

Modernizing EZ-EDS for the Future of Device Description Development

Vivek Hajarnavis, Rockwell Automation
Jim Jarrett, Rockwell Automation
Joakim Wiberg, ODVA Inc.

October 21, 2026

Agenda

- Why EZ-EDS needs to change
- EDS improvements and new minimum requirements
- Developer and integrator workflows
- Usability improvements: interface and workflow
- Automated EDS generation, and an open question

© 2026 ODVA, Inc.
All rights reserved.

EDS files are the contract between a device and the tools

Describes the device

Supported CIP objects, attributes, services, connection types and data formats

Consumed by interpreters

Scanners and connection originators parse it to recognize and configure the device

Quality matters directly

EDS accuracy and completeness drive the commissioning experience

© 2026 ODVA, Inc.
All rights reserved.

A circular dependency stalled optional CIP functionality

Control vendors

Reluctant to implement a feature until enough devices offer it

⇄

Device vendors

Little reason to implement a feature control-side tools don't yet support

Neither side moves first — and with no standardized minimum requirements for EDS Interpreters, each vendor picked its own bar, and device vendors often stuck to only the most basic, widely supported keywords.

© 2026 ODVA, Inc.
All rights reserved.

New requirements bring new EDS needs

Non-modular design

Devices without modular components

No bridging or routing

Standalone units, not gateways or bridges

Static assemblies

Fixed configurations, not runtime- or user-defined

Edition 3.40 of CIP Networks Library Volume 1 (April 2026) put this bar into effect: clarified spec text, explicit interpreter-vs-device obligations, and support requirements split by who benefits.

Two use cases anchor the new requirements

USE CASE 1

Parameter access & visualization

Monitor and configure real-time values, organized into logical groups. A 16-bit unsigned integer is presented as such; enumerations show defined labels, not raw numbers; bit-level parameters show named bits.

USE CASE 2

Class 1 connections

Tools and originators parse and interpret input, output, and configuration assemblies — presenting data consistent with the meaning defined in the EDS file, and supporting configuration by end users.

© 2026 ODVA, Inc.
All rights reserved.

Requirements updates

Entry Keyword	EDS Interpreter Requirement	Device EDS Requirement
ParamN	Required	Required
ConstructedParamN	Required	Optional
EnumN	Required	Conditional

Not every device has data that needs ConstructedParamN, so device vendors aren't obligated to include it. But an EDS Interpreter must work with *any* compliant device — so it must support it unconditionally.

© 2026 ODVA, Inc.
All rights reserved.

What this means for EZ-EDS

Continuous validation against Device EDS requirements

Evaluate continuously

Check the open EDS file against the full set of Device EDS requirements, not just at save time

Show what's missing

A clear, actionable list of required elements not yet present in the file

Catch gaps early

Structural omissions surfaced during authoring, not late when remediation is costlier

EZ-EDS is a Device EDS authoring tool — so it's the Device EDS requirements, not the Interpreter requirements, that drive these changes.

© 2026 ODVA, Inc.
All rights reserved.

Two development paths. One tool that no longer fully covers either.

Manual editing

IDE or GUI-based — what most developers use, and for many, all they'll ever need

→

Automated generation

CLI-driven, programmatic — often building on a file already started in a GUI tool

© 2026 ODVA, Inc.
All rights reserved.

Usability improvements

EZ-EDS solved the problem it was built for — but expectations for tool usability have moved on since, and no systematic effort has revisited its interface since launch.

Going forward, EZ-EDS must be refreshed and expanded to support both manual and automated workflows. The EZ-EDS code base, while well written, is tied to a very old development environment with no straightforward migration path. A promising foundation already exists: the Conformance Testing tool's EDS parser and validator — fully separable and reusable.

© 2026 ODVA, Inc.
All rights reserved.

From Multiple Document Interface to tabs

TODAY — MDI

Child windows nested inside one parent window. Common in the early 2000s, but a practical limitation in multi-monitor environments.

→

PROPOSED — TABS

The dominant pattern across browsers, IDEs, and modern Office apps. Near-zero learning curve; tabs can detach into independent windows on demand.

Microsoft moved Office from MDI to SDI in 2013. A tabbed model scales gracefully whether two or twenty files are open.

© 2026 ODVA, Inc.
All rights reserved.

From a blocking modal to a persistent diagnostic panel

TODAY — MODAL

Errors and warnings block the app. Users must memorize the list, dismiss it, then navigate to find each issue — and re-trigger validation if they lose track.

→

PROPOSED — PANEL

Docked at the bottom, always visible. Entries are clickable and jump to the source. Filterable by severity and by section or keyword.

The panel doubles as an at-a-glance view of the file's current compliance state — not just a tool for the moment an error appears.

© 2026 ODVA, Inc.
All rights reserved.

A second mode of development, built around a CLI

Batch update

A companion file applies a common set of changes across many existing EDS files — add what's missing, update what already exists

→

Generation from spec

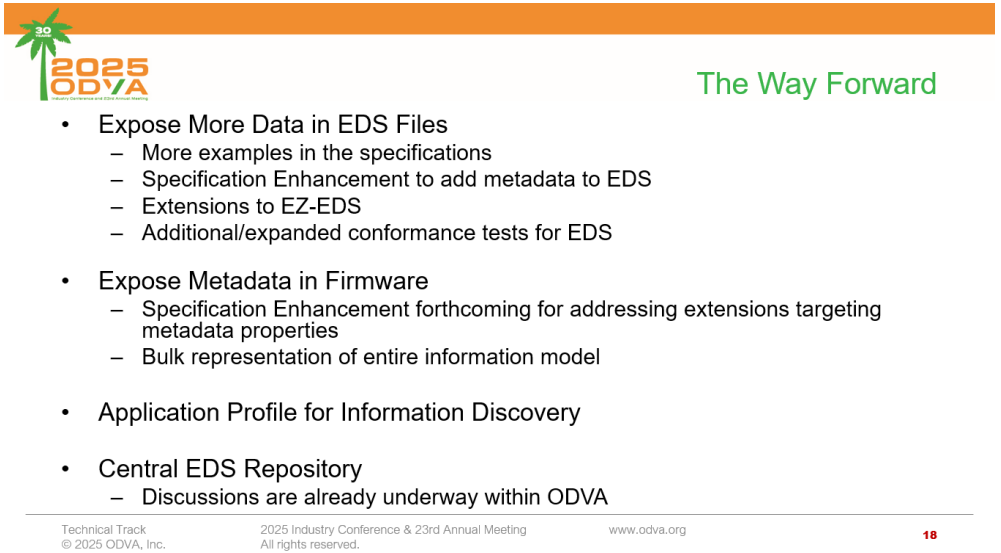
A structured input file produces a complete, valid EDS — invoked by a developer or rebuilt automatically with every CI/CD run

For product families and CI/CD environments, manual GUI authoring isn't the right fit. A structured input — JSON or XML — becomes the source of truth, with the EDS file produced as a derived artifact.

© 2026 ODVA, Inc.
All rights reserved.

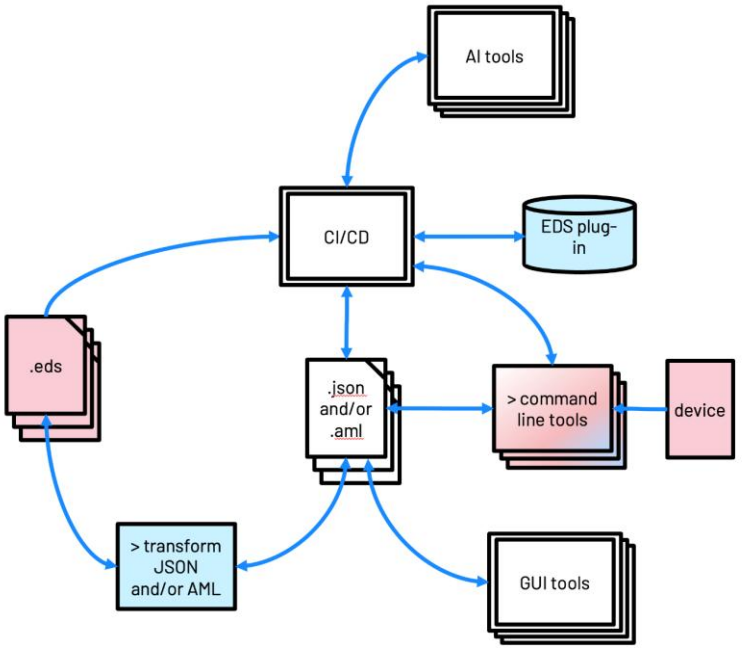
Supporting new requirements

There needs to be an easy way to achieve new upcoming EDS requirements.



Supporting new requirements and modern workflows

There needs to be ways to integrate creation of EDS files into CI/CD systems.



© 2026 ODVA, Inc.

All rights reserved.

Conclusion

- **Requirements are tightening.** Optional EDS content is becoming mandatory for conformance. Tooling must make conformant EDS files easier to produce, not harder.
- **Two development paths, two kinds of user.** Manual editing in an IDE or GUI is where most developers start, and for many it is all they need. Automated generation updates EDS files programmatically, often from a GUI-authored file and often within CI/CD.
- **EZ-EDS covers neither path fully.** Its code base, while well written, is tied to a very old development environment with no straightforward migration path. New tooling will likely n
- **A foundation already exists.** The Conformance Testing tool's EDS parser and validator are fully separate from the rest of its code base, and could be reused for an EZ-EDS successor.

© 2026 ODVA, Inc.
All rights reserved.

The question for this room

What tooling should ODVA provide to make EDS development and maintenance easier — for individual products, and for entire product families?

© 2026 ODVA, Inc.
All rights reserved.


2026
ODVA
Industry Conference and Annual Meeting