IntuitionLabs

INTUITIONLABS / RESEARCH & PRACTICAL GUIDANCE

Define-XML 2.1: Metadata QA and Reviewer Navigation

> Build practical AI for pharma and biotech with IntuitionLabs. We help life-science teams turn complex information and workflows into useful software, governed knowledge systems and AI tools.

Published by IntuitionLabs · 2026-09-20 · define-xml 2.1 · define-xml implementation · metadata validation · cdisc · fda submissions · reviewer navigation · value-level metadata · clinical data standards · submission qa

Original article: https://intuitionlabs.ai/articles/define-xml-2-1-implementation-guide

In this article

Executive Summary

Define-XML 2.1 implementation is not a final XML-generation task. It is a controlled reconciliation of source metadata, submitted datasets, referenced documents, machine-readable XML, and the reviewer-facing rendition. As of September 2026, CDISC labels the current package **Define-XML v2.1.11**, published on April 6, 2026 (cdisc.org). The release aligns its enumerations schema with **Controlled Terminology Package 61** (cdisc.org). Version selection remains a study-level regulatory decision, however. FDA says the format must be one it supported at the study start date (fda.gov), while its June 2026 technical guide is nonbinding guidance that permits another approach satisfying applicable requirements (fda.gov).

Readiness should be decided through four evidence gates. First, reconcile every submitted dataset and variable to metadata. Second, verify semantic traceability through **origin, method, comments, codelists, external-dictionary versions, and value-level metadata**. Third, validate XML structure and applicable conformance rules. Fourth, open the relocated package and complete a reviewer-navigation dry run. FDA expects code lists and origins to be easy to access (fda.gov); implementation guidance places the stylesheet beside define.xml (pharmasug.org). FDA also recommends testing against standards-development-organization rules, technical rejection criteria, and FDA Business Rules (fda.gov). Schema success alone therefore cannot establish reviewability.

The operational artifact in this report is a **field, control, owner, and evidence matrix**. Its highest-risk checks are dataset and variable reconciliation, mixed-origin handling, executable where-clause tests, external-dictionary version agreement, referenced-leaf resolution, and final-folder rendering. Define-XML 2.1 permits origin at variable or value level (cdisc.org); SDTM guidance directs mixed collected and derived values to value-level metadata (cdisc.org). Relative links require a known base URI to remain reliable (datatracker.ietf.org), so link tests must run after package relocation, not only in a generator workspace.

Three project metrics make the release decision reproducible: **metadata completeness = populated applicable required fields / applicable required fields**, **broken-link rate = failed referenced leaves / referenced leaves tested**, and **reconciliation coverage = variables represented in Define-XML / variables**

in submitted datasets. These are sponsor QA measures, not FDA or CDISC pass thresholds. A package is ready only when blocking defects are closed, owner approvals are recorded, and the evidence pack preserves source snapshots, generator configuration, validation output, reconciliation results, and navigation-test results.

Introduction and Background

Define-XML is the metadata layer that makes Study Data Tabulation Model (SDTM), Standard for Exchange of Nonclinical Data (SEND), and Analysis Data Model (ADaM) datasets interpretable. FDA describes the file as metadata for datasets, variables, possible values, controlled terminology, and codes (fda.gov). Separate definition documents are also a cross-regulator packaging pattern: PMDA's English guide describes separate documents for SDTM and ADaM (pmda.go.jp). The practical question is not whether a generator can emit XML. It is whether the metadata truth remains complete, internally consistent, regulator-appropriate, and navigable after every handoff.

CDISC added explicit identification of referenced standards and controlled-terminology versions in 2.1 (cdisc.org). It also revised origin handling and supports dataset subclasses. These capabilities improve traceability, but they create upstream data requirements that cannot be reconstructed reliably at the end of publishing. A metadata repository needs stable identifiers, relationships, version declarations, document targets, and owner-approved descriptive text before generation.

The audience for this Define-XML 2.1 implementation guide is the team approving the handoff: standards leads, statistical programmers, submission operations, and validation owners. It treats **Define-XML 2.1 metadata validation**, **Define-XML 2.1 validation rules**, **Define-XML metadata quality assurance**, **CDISC Define-XML 2.1 best practices**, and **FDA Define-XML submission requirements** as related but distinct control layers. The consultancy perspective is appropriately adjacent. IntuitionLabs describes an information layer that connects systems to authoritative sources with permissions, citations, evaluation, and accountable operation (intuitionlabs.ai). Applied here, that means treating Define-XML as a governed output of metadata operations, not as a document edited without traceable source control.

Key Changes in Define-XML 2.1

Standards and version declarations

The first gate is an explicit **standards applicability record**. The repository should identify the study, dataset family, implementation-guide version, Define-XML version, controlled-terminology package date, regulator, and study-start-date rationale. Define-XML 2.1 added `def:StandardsOID` so a dataset definition can reference a declared standard (cytel.com). The control prevents a technically valid file from describing its implementation context ambiguously.

Version selection must not be inferred from “latest.” FDA's posted catalog page in this research session identified **March 2025** as the catalog release (fda.gov). Recheck the live catalog immediately before submission because support and requirement dates can change.

Origin, no-data, and nonstandard metadata

Define-XML 2.1 added **Type** and **Source** attributes to `def:Origin` (cdisc.org). For SDTM, the implementation material describes both as required except that Source is not used for a Predecessor origin (cdisc.org). QA therefore needs conditional rules, not a blanket non-null test.

Conditional flags such as `def:HasNoData` and `def:IsNonStandard` need a reason, reviewer impact, and accountable approval in the source repository. The control should test the condition and its explanation together rather than treating either as isolated text.

Dataset class, subclass, and contextual codelists

Version 2.1 reimplemented dataset Class as a child element and added Subclass support (cdisc.org). Migration QA should compare intended class and subclass values, not only element presence.

When codelists vary by context, the documented pattern uses one value list, two item references, and two where clauses (cdisc.org). That makes where-clause correctness a semantic control. Schema validation cannot determine whether the predicate selects the intended records.

Implementation Considerations and Process Changes

Inputs and accountable handoffs

Define-XML generation should begin only after five controlled inputs are baselined:

- **Dataset inventory:** submitted filename, dataset label, class, subclass, purpose, structure, keys, and document location.
- **Variable inventory:** variable name, label, type, length, order, codelist, origin, role, and value-list relationship. Define-XML variable order must mirror the submitted dataset (cdisc.org).
- **Semantic metadata:** methods, comments, aliases, origins, value-level predicates, codelists, and external dictionaries.
- **Document registry:** stable leaf ID, title, relative href, media type, page references, and package destination.
- **Applicability record:** standards versions, terminology dates, regulator context, and study-start-date decision.

Ownership should follow the evidence. Statistical programming owns dataset and variable reconciliation. The standards lead owns model choices and terminology. Document owners approve derivation text, origin, and comments. Submission operations owns folder placement and electronic Common Technical Document (eCTD) metadata. Validation owns independent execution and retained output. The release approver accepts residual explained discrepancies.

Generation as a reproducible build

The generator should consume a versioned snapshot and produce the same output for the same configuration. CDISC's Data Definition Engine demonstrates this separation by reading Data Definition Specification JSON and producing a Define-XML 2.1 file (github.com). Its documentation also exposes post-write schema validation (github.com) and a separate Extensible Stylesheet Language Transformation (XSLT) rendering step (github.com).

A defensible build record includes:

- **Source snapshot ID** and extraction timestamp.
- **Generator name, version, configuration, and dependency lock.**
- **Schema and rule-set versions** actually executed.
- **Checksums** for define.xml, stylesheet, datasets, and referenced documents.
- **Automated logs** with rule identifier, severity, object, and disposition.
- **Human-review record** for the rendered view and every navigation class.

The separation is not tool-specific. The `defineutils` project, for example, exposes HTML rendition and schema validation as separate modules (pypi.org).

Do not silently repair metadata in generated XML. Route the correction back to the authoritative source, rebuild, and preserve the superseded evidence. This prevents the repository and submitted package from diverging.

Field-Level Metadata QA Matrix

Table 1 is the core Define-XML 2.1 compliance checklist. “Required” in the control column means required by the cited source or by the project's approved applicability rules. Sponsor checks are labeled as such.

Field or object	Pre-generation control	Post-generation test	Accountable owner	Retained evidence
Study and standards declarations	Approve study identity, Define-XML version, implementation-guide version, terminology package, and regulator rationale.	Resolve every standards OID and compare displayed versions with the applicability record.	Standards lead	Approved applicability record, rendered standards panel, XML extract (cytel.com)
Datasets	Reconcile one row per submitted dataset, including label, class, subclass, structure, purpose, keys, and relative leaf.	Compare package inventory to every `ItemGroupDef`; test duplicates, omissions, order, and href.	Lead programmer	Bidirectional reconciliation report (pharmasug.org)
Variables	Baseline name, label, type, length, order, role, codelist, and value-list relationship from final datasets.	Compare every dataset column to `ItemDef` and `ItemRef`; verify order equals physical dataset order.	Statistical programming	Column reconciliation and exception disposition (pharmasug.org)

Field or object	Pre-generation control	Post-generation test	Accountable owner	Retained evidence
Origins	Assign Type and conditional Source; use value-level origin for mixed populations.	Trace collected origins to annotated case report form pages and derived origins to methods.	Standards lead and document owner	Origin exception report, sampled trace links (pharmasug.org)
Methods and derivations	Approve readable algorithm text and stable method IDs; record formal expressions where useful.	Verify every reference resolves and each method renders beside the intended item.	Statistical programming	Method-to-variable matrix and rendered screenshots (certara.com)
Comments	Store explanatory text once, with explicit object linkage; require an explanation for conditional no-data assertions.	Detect orphan, duplicated, empty, or unresolved comment references.	Standards lead	Comment-link report and owner approval
Codelists	Reconcile internal terms, decoded values, data types, ordering, and nonstandard flag.	Compare observed values with allowed values and verify codelist links in the rendition.	Standards and programming	Observed-value comparison and rendered-link test
External dictionaries	Record dictionary name, version, and study-level use; reconcile with Trial Summary where applicable.	Compare Define-XML version text with the full TS domain.	Standards lead	Version reconciliation (fda.gov)
Value-level metadata	Identify variables whose metadata changes by record context; specify mutually intelligible predicates.	Execute each where clause against submitted data; test zero-match, overlap, uncovered records, and expected codelist/method/origin.	Statistical programming	Predicate test output and coverage report (pharmasug.org)
Referenced leaves	Register unique ID, title, relative href, target file, media type, and page convention.	Resolve each href from the final folder and verify target title and page.	Submission operations	Leaf crawl and click-test log (pharmasug.org)
Schema and namespaces	Lock approved schema assets and namespace declarations.	Test well-formedness, namespace use, and schema validity with an independent processor.	Validation	Processor version, command, complete output (w3.org)
Stylesheet and rendition	Approve the stylesheet file and colocated relative reference.	Render from the final folder in the supported review environment; compare all sections and links.	Submission operations and validation	Stylesheet checksum, environment record, screenshots (pharmasug.org)

The table makes one distinction essential: **presence is not correctness**. A present where clause can select the wrong records. A valid href can open the wrong document. A populated origin can be assigned at the wrong level. The evidence must therefore combine machine checks with traceable semantic review.

Controlled terminology and version control

As of September 2026, CDISC's current terminology page identifies **March 27, 2026** as the new file version date (cdisc.org). NCI permits CDISC terminology use without licensing restrictions (cancer.gov) and versions it by date while retaining older versions (cancer.gov). This supports a reproducible control: store the exact package date, not “current CT.” The March release reported about **248 new QRS terms and 1,124 new terms** across the named foundational files (cdisc.org). Those counts show why an unrecorded update can materially change validation results.

The NCI Define-XML terminology includes a codelist for dataset origin type (evs.nci.nih.gov). The repository should preserve controlled codes, submitted values, decoded labels, extensibility status, and package date as separate fields. Free-text labels alone are insufficient for deterministic generation.

XML, Package, and Reviewer Navigation Tests

Layered technical validation

Run technical checks in increasing semantic depth:

1. **Parse:** confirm the complete document is well formed and namespaces are usable. `xmllint` can return an error for namespace defects even when a document is otherwise well formed (gnome.pages.gitlab.gnome.org).
2. **Schema:** validate against the approved Define-XML schema package. `xmllint`, for example, supports a named W3C XML Schema file (gnome.pages.gitlab.gnome.org). W3C defines `schemaLocation` as namespace and location URI-reference pairs (w3.org), but describes those locations as hints rather than a command to retrieve a schema (w3.org). A validation job should therefore name and preserve the actual schema asset it used.
3. **Conformance rules:** execute the applicable CDISC rules. CDISC says its 2.1 rules guide construction conforming to the specification and schema (cdisc.org).
4. **Regulator rules:** execute current, applicable technical-rejection and business-rule sets. FDA's resource page identifies Validator Rules **v1.6, December 2022** (fda.gov).
5. **Cross-file semantics:** reconcile datasets, values, versions, documents, and page targets.
6. **Human rendition:** render through XSLT and inspect content, ordering, labels, and navigation. Browser-side XSLT processing imports an XSL file into an `XSLTProcessor` (developer.mozilla.org).

Independent implementations help expose configuration mistakes. lxml supports validation during parsing against XML Schema (lxml.de) and raises an exception on parse-time validation failure (lxml.de). Saxon offers schema-aware XSLT processing (saxonica.com). Tool diversity is a sponsor QA choice, not a regulator mandate.

Final-folder dry run

A reliable dry run starts from a clean copy of the exact handoff directory. FDA says define.xml should not be compressed (fda.gov). FDA also recommends the specified directory structure because it supports automated detection and review preparation (fda.gov). PMDA's English guide similarly places the reviewer guide, define.xml, and stylesheet with the corresponding datasets (pmda.go.jp).

The tester should complete these navigation paths. A separate HTML report can expose links for navigating the data structures (search.r-project.org).

- **Contents to dataset:** every dataset row opens the intended transport file or documented target.
- **Dataset to variable:** variable order, label, type, length, key status, and role match the submitted file.
- **Variable to codelist:** every codelist link resolves, and decoded values match the intended context.
- **Variable to origin:** collected items reach the correct annotated case report form page; use the PDF page convention (pharmasug.org).
- **Variable to method or comment:** the link opens the correct reusable definition, with no orphan references.
- **Value level to predicate:** each predicate is understandable and its contextual codelist, origin, and method render together.
- **Leaf to document:** every relative href resolves after relocation. ICH recommends relative paths for links across documents (admin.ich.org).
- **Rendition completeness:** every dataset, variable, value list, codelist, method, comment, and document section is visible and ordered predictably.

Rendered links depend on correctly formed leaf elements (certara.com). W3C says relative XLink href references must be resolved to absolute form before use (w3.org). The test record should therefore contain the launch location, browser or renderer version, path exercised, expected target, actual target, result, and evidence reference.

Data Analysis and Evidence

No authoritative source establishes a universal acceptable percentage for metadata completeness, broken links, or reconciliation coverage. The following are **project measurements**, intended to make review and approval reproducible.

Table 2 defines the calculations and their decision use.

Metric	Reproducible formula	Scope rule	Interpretation
Metadata-completeness rate	Populated applicable required fields / applicable required fields	Count conditional fields only where their predicate is true; exclude genuinely inapplicable fields with a reason.	Use the exception list, not the percentage alone, to decide release.

Metric	Reproducible formula	Scope rule	Interpretation
Broken-link rate	Failed referenced leaves / referenced leaves tested	Test from the final relocated package and count each unique intended target once.	Any failure can block review even when the overall rate appears small.
Reconciliation coverage	Variables represented in Define-XML / variables in submitted datasets	Compare physical columns for every submitted dataset; report unexpected metadata-only items separately.	The numerator cannot prove semantic correctness, but omissions become measurable.
Where-clause coverage	Records matched by exactly one intended predicate / applicable records tested	Report zero-match, multi-match, and uncovered groups independently.	A perfect schema result cannot substitute for predicate execution.

The metrics are useful because denominators force teams to define scope. For example, a study with 1,000 applicable metadata fields and 990 populated fields has **99% completeness**, but the ten missing fields could all be dataset leaves. The release decision must examine defect type and reviewer impact, not reward a high aggregate score.

Table 3 supplies a severity rubric for consistent triage. It is an editorial implementation choice, not a regulator classification.

Severity	Decision rule	Examples	Release treatment
Blocking	Prevents parsing, schema validation, required reconciliation, package opening, or correct reviewer navigation.	Missing submitted dataset, unresolved stylesheet, broken required leaf, wrong standards version.	Correct and rerun affected gates before approval.
Major	Metadata is present but materially misstates context or traceability.	Wrong origin level, overlapping where clauses, dictionary version mismatch, method linked to wrong variable.	Correct unless an accountable owner documents a regulator-appropriate disposition.
Minor	Meaning remains usable, but consistency or presentation is impaired.	Nonmaterial capitalization drift, ordering inconsistency, awkward but accurate display text.	Correct when feasible and record disposition.
Observation	Improvement opportunity with no demonstrated effect on correctness or navigation.	Evidence naming convention or optional report enhancement.	Track outside the submission block list.

Quantitative evidence about terminology reinforces the need for explicit versioning, but it does not create a pass threshold. The current CDISC download formats include Excel, text, ODM XML, PDF, HTML, and OWL/RDF (cdisc.org). Teams should retain the actual consumed artifact and hash, not only a web-page date.

Defect Disposition and Evidence Pack

Validation output becomes useful only when each finding has an object, owner, decision, and retest. FDA tells sponsors to correct discrepancies or explain meaningful ones in the relevant reviewer guide (fda.gov). An Analysis Data Reviewer's Guide adds context but does not replace a complete Define-XML file (fda.gov).

For each finding, record:

- **Rule and layer:** parser, schema, CDISC conformance, regulator rule, reconciliation, or navigation.
- **Affected object:** study, dataset, variable, value-level item, codelist, method, comment, or leaf.
- **Observed and expected result:** exact values, not a generic failure statement.
- **Severity and rationale:** based on review impact and the approved rubric.
- **Root metadata record:** repository key and source owner.
- **Disposition:** corrected, accepted with explanation, not applicable, or false positive.
- **Retest evidence:** build ID, tool version, timestamp, and result.

The final evidence pack should include the applicability decision, input inventory, source snapshot, build manifest, generated XML checksum, schemas and stylesheets used, rule-set versions, complete validation reports, dataset and variable reconciliation, terminology and dictionary reconciliation, where-clause tests, leaf crawl, rendered-view checklist, defects and dispositions, and named approvals.

Before release, the approval record should answer six binary questions:

- **Applicability approved:** Is the version decision tied to the study and regulator context?
- **Reconciliation complete:** Are all submitted datasets and variables represented and all metadata-only objects explained?
- **Semantics approved:** Are origins, methods, codelists, comments, and where clauses correct for their contexts?
- **Technical checks passed:** Did parsing, schema, conformance, and applicable regulator rules complete on the released build?
- **Navigation passed:** Did every required path resolve from the relocated final folder?
- **Evidence retained:** Can another reviewer reproduce the build, checks, dispositions, and approval?

PHUSE's reviewer-guide package includes a template, completion guidance, examples, and traceability diagrams (advance.hub.phuse.global). IntuitionLabs' public approach likewise emphasizes tracking quality, risk signals, reliability, and support burden before scaling (intuitionlabs.ai). Here, those are evidence-design principles, not claims of a Define-XML product.

Implications and Future Directions

The immediate direction is toward **machine-readable source metadata with separately testable generation, validation, and rendition**. CDISC describes its Data Definition Specification as a structured, version-controlled specification for clinical research pipelines (github.com). This architecture makes controls earlier and more observable: a missing origin can fail repository completeness before XML generation, while an XSLT defect can fail rendition without being confused with the metadata source.

Cross-regulator reuse needs explicit jurisdictional profiles. PMDA's current page identifies validation rules **Version 6.0** for gateway submissions (pmda.go.jp), and its technical materials include XML-structure checks for define.xml (pmda.go.jp). EMA's January 2026 clinical-study-data proof-of-concept remains voluntary (ema.europa.eu). These facts should not be collapsed into a supposed universal mandate.

Automation can expand, but approval should remain evidence based. Automated extraction of page references, origin proposals, or methods can create candidates. Owners still need to approve semantic assignments, execute context predicates, and inspect reviewer paths. The durable improvement is not simply fewer manual steps. It is a shorter path from a failed control to the authoritative metadata record that must change.

Frequently Asked Questions (FAQs)

What is the minimum Define-XML 2.1 validation sequence?

Parse the complete XML, validate namespaces and schema, run applicable CDISC and regulator rule sets, reconcile datasets and variables, execute value-level predicates, resolve leaves from the final folder, render through the packaged stylesheet, and complete reviewer navigation.

Which defects require remediation before submission?

Correct any defect that prevents parsing, applicable validation, complete dataset or variable representation, accurate traceability, target resolution, or usable rendering. Major semantic defects normally require correction. Explain a residual meaningful discrepancy in the appropriate reviewer guide only after accountable assessment; reviewer guides should describe special directions and considerations (fda.gov).

Does schema-valid XML prove FDA readiness?

No. FDA distinguishes its requirements from labels used by standards organizations (fda.gov). Schema validation tests structure. Readiness also depends on catalog applicability, regulator rules, semantic reconciliation, folder packaging, and human navigation.

How should a team test Define-XML reviewer navigation?

Relocate the final package to a clean directory, open define.xml through the intended rendition path, and traverse dataset, variable, value-level, codelist, origin, method, comment, and document links. Retest format and links after migration from the working location (pharmasug.org).

Conclusion

Define-XML 2.1 readiness is an evidence-backed release decision, not the absence of generator errors. The implementation begins with an approved standards profile and governed metadata, continues through reproducible generation and layered validation, and ends only after final-folder rendering and reviewer navigation succeed.

The most effective control set is bidirectional. Every submitted dataset and variable must have the intended metadata, and every metadata object must resolve to a submitted object, approved explanation, controlled term, executable context, or packaged document. Origins, methods, comments, codelists, and value-level metadata need semantic tests because XML structure alone cannot establish their correctness.

Teams should preserve the matrix, metrics, defect decisions, and complete evidence pack with the released package. The formulas in this report make scope and exceptions visible, but they are not regulatory thresholds. The final approval should rest on applicability, traceability, reproducibility, and a successful reviewer dry run using the same files that will be handed off.

Source links

Links cited in this article, in order of first appearance. Full addresses are provided for readers using extracted text.

1. https://www.cdisc.org/standards/data-exchange/define-xml/define-xml-v2-1
2. https://www.fda.gov/industry/study-data-standards-resources/study-data-submission-cder-and-cber
3. https://www.fda.gov/regulatory-information/search-fda-guidance-documents/study-data-technical-conformance-guide-technical-specifications-document
4. https://www.fda.gov/media/153632/download
5. https://pharmasug.org/proceedings/2024/SS/PharmaSUG-2024-SS-377.pdf
6. https://www.cdisc.org/sites/default/files/2023-11/2023_us_interchange_paper_-_your_guide_to_successfully_upversioning_cdisc_standards_0_0.pdf
7. https://www.cdisc.org/standards/foundational/sdtmig/sdtmig-v3-3/html
8. https://datatracker.ietf.org/doc/html/rfc3986
9. https://www.pmda.go.jp/files/000267935.pdf
10. https://intuitionlabs.ai/
11. https://cytel.com/perspectives/its-time-to-move-time-to-move-to-define-xml-2-1/
12. https://www.fda.gov/regulatory-information/search-fda-guidance-documents/data-standards-catalog
13. https://github.com/cdisc-org/data-definition-engine
14. https://pypi.org/project/defineutils/
15. https://pharmasug.org/proceedings/2025/SS/PharmaSUG-2025-SS-276.pdf
16. https://www.certara.com/blog/common-questions-about-define-xml-and-the-stylesheet/
17. https://www.w3.org/TR/xml/
18. https://www.cdisc.org/standards/terminology/controlled-terminology
19. https://www.cancer.gov/about-nci/organization/cbiit/vocabulary/cdisc
20. https://evs.nci.nih.gov/ftp1/CDISC/Define-XML/Define-XML%20Terminology.html
21. https://gnome.pages.gitlab.gnome.org/libxml2/xmllint.html
22. https://www.w3.org/TR/xmlschema-0/
23. https://www.cdisc.org/standards/foundational/define-xml/conformance-rules-define-xml-v2-1
24. https://intuitionlabs.ai/articles/fda-study-data-standards-version-matrix

25. https://www.fda.gov/industry/fda-data-standards-advisory-board/study-data-standards-resources
26. https://developer.mozilla.org/en-US/docs/Web/API/XSLTProcessor
27. https://lxml.de/validation.html
28. https://www.saxonica.com/html/documentation13/schema-processing/index.html
29. https://search.r-project.org/CRAN/refmans/defineR/html/write_define.html
30. https://admin.ich.org/sites/default/files/inline-files/Specification_for_Submission_Formats_for_eCTD_v1_2.pdf
31. https://www.w3.org/TR/xlink/
32. https://advance.hub.phuse.global/wiki/spaces/WEL/pages/26806088
33. https://github.com/cdisc-org/DataExchange-DDS
34. https://www.pmda.go.jp/english/review-services/reviews/0002.html
35. https://www.pmda.go.jp/files/000274654.pdf
36. https://www.ema.europa.eu/en/documents/other/questions-answers-about-clinical-study-data-proof-concept-pilot-industry_en.pdf

THE PUBLISHER

About IntuitionLabs

IntuitionLabs is an AI consulting, custom software development and data engineering firm serving pharmaceutical, biotechnology, medical-device and other life-science organizations. We work with clinical, regulatory, medical-affairs, commercial, quality and IT teams to connect technology decisions with the work people need to accomplish.

AI consulting and adoption

Our AI enablement services cover readiness assessments, use-case selection, governance and policies, team workshops, adoption measurement and ongoing advisory support. We help organizations structure the information layer behind AI: source material, context, permissions and maintained knowledge that make generated answers useful and reviewable. Private LLM inference and hosted AI options support teams evaluating how to operate AI with appropriate control over their data and infrastructure.

Software, data and life-science workflows

IntuitionLabs develops custom software for pharma and biotech, integrates enterprise systems, and builds data engineering and business intelligence solutions. Areas of focus include AI agents, regulatory research, medical writing, medical affairs, CMC information, competitive intelligence and clinical-document workflows. Our eTMF intelligence work includes cross-system reconciliation and inspection-readiness support.

Enterprise platforms and regulated delivery

We provide Veeva services, application support, managed services, integrations and custom applications, alongside enterprise content work involving platforms such as Egnyte. For regulated workflows, our services include GxP enablement, computer-system validation and software development addressing 21 CFR Part 11 requirements. The applicable controls, validation responsibilities and acceptance criteria are defined for each engagement.

Work with IntuitionLabs

Explore AI enablement, pharma and biotech software development, data engineering and BI, and Veeva services. Contact IntuitionLabs to discuss your workflow, information sources and implementation needs.

IntuitionLabs publishes educational research to help life-science teams make informed technology decisions. Coverage of a product or organization does not imply a client relationship, endorsement or partnership.

Website: https://intuitionlabs.ai