Where no SCADA has gone before

Real-time database 4.0

A key differentiation point of the FactoryStudio platform is its ability to be applied in both hard Process Control with real-time, in-memory information and also on Level 2 and Level 3 applications, where the information is stored on SQL databases and exchanged with external applications. The real-time core of FactoryStudio is an in-memory, event-driven database. That database is the result of more than 25 years of continuous learning and it is a complete new design, leveraging the current technologies and the past experiences.

> Real-time Information Management, IT and SCADA combined

The function of real-time database is to allow the modularity of the system, creating an abstraction layer that allows you to isolate tasks to communicate with relational databases, communication with other systems and with the field, the user interfaces and systems of calculation and optimization. Its structure enables synchronization between the various processes independent of values in real-time, event notification and update tables of information.

This manager uses components designed specifically for the .NET platform with the infrastructure for tracking events. Making a comparison with a real-time supervisory, or a level 1, we have the following key differences:

HMI/SCADA databases	FactoryStudio real-time database
Basic Tags Types, such as boolean, int, float (real) and texts	In addition to these, also manages events, dates, tables, queries, and structured data.
Server-centric and centralized module processing	Distributed processing, multi-core and multi-processes, client and advanced control stations.
Proprietary access Interfaces	Access via Interfaces classes.NET or web services
Definition of variables on databases on proprietary systems	Definition of variables in standard databases with support for SQL and ADO.NET
Does not allow hot swap (swapping the project configuration without stopping the application)	Allows hot updates with built-in version management.

On this page

Object Model and Namespaces

More advanced than most systems, where you must create Tags or Variables for all internal properties and custom logic for your projects, FactoryStudio allows your application(s) to directly access all the business objects that were created in your project. This means that user-created temporary tags are not required to manage the status of PLC network nodes, the total number of alarms in a group, or the number of rows in a dataset. You can now access runtime objects, business objects (representing a network node), an alarm group or dataset, and display required information or take action directly through their built-in properties.

FactoryStudio has an underlying .NET object model, 100% managed code, specifically targeting the development of Real-Time data management applications. The hierarchical object model includes the following top-level objects, which correspond to the main modules in FactoryStudio:

Tags	Dataset
Historian	Script
Security	Server
Alarm	Client
Device	Info

That top-level hierarchy is implemented as .NET Namespaces. Each Namespace has the .NET classes and objects created when building a project configuration. Besides having the configuration settings, those objects also have runtime properties, methods and status.

For instance that Tag namespace has all the tags in the application and each tag has built-in field properties such as Quality, TimeStamp, Min, Max, Units and many others.

Examples

Tag.tagname1.bit0, tag.tagname2.timestamp
The same concept of the tag fields applies to all namespaces, for instance:
Alarm.TotalCount:, Alarm.Group.Warning.Disable:

When building the project configuration, filling input fields or creating scripts, the system always has the Intellisense auto-completion, which guides you to the existing properties that are allowed to use according to what you are editing. This feature allows you to easily "drill down" to a specific property.

When accessing a project object in the .NET Script Editor, it is necessary to prefix the namespace with "@" symbol in order to avoid conflict with the .NET local variables names.

Examples

In Script-Tasks and CodeBehind, use:
@Tag.Analog1
@Device.Node.Node1.Status

The @ symbol is not necessary on Grids and Dialogs. Some input fields may require objects of only one type, such as Tag or Display, the Intellisense will automatically guide you to the allowed objects.

For some users that don't have previous experience in .NET or similar object-oriented systems, those concepts are abstract at the beginning, but when learning the engineering configuration tools and the FactoryStudio modules, the power of those concepts will be clear. What is completely sure is that when getting used with object models and Intellisense, there is a huge productivity increment and you no longer accept working with systems lacking those features.

SQL to the Core

Built-in Embedded SQL

Every FactoryStudio system includes a full-featured embedded SQL engine. This provides several advantages including:

- A safe and secure location for your entire project configuration.
- It can be used as the historian database to log tags, alarms and events on small to medium systems (up to 10GB of data).
- On Large systems, it can be used as a local Store and Forward location, when the remote database is not available.
- It provides an ideal system to store local runtime settings, retentive information, local recipes, schedules and tables and queries when preparing reports.

Multiple Database Connections

FactoryStudio provides seamless integration with any third-party database, including Microsoft SQL Server, Oracle, MySQL, Informix, SQL Lite, PostgreSQL and others.

Real-time Queries and Tables

Any data source with ODBC, OLE-DB or ADO. NET support can be connected with the application; an easy syntax allows you to create or customize any query or table search with real-time tags.

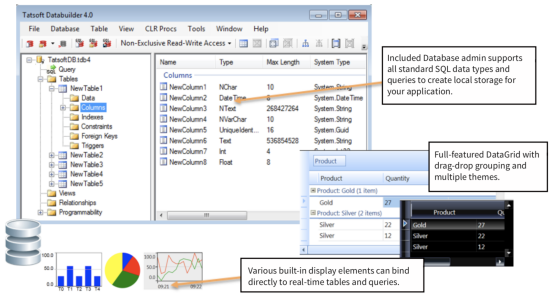
Advanced DataGrid

FactoryStudio provides a fully-featured DataGrid object to present tables and queries from databases, as well as show contents of any tag, asset or real-time object. Just drop the table, query or tag to the Grid Data Source to create front-end visualization or edit any real-time object or database.

Client-Server Architecture

The Convergence of SQL database, HMI software and .NET

Scheduling, Process Recipes and real-time data consolidation made easy.



Real-time queries can be processed either at the server or from the client computer, asynchronously or synchronously. To achieve better performance, multiple requests from distributed clients are cached and synchronized at the server.

Data Gateway

Connecting client queries through Firewall protected security zones, such as moving data between the Automation Network and the IT network, is no longer an issue. FactoryStudio provides a built-in firewall friendly data gateway. Data queries from clients are routed in a secure way through any FactoryStudio system.

.NET to the Core

Built-in Code Editor

FactoryStudio includes an integrated script editor for developers to create custom functionality for the application. The editor provides a powerful set of tools to help you test and evaluate your scripts. Debugging tools include assigning breakpoints, stepping into code, stepping over code, executing line by line and watch values of objects changing with each step.

Scripts are executed natively as managed code within the .NET framework, meaning you cannot create a script that would inadvertently cause the system to shut down. This provides a greater level of security and up-time for your application.

Server and Client Domains

Using the FactoryStudio script editor you can create scripts that execute on the server for global reach, or they can execute on the client side for local reach.

Tasks, Classes and Expressions

You can create Tasks, .NET classes and function libraries. In some cases it may be more efficient or desirable to create one- line expressions, rather than full methods. For that purpose, FactoryStudio provides an expression editor allowing access to all .NET operands and classes.

Object Orientation, Project Elements

All project elements, including Tags, Datasets, Alarms, devices, status of communications are immediately accessible via IntelliSense, as they are native .NET objects, no temporary tags calls are required. With a single move data from Tags to .NET external Data-tables.

C#, VB.NET and Code Translation

FactoryStudio includes industry standard languages of C# and VB.NET so engineers no longer have to suffer through using old proprietary, single-threaded or interpreted scripting editors. Translate your code between C# and VB.NET anytime to better leverage your expertise.

Events and Scheduling

Tasks and expressions can be triggered to execute by date, time, condition, calendar, tag change or interval. Execution is distributed among processes, each running in its own application domain, isolated from the real- time database, for maximum system security and performance.

Complete debugging tools

Online project changes and configuration while running and debugging.