Building Solutions

Understanding the configuration model for FrameworX solutions.
Platform **Architecture** Workspaces | Building Solutions | Deployment

Overview
Key Principles
The Four Pillars
How Modules Connect to UNS
Getting Started

Overview

The Core Concept

When building FrameworX solutions, there's one essential principle: **everything connects to the UNS**. The UNS (Unified Namespace) is the central data layer. You define your process variables (Tags) in the UNS, and all modules reference those Tags. Modules don't connect to each other — they all connect to the UNS.

FrameworX manages the complete Solution Lifecycle, from development through production, using three integrated workspaces::

UNS (Unified Namespace / Real-Time Tag DB)
Your Tags • Built-in Namespaces • Dynamic TagProviders.

All modules connect to the UNS — not to each other.

Industrial Operations	Business Modules	UI Modules.
Devices	Datasets	Displays
Alarms	Reports	Security
Historian	Scripts	

Key Principles

Principle	What It Means
UNS is the center	All modules connect to UNS, not to each other
Modules are independent	Each module has its own configuration — configure in any order
Tags first (conceptually)	Define your data in UNS, then modules reference those Tags
Flexibility in practice	Start anywhere — many begin with Displays using placeholder data, connect to UNS later

The Four Pillars

The Four Pillars provide a framework for learning and developing solutions. Each pillar groups related modules by their purpose.

Pillar	Focus	Modules	What You Configure
1. Data	Your namespace	UNS	Tags, UserTypes, Folders, TagProviders
2. Industrial Operations	Factory floor	Devices, Alarms, Historian	Field communication, alarm conditions, time-series storage
3. Business	Logic & workflows	Datasets, Reports, Scripts	SQL access, document generation, custom code
4. UI	Operator interface	Displays, Security	Screens, layouts, users, permissions

You don't have to build in this order. The Four Pillars is a learning framework, not a required sequence. Many integrators start with Displays to prototype the UI, using simulation data, then connect to real Tags later.

How Modules Connect to UNS

Every module references Tags from the UNS. The module configurations specify *which* Tags to use — the UNS provides the *data*.

Module	How It Uses Tags
Devices	Points map Tags device addresses
Alarms	Items monitor Tags for alarm conditions
Historian	HistorianTags record Tag values over time
Datasets	Queries include Tag values via `{TagName}` syntax
Scripts	Code reads/writes Tags directly
Displays	Elements bind to Tags for visualization and input

Tag Access Syntax

All modules and displays access Tags using consistent syntax:

Location	Syntax
Root folder	`Tag.TagName`
In hierarchy	`Tag.Folder1.Folder2.TagName`
UserType member	`Tag.Motor1.Speed`
Tag property	`Tag.Motor1.Speed.Quality`
Asset syntax (equivalent)	`Asset("Folder1/Folder2/TagName")`

Configuration Independence

Because modules connect to UNS and not to each other, you can:

- Configure modules in any order
- Change one module without affecting others
- Test modules independently
- Add new modules to existing solutions

This architecture makes solutions maintainable and scalable.

Getting Started

Step	Action
1	**Create a Solution** in Solution Center
2	**Define Tags** in UNS (or start with Displays using `Tag.PlaceHolder`)
3	**Configure modules** as needed — order doesn't matter
4	**Run** to test in Runtime
5	**Iterate** — modules are independent, change any without affecting others

---

Next Steps

Data Foundation — Configure Tags, UserTypes, and TagProviders

Modules — Explore each module's capabilities

Object Model Quick Reference — Complete reference of all objects, dependencies, and minimum configuration

Tutorials — Step-by-step learning guides

In this section...