FrameworX Enterprise IT Readiness Reference

Install, identity, audit, and monitoring posture for IT teams operating FrameworX in enterprise and regulated environments.

Technical Reference **Solution Lifecycle Reference** FrameworX Enterprise IT Readiness Reference

Version 10.1.5+

This reference consolidates the IT-side story for FrameworX: how the product installs at scale, authenticates operators and AI agents, emits audit evidence, and fits an existing enterprise monitoring stack. It is the operations-time companion to the engineer-focused DevOps and Version Control Reference.

Overview

The Enterprise IT Readiness Reference is the operations-time companion to the DevOps and Version Control Reference. Engineering teams managing solution change use the DevOps reference (Track Changes, JSON / Git export, impact analysis — author-time concerns). IT teams deploying and operating the product use this page (install, identity, audit, monitoring — operations-time concerns). The two paths overlap on audit and compliance and cross-link there.

Audience: IT administrators, security officers, and compliance leads at customer organizations. The page is a navigation hub — each section either summarizes the platform posture or points at the deep reference page that documents it.

Three Workspaces and where AI lives

FrameworX has three workspaces. Each has a different role in the lifecycle, a different identity model, and a different AI tool surface.

Workspace	Purpose	AI tool surface (10.1.5)
Solution Center	Solution management, licensing, server configuration	None
Designer	Solution authoring — UNS, modules, displays, scripts	DesignerMCP and ConsoleMCP — config-time, author identity
Runtime	Solution execution — TServer and TWebServices	RuntimeMCP (STDIO) and RuntimeMCPHttp — operations-time, service or operator identity

The two MCP families serve different jobs, run on different hosts, authenticate differently, and need different audit. A Designer-side write to a tag definition is a config change; a runtime-side tool call that reads a tag value is an observation. Different audit posture, different threat model.

Identity Model Today

Workspace or channel	Identity in 10.1.5 GA
Designer	Windows Authentication. External IdP integration is not in scope for Designer; that path stays on Windows Auth indefinitely.
Runtime operator login	Local users plus optional LDAP, configured under **Security** **Users & Groups**.
Runtime MCP — Chain A (external AI client)	Service-account connection. No operator identity flows through Chain A in 10.1.5.
Runtime MCP — Chain B (operator chat embedded in TServer)	HMI-authenticated operator through the existing Display security stack.

OIDC and OAuth 2.0 support for runtime operator login lands in 10.1.6 — see the roadmap section below.

MCP Scoping — the 5 gate bits

The runtime tool surface is gated by five bit flags on **Solution Settings Data Servers MCP for Runtime**. These are the security officer's primary control surface. There is no per-tool ACL and no role-based binding from operator identity to specific tools — the five bits are the scope model.

Setting	What it gates
Enable Custom Tools	**The only mutation control.** Toggle off and the runtime MCP surface is read-only by construction; an AI client cannot effect any change in the running solution. Toggle on to register solution-authored tools that perform writes (tag updates, command sequences, protocol calls).
Enable Runtime MCP	Master switch. With this off, no MCP client can connect.
Enable UNS Tools	Five read-only tools that walk the Unified Namespace: read tag values, browse the object model, search, describe an object, list instances.
Enable Alarm Tools	Read-only access to live alarms and to alarm history. Historical queries are constrained to `SELECT` — the dispatcher rejects any other SQL.
Enable Historian Tools	Read-only access to historian time-series data.

For deployments where the solution does not author Custom Tools, leaving **Enable Custom Tools** off makes the runtime MCP surface read-only by construction. No further controls are required to keep AI clients from changing plant state.

Three Audit Sinks

FrameworX emits audit evidence into three sinks. Each is owned by a different layer and has a different regulatory weight. The platform does not federate them into a single internal log — that integration belongs at the customer SIEM (Splunk, Datadog, Azure Sentinel, the Elastic Stack).

Sink	Owner	Scope	Regulatory weight
Kernel AuditTrail	Runtime kernel	Runtime control actions, alarm acknowledgement, tag changes, refused MCP connections, Custom Tool execution. RTDB-bound.	High — this is the regulatory log.
Designer LogMCP	Designer	Designer-side AI activity. Internal diagnosis aid.	Low — not a regulatory record.
Windows System Event Log	TManageServices and Windows Service Control Manager	Install, uninstall, service start, service stop, unexpected service termination.	Medium — SCOM, Datadog, Azure Monitor, Splunk, and PRTG pick this up automatically without custom configuration.

The customer points each sink at their collector and joins on timestamp, host, and correlation tokens at the SIEM layer. Making FrameworX an internal aggregator would build a half-baked SIEM and a new attack surface; both are deliberately out of scope.

MCP Audit for Custom Tool Execution

Every Custom Tool execution by an MCP-connected client is recorded in the kernel AuditTrail under the **MCPCustomTools** category. The category is enabled per solution from **Alarms Global Settings AuditTrail**; same enable-flag model as every other AuditTrail category.

What is logged:

- Tool name — the user method invoked.
- Arguments — full JSON, recorded in the audit row. Mirrors the LoadDatasets precedent already used for SQL queries against datasets.
- Caller chain — **A** for an external AI client connected through RuntimeMCP or RuntimeMCPHttp, **B** for the TServer-embedded operator chat.
- User — the HMI-authenticated operator for Chain B; the literal token `MCP` for Chain A in 10.1.5 (Chain A carries no operator identity until OIDC ships in 10.1.6).
- Result status — OK or error.

What is not logged: read-only MCP tool calls (tag reads, object-model browse, UNS search, alarm and historian reads). Reads are not regulated by FDA 21 CFR Part 11, NERC CIP-007, or IEC 62443 SR 6 — auditing every read produces noise no compliance officer asked for.

Privacy posture. Arguments are recorded in cleartext, mirroring how every other AuditTrail category that captures input data behaves — LoadDatasets logs the full SQL, TagChanges logs old and new values, OperatorActions logs tag-write payloads. Privacy is gated by the AuditTrail enable-flag, not by the message content. Hashing was considered and rejected: it would be the only category in the audit framework that hides input, and a hash cannot be diffed against a later support-ticket complaint.

Refused MCP connections are already audited under the existing **RemoteConnections** AuditTrail category — client IP, module, username, and refusal reason. No additional configuration required.

Compliance Hooks

FrameworX is validated by customers in regulated deployments. The platform provides the audit and configuration evidence the regimes below require; the customer maps their own controls and their own validation evidence to that.

FDA 21 CFR Part 11 §11.10(e)

Audit-trail record of control actions by authenticated users. The kernel AuditTrail (TagChanges, OperatorActions, RemoteConnections, MCPCustomTools) provides the record. Electronic-signature requirements (§11.50, §11.70) are covered by the existing **Security Policies Electronic Signature** configuration and remain independent.

IEC 62443 SR 6.1 — Audit Log

Audit log entries for control actions by users and processes. The same three AuditTrail categories cover users (Chain B operators, runtime login) and processes (Chain A service accounts).

NERC CIP-007 R4 — Security Event Monitoring

Logging access and changes to Bulk Electric System cyber assets. Refused MCP connections, MCP Custom Tool executions, and Windows Service Control Manager start / stop / termination events together cover cyber-asset access and change.

ISO 27001 A.8.15 — Logging

The three-sink model (Kernel AuditTrail, Designer LogMCP, Windows System Event Log) provides logging across the platform stack. Sink-level retention and forwarding policy is configured by the customer.

Validated by customers in regulated deployments — not certified. Formal product-level certifications (ISO 27001, SOC 2 Type II, IEC 62443-4-1) are on the roadmap; see the roadmap section below.

In-flight 10.1.5 Deliverables

Pages and references that an IT team starting a 10.1.5 deployment will read alongside this one:

- IT Deployment RunBook — the operational install / configure / monitor / update workflow for Windows Server deployments.
- Docker Deployment Guide — container-based deployment.
- TLS and SSL Configuration — transport security for runtime client-server, web client, and REST endpoints.
- TManageServices CLI — script-friendly Windows Service install, uninstall, and lifecycle automation.
- Multi-Language UI Localization — localized operator UI for international deployments.
- Security Hardening Reference — detailed hardening checklist for production runtime servers.

Health Endpoint Posture

Every FrameworX runtime HTTP listener — TServer and TWebServices — exposes `/health` and `/ready` endpoints. The endpoints are designed for use as Kubernetes liveness and readiness probes and for native pickup by SCOM, Datadog, Azure Monitor, Splunk, and other HTTP-probing monitoring tools.

Endpoint	Purpose	Healthy response	Starting / unhealthy response
`/health`	Liveness probe — "is the process up and responsive"	HTTP 200, `status:"healthy"`, real version, real uptime	HTTP 200, `status:"starting"` during initialization; HTTP 503 if the listener throws while building the snapshot
`/ready`	Readiness probe — "is the process ready to serve traffic"	HTTP 200, `IsReady:true`	HTTP 503, `IsReady:false` during initialization

Both endpoints return JSON with `Content-Type: application/json` and `Cache-Control: no-store`. The body carries the product version, uptime in seconds, and process start timestamp — enough for a probe to differentiate "service restarted recently" from "service has been up for hours."

Per-module status detail (a `moduleStatuses` dictionary listing each runtime module's individual health) is not part of the 10.1.5 endpoint payload; that field lands in a future release. For 10.1.5, the global `status` and `IsReady` fields are the readiness contract.

For deployments that need finer-grained insight than the HTTP probes provide, the Runtime System Monitor exposes live per-module metrics, client session counts, memory, and tag throughput.

Roadmap Pointer

Capabilities planned for the next minor releases. Detail will land on dedicated reference pages as each capability ships; this section is a stub-level pointer.

OIDC for runtime operator login (10.1.6)

OpenID Connect support for runtime operator login through the HTML5 / WebAssembly client. Schema, Designer UI under **Security Identity Providers**, and runtime dispatch all land in the same release. End-to-end smoke-tested against a verified-target IdP. Identity propagation to MCP Chain A is the operationally interesting consequence: an HTML5 client logged in via OIDC carries that identity through to subsequent MCP calls, and the connected user surfaces in the Custom Tool audit row instead of the literal `MCP` token.

OAuth 2.0 generic (after OIDC)

Generic OAuth 2.0 client support, configured through the same Identity Providers list as OIDC. Same operator UX, different engineering wiring (no discovery endpoint, manual JWKS handling).

Fleet management

Multi-node management — monitor status, deploy solutions, batch start / stop, fleet-wide updates across N FrameworX installations. Specification phase. No 10.1.5 commitment.

ProgramData logs and Windows Event Log integration

A custom **FrameworX** Windows Event Log source for granular events — license failure, solution load error, module exception — with optional default routing of operational data under `C:\ProgramData\Tatsoft\FrameworX\`. In 10.1.5, Windows Service Control Manager already logs basic service start, stop, and unexpected termination events to the System log under source `Service Control Manager` for any installed FrameworX service; SCOM, Datadog, Azure Monitor, and PRTG pick those up natively.

winget package

winget convenience package layered over the existing signed silent installer. The 10.1.5 silent installer already integrates directly with SCCM, Intune, Ansible, and PowerShell DSC; winget is a future convenience layer, not a blocker.

In this section...
