцман В., Емельянов Г. Теория передачи дискретной информации
29. Андреев Р., Краснов Р., Чепелев М. Теория электрической связи. 2014. 230 с.
30. Бурштинський М., Хай М., Харчишин Б. Давачі : навч. посіб. 2-ге вид. Львів : Простір М, 2014. 202 с.
31. E.T. Powner, F. Yalcinkaya, (1995),"Intelligent sensors: structure and system", Sensor Review, Vol. 15 Iss 3 pp. 31 – 35
32. What makes sensor devices and microsystems 'intelligent' or 'smart'?. Smart sensors and MEMS Intelligent devices and microsystems for industrial applications. 2014. No. 51. P. 1–21.
33. Huijsing, J. H. (2008), 'Smart sensor systems: Why? Where? How?', in Smart Sensor Systems, G. C. M. Meijer (ed.), Chichester, UK, John Wiley & Sons
34. Raspberry Pi | Data Acquisition (DAQ) Products |Data Logger | Measurement Computing. Data Acquisition (DAQ) - Measurement Computing. URL: https://www.mccdaq.com/DAQ-HAT.aspx.
35. Data Acquisition and Monitoring devices - SAAB RDS. SAAB RDS. URL: https://saabrds.com/solutions/data-acquisition-and-monitoring/.
36. Di Paolo Emilio M. Data Acquisition Systems From Fundamentals to Applied Design. 2013.
37. Young S. S. 5 and Analysis for the Life Sciences: A Hands-on Guide. Cambridge University Press, 2001.
38. ArchestrA Technology. URL: https://docplayer.net/23254088-Archestra-technology-understanding-microsoft-s-net-technology-its-impact-on-automation-application-development-deployment.html
39. Wonderware Application Server User's Guide. 2015. 468 с.
40. AVEVA System Platform. URL: https://www.aveva.com/en/products/system-platform
41. HMI и Диспетчеризация. URL: https://docplayer.com/46795314-Hmi-i-dispetcherizaciya.html
42. Application Server. URL: http://archestra.info/index.php/Application_Server
43. Общее описание архиватора Wonderware Historian. URL: https://docplayer.com/72850293-Obshchee-opisanie-arhivatora-wonderware-historian.html
44. Wonderware Historian. URL: https://www.servo.com.sg/sites/default/files/2018-01/Datasheet_SE-LIO-

Wonderware_Historian_05-17_0.pdf
45. Wonderware Information Server. URL: https://cdn.logic-control.com/media/Datasheet_WW_InformationServer2014R2.pdf
46. Продукты для организации ЧМИ и систем диспетчерского управления. WONDERWARE INTOUCH HMI. URL: https://klinkmann.ru/products/wonderware/produkty-dlya-organizatsii-chmi-i-sistem-dispetcherskogo-upravleniya/wonderware-intouch-hmi/
47. SCADA-система InTouch. URL: http://www.promsat.com/page/9/
48. InTouch HMI Concepts and Capabilities Guide. 2007. 114 с.
49. Руководство пользователя ИСР Archestra. 2006. 240 с.
50. Сервер приложений Wonderware Руководство пользователя. 2008.266 с.
51. System Platform 2017 Update 3 Getting Started Guide. 2018. 58 с.
52. Bootloader. URL: https://docs.arduino.cc/hacking/software/Bootloader
53. ATMEGA328P Datasheet (PDF) - ATMEL Corporation. URL: https://pdf1.alldatasheet.com/datasheet-pdf/view/241077/ATMEL/ATMEGA328P.html
54. Kaul L. Practical Arduino Robotics. Packt Publishing, 2023.
55. Getting Started with Arduino IDE 2. URL:https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2.
56. BME280. Combined humidity and pressure sensor. URL: https://pdf1.alldatasheet.com/datasheet-pdf/view/1132060/BOSCH/BME280.html
57. CCS811. Ultra-Low Power Digital Gas Sensor for Monitoring Indoor Air Quality. URL: https://pdf1.alldatasheet.com/datasheet-pdf/view/1047395/AMSCO/CCS811.html
58. DS3231. Extremely Accurate I2C-Integrated RTC/TCXO/Crystal. URL: https://pdf1.alldatasheet.com/datasheet-pdf/view/112132/DALLAS/DS3231.html
59. What is OPC? URL: https://opcfoundation.org/about/what-is-opc/
60. Мишунин В. Информационно-измерительные и управляющие системы. 2010. 129 с.