PharmaSUG China 2026 - Paper AA-213

From Spreadsheets to Smart Platforms: A Framework for Modernizing SDTM Specification Management Using the Microsoft 365 Ecosystem

Shuting Lou, Pfizer Inc.

ABSTRACT

In clinical programming, Microsoft Excel has long been the primary tool for managing study deliverables such as SDTM mapping specifications, tracking logs, and operational documents. Despite its flexibility, spreadsheet-based workflows present persistent limitations such as fragmented data across multiple files, limited searchability and manual version control.

This paper presents a framework for transitioning from spreadsheet-based processes to an integrated approach using the Microsoft 365 ecosystem — SharePoint Lists, Power Apps, and Power Automate — as a no-code/low-code alternative and demonstrates how these components combine to create a structured, interactive workflow environment for clinical programming teams.

In this paper, we present a SharePoint-based SDTM Specification Explorer. SharePoint Lists replace Excel-based metadata storage with structured, filterable, version-controlled specification data. Power Apps provides an intuitive interface that enables users to browse and filter across domains, variables, and SDTM IG versions, while also surfacing cross-domain relationships and mapping dependencies that are difficult to trace in traditional multi-tab workbooks. Power Automate introduces event-driven automation such as notifications triggered by specification updates, keeping all team members aligned in real time. With this platform, it is also possible to extract a define-ready specification in one-click.

This paper highlights how leveraging existing enterprise tools can significantly improve efficiency, traceability, and usability in clinical programming workflows, without requiring custom software development. The proposed approach provides a generalizable model for modernizing spreadsheet-driven processes across different studies and organizations.

INTRODUCTION

Background

Statistical programming in the pharmaceutical industry relies heavily on standardized data specifications. The Study Data Tabulation Model (SDTM) mapping specifications serve as the foundational document defining how raw clinical trial data are transformed into standardized, regulatory-compliant datasets. These specifications are referenced throughout the study lifecycle — from study setup, programming, validation, to submission.

Traditionally, SDTM specifications are maintained as multi-tab Microsoft Excel workbooks, where each domain (e.g., DM, AE, LB) occupies a separate worksheet/workbook, supplemented by additional tabs for version history, change logs, controlled terminology, and domain mapping summaries. While Excel provides a familiar and flexible environment, its limitations become increasingly apparent as studies grow in complexity and specifications evolve across SDTM Implementation Guide (IG) versions.

Pain Points of Spreadsheet-Based Workflows

Although Excel-based specifications are widely adopted in clinical programming, several limitations are commonly encountered in day-to-day use:

1. **Fragmented data structures.** A typical SDTM specification workbook contains 30+ worksheets covering domain definitions, variable metadata, mapping rules, version history, etc. Information about a single variable may be distributed across different tabs. Cross-referencing requires opening multiple tabs and manually correlating data.
2. **Limited searchability.** Excel's built-in search function operates on a single worksheet at a time. Locating a specific variable across domains or finding all variables that reference a particular controlled term, requires either repeated single-tab searches or maintenance of separate index sheets.

3. **Difficulty managing study-level spec changes from standard.** Each study typically maintains its own copy of the standard specification, with study-specific modifications embedded inline. Comparing how different studies handle the same variable, or identifying patterns of spec change across studies, becomes a manual archaeological exercise.

These limitations not only reduce productivity but also introduce risks to data quality, traceability, and consistency across studies.

Objective

This paper proposes a no-code framework that addresses these challenges by leveraging the Microsoft 365 ecosystem — specifically **SharePoint Lists**, **Power Apps**, and **Power Automate** — to build an interactive **SDTM Specification Explorer**. The framework explicitly separates the standard specification from study-specific implementations and integrates structured review workflows. The remainder of this paper describes the architecture, implementation details, key user scenarios, additional capabilities, and practical considerations for adopting this framework in a clinical programming environment.

THE MICROSOFT 365 ECOSYSTEM

The Microsoft 365 platform provides a suite of integrated, enterprise-grade tools that, when combined, offer a powerful no-code/low-code alternative to traditional spreadsheet-based workflows. Three components form the foundation of the proposed framework: SharePoint Lists, Power Apps, and Power Automate. Each serves a distinct purpose, and their combined value lies in seamless integration through a unified data layer.

SHAREPOINT LISTS

SharePoint Lists function as the underlying data store. Unlike Excel worksheets, SharePoint Lists are structured, schema-defined data containers that natively support:

- **Typed columns** including text, number, choice, date/time, person, lookup, and yes/no.
- **Built-in version history** that automatically captures every modification, including the user, timestamp, and previous values, without additional configuration.
- **Role-based permissions** managed at the list, view, or even item level, allowing fine-grained access control aligned with organizational structure.
- **Conditional formatting** through JSON-based column and row formatting, enabling visual cues (e.g., color-coded badges for variable Core status) without writing application code.
- **APIs and integration endpoints** that allow consumption by other Microsoft 365 tools as well as external systems.

Critically, SharePoint Lists scale to up to 30 million items per list, with optimal performance for list views containing fewer than 5,000 items per query — a constraint that is well within the size of typical SDTM specifications.

POWER APPS

Power Apps is a low-code platform for building custom business applications with a drag-and-drop interface. It uses Power Fx, a formula language similar in spirit to Excel formulas, making it accessible to users without traditional software development experience. Power Apps can connect directly to SharePoint Lists as a data source, enabling rich, interactive user interfaces to be built on top of list data.

Key capabilities relevant to this framework include:

- **Gallery controls** for displaying collections of items as lists, cards, or grids with built-in filtering and sorting.
- **Navigation between screens** with parameter passing, enabling multi-step workflows.
- **State management** through global variables, context variables, and collections.

- **Conditional rendering and formatting** based on data values, enabling dynamic visual cues such as color-coded badges or status indicators.
- **Form controls** for structured data entry with built-in validation.

POWER AUTOMATE

Power Automate is a workflow automation engine supporting event-driven triggers and a wide library of pre-built connectors. It can react to changes in SharePoint Lists (e.g., item created, item modified) and execute downstream actions such as sending notifications, routing approvals, or updating other systems.

Common patterns supported include:

- **Event-driven notifications**: triggered emails or Microsoft Teams messages when specification items are created, modified, or status-changed.
- **Approval workflows**: structured multi-step or parallel approvals routed through Microsoft 365 Approvals connector.
- **Scheduled operations**: nightly aggregation jobs or periodic reminders.

Why These Three Together?

The combined value of these three components lies in clear **separation of concerns**: data storage, user interaction, and workflow automation, each handled by a tool optimized for its purpose. This modular architecture enables incremental adoption and continuous evolution as team needs grow.

ARCHITECTURE OVERVIEW

The proposed SDTM Specification Explorer follows a **three-layer architecture** (Table 1), with each layer mapped to one of the Microsoft 365 components:

Layer	Component	Responsibility
Data Layer	SharePoint Lists	Structured storage of specification metadata, version history, and audit trails
Interface Layer	Power Apps	User-facing interactive application for browsing, filtering, and tracking specifications
Automation Layer	Power Automate	Event-driven workflows for notifications, change tracking, approvals, and aggregation

Table 1. Three Layer architecture

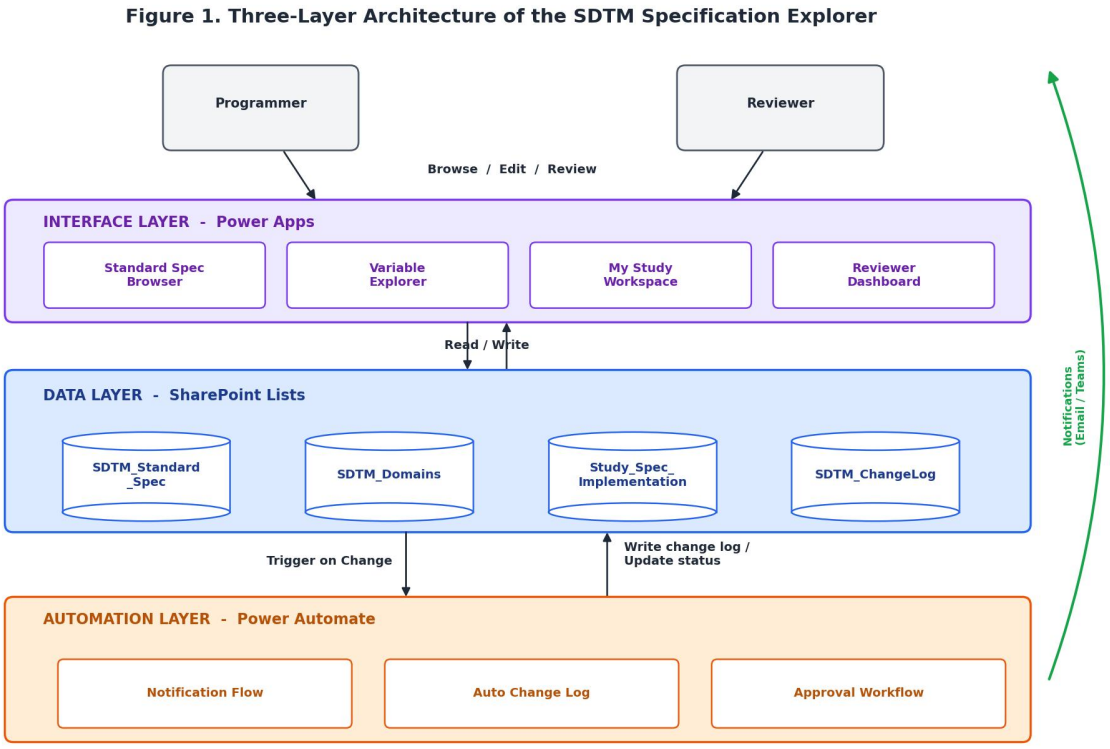

Figure 1. Three-layer architecture of the SDTM Specification Explorer.

Users interact through the Power Apps interface layer (top), which reads from and writes to the SharePoint Lists data layer (middle). Changes to the data layer trigger Power Automate flows (bottom), which write back to the change log list and deliver notifications to users.

Users interact exclusively through the Power Apps interface, which queries SharePoint Lists in real time. When a specification record is updated — either through Power Apps or directly in SharePoint — change events automatically trigger Power Automate flows, which deliver notifications, append entries to a dedicated change log list, and route approval requests when needed. This workflow is designed to capture changes made through the governed application and list interfaces, maintain a centralized history of specification updates, and reduce reliance on manual notification processes. In a production deployment, additional controls—including flow monitoring, permission management, retention configuration, and exception handling—would be required to support reliable operation.

A key advantage of the proposed architecture is its modular structure. Each layer can be extended with limited impact on the others, provided that shared schemas and interfaces are managed consistently:

- The data layer can be extended with additional lists, such as ADaM specifications, controlled terminology, or study-tracking metadata.
- The interface layer can incorporate additional screens or views as new user requirements emerge.
- The automation layer can include additional flows for approval routing, scheduled reporting, export, or integration with other systems.

This structure supports incremental adoption. However, changes to list schemas, lookup relationships, connectors, or workflow states may require corresponding application and flow updates; the layers are therefore logically separated rather than completely independent.

DATA LAYER: SHAREPOINT LISTS

SCHEMA DESIGN OVERVIEW

The data layer is built on four SharePoint Lists, each serving a distinct purpose (Table 2):

List	Purpose	Approximate Size
SDTM_Standard_Spec	Variable-level metadata for the standard specification	~600 rows per IG version, all domains
SDTM_Domains	Domain-level metadata (class, structure, key variables)	~30 rows per IG version
Study_Spec_Implementation	Study-specific spec changes, acknowledgments, and new variables	Variable per study (typically 5–50 per domain)
SDTM_ChangeLog	Auto-populated change tracking	Grows over time

Table 2. Schema Design Overview

SDTM_STANDARD_SPEC — VARIABLE-LEVEL METADATA

This list serves as the single source of truth for the standard SDTM specification. Each row represents one variable within one domain within one IG version.

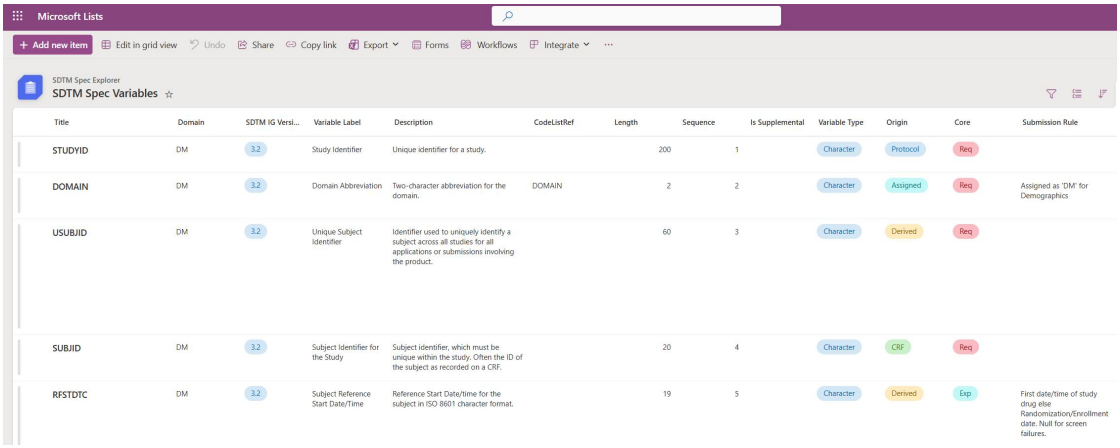

Figure 2. SDTM Standard Spec Sharepoint List

The key design decision is that **Domain** and **SDTM IG Version** are columns rather than separate worksheets or lists. This allows a single query to retrieve, for example, all variables across all domains for IG 3.4, or all changes to the AE domain from IG 3.3 to 3.4 — operations that require manual cross-referencing in Excel.

SDTM_DOMAINS — DOMAIN-LEVEL METADATA

This list captures domain-level attributes that apply uniformly across all variables in a domain, such as domain class (Findings, Events, Interventions, Special Purpose), structure (e.g., "One record per subject"), and parent–child relationships (e.g., SUPPAE → AE).

STUDY_SPEC_IMPLEMENTATION — STUDY-SPECIFIC RECORDS

This list captures study-specific changes from the standard specification. It follows a **delta-only** pattern: variables that do not appear in the list are assumed to follow the applicable standard specification. This

reduces duplicate records while allowing differences from the standard to be reviewed and compared across studies.

Key columns include:

- **StudyID** — the study identifier (e.g., B1234567)
- **StandardVariable** — Lookup linking back to the corresponding row in SDTM_Standard_Spec
- **ImplementationType** — choice: Modified / New / Dropped.
- **StudyProgrammerRule** — the study-specific mapping rule (when deviating)
- **Reason** — required justification text
- **Status** — Draft / Submitted / Under Review / Approved / Returned

For modified or dropped standard variables, the combination of StudyID, IG version, domain, and standard-variable reference identifies the applicable override. New variables are stored as study-specific records without replacing the corresponding standard record. In a production implementation, uniqueness checks and workflow controls would be needed to prevent conflicting active changes for the same study, version, domain, and variable.

SDTM_CHANGELOG — AUTOMATED AUDIT TRAIL

The SDTM_ChangeLog list is populated by Power Automate when monitored specification records are modified. Each log entry may capture the affected item, the user, the modification time, the changed field, and the relevant before-and-after values when those values are available to the workflow.

The list provides a centralized and queryable history of specification changes. It should not, by itself, be considered a validated or tamper-proof audit trail. The strength of the control depends on the configured permissions, version-history settings, retention policies, flow ownership, monitoring, and the governance of direct access to the underlying lists.

CONDITIONAL FORMATTING

SharePoint Lists support JSON-based column formatting to provide visual cues directly in the list view. For example, the **Core** column can be rendered with color-coded badges:

```
{
 "$schema": " https://developer.microsoft.com/json-schemas/sp/v2/column-formatting.schema.json",
 "elmType": "div",
 "txtContent": "@currentField",
 "style": {
 "padding": "2px 10px",
 "border-radius": "12px",
 "color": "white",
 "font-weight": "600"
 },
 "attributes": {
 "class": "=if(@currentField == 'Req', 'sp-css-backgroundColor-errorBackground', if(@currentField == 'Exp', 'sp-css-backgroundColor-warningBackground', 'sp-css-backgroundColor-successBackground'))"
 }
}
```

This produces red badges for **Required**, amber for **Expected**, and green for **Permissible** variables, providing immediate visual distinction that would require manual cell-by-cell formatting in Excel.

INTERFACE LAYER: POWER APPS

The Power Apps front end consists of four primary screens designed around two user personas: **Programmers** and **Reviewers**. The screens are accessible through a unified navigation bar, with state passed between screens via Power Fx context variables.

STANDARD SPEC BROWSER

The landing screen presents all SDTM domains as a visual gallery (Figure 3). Each card shows the domain code, name, class, number of variables (including supplemental qualifiers), and the last update timestamp. A segmented control at the top allows the user to switch IG versions (3.2 / 3.3 / 3.4), which dynamically updates the displayed counts.

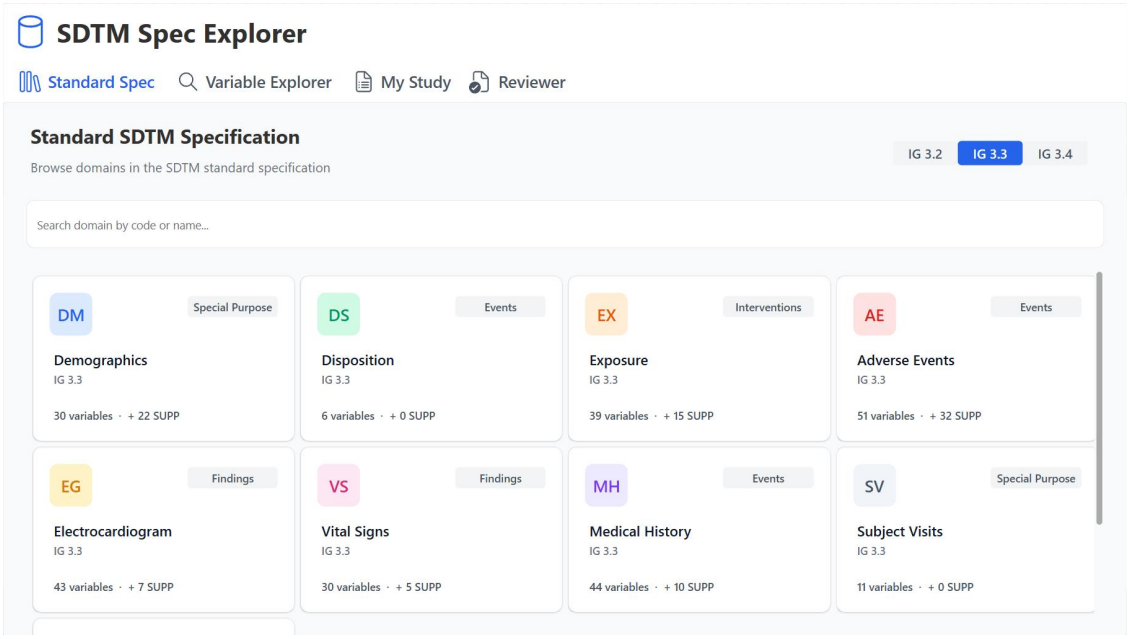

Figure 3. Standard Spec Browser

The card-based gallery replaces the experience of navigating dozens of Excel tabs. Selecting a card navigates the user to the Domain Detail screen — a SharePoint List-style table showing all variables in that domain. Each row shows the variable name, label, Core status (with color-coded badge), Origin, Type, Length, and an indicator of how many studies have deviated from the standard. Quick-filter chips

above the table enable rapid filtering by Core, Origin, or supplemental status.

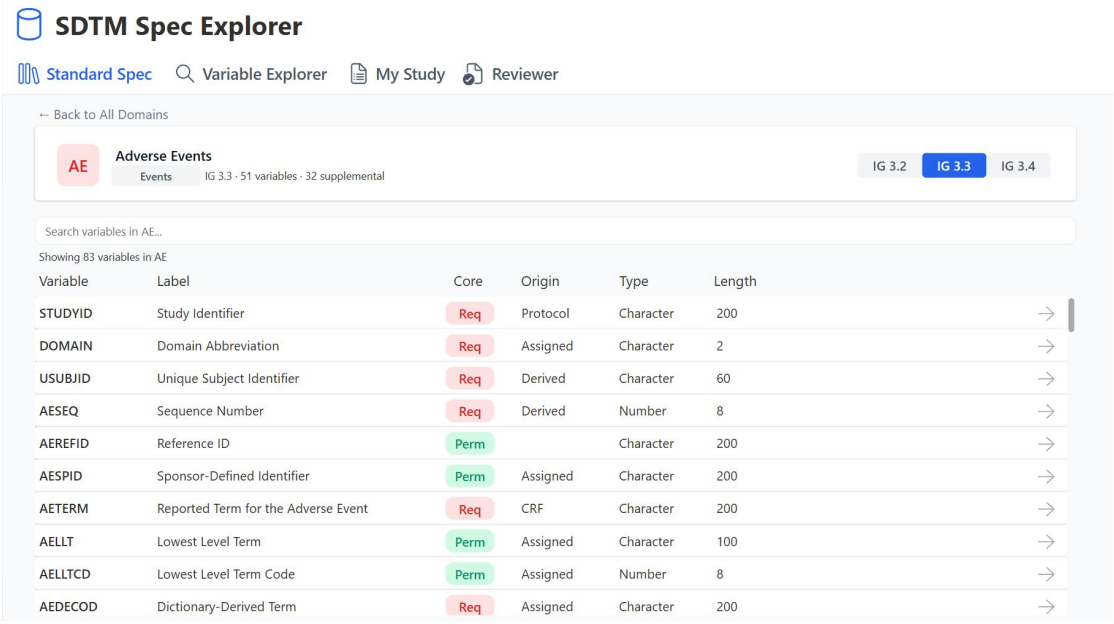

Figure 4. Domain Details

VARIABLE EXPLORER

While the Standard Spec Browser supports top-down navigation (Domain → Variable), the Variable Explorer supports **bottom-up search-driven discovery** (Figure 5). Users can search for a variable by name, label, description, or programmer rule content, and apply multi-dimensional filters (Domain, IG Version, Core, Origin) through chip controls.

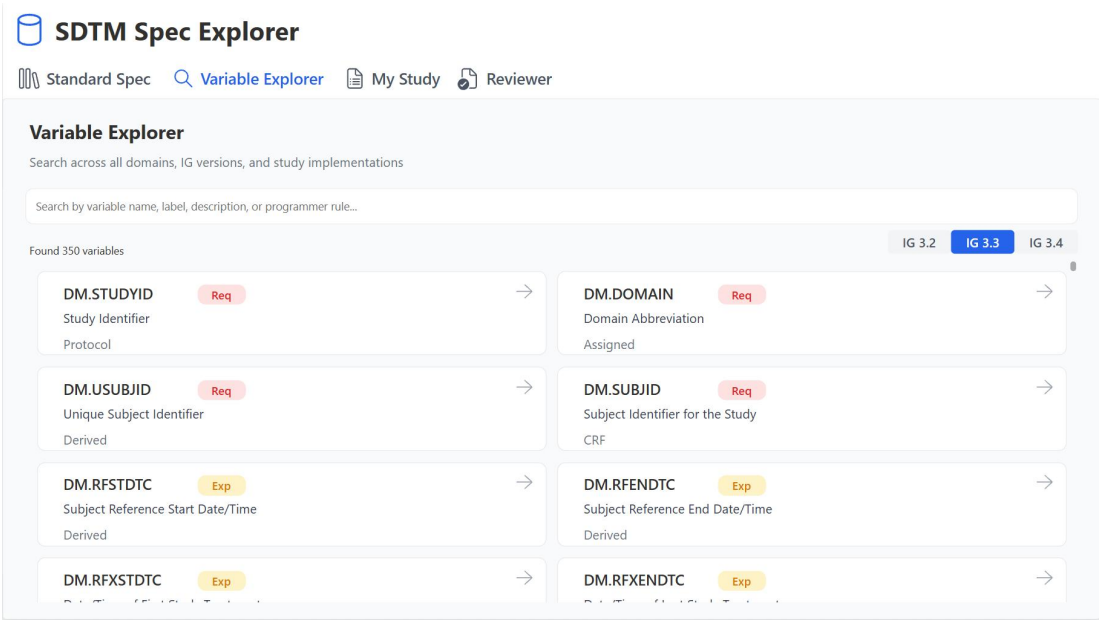

Figure 5. Variable Explorer

Results are presented as cards in a 2-column grid, each summarizing the variable's domain, label, Core status, Origin, and cross-study usage. Selecting a card opens the Variable Deep View — a detailed page combining the standard specification, all study-level implementations, related programmer-rule references, and recent change information in a single page. (Figure 6).

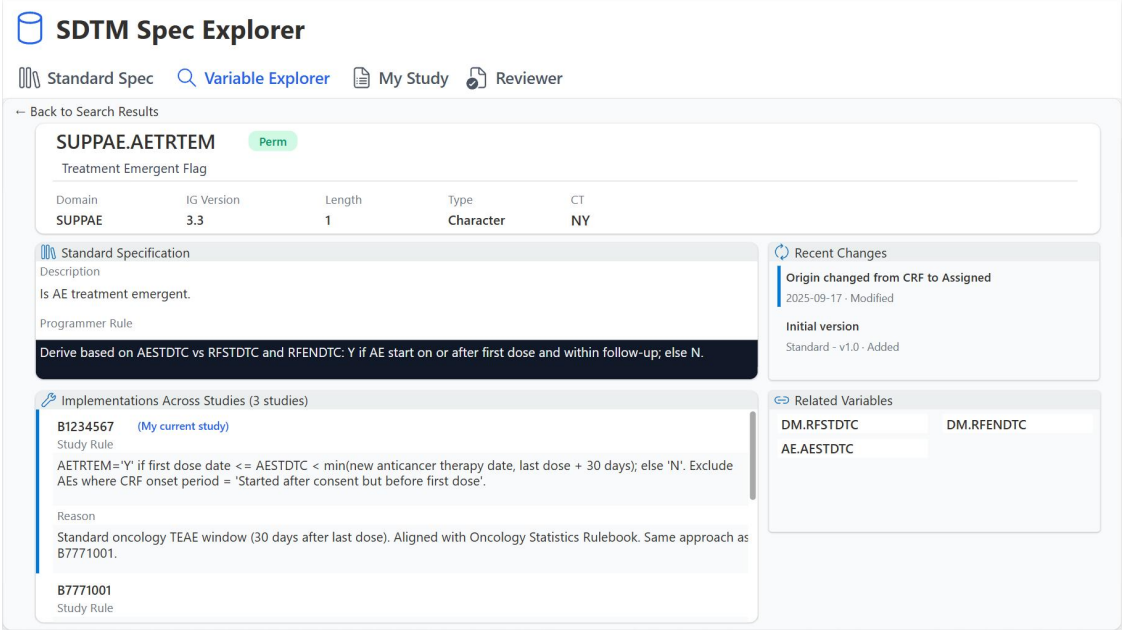

Figure 6. Variable Deep View

The Variable Deep View is the framework's most distinctive feature. In a single view, a user can see:

- The **standard** programmer rule and metadata
- Available **study-level implementations**, including the implementation type, study-specific rule, justification
- Variables whose names are referenced in related programmer-rule text, with navigation to the corresponding variable records
- Available version and change-history information for the selected variable

This consolidates information that would otherwise require opening multiple Excel files (one per study) and manually correlating data.

MY STUDY WORKSPACE

The My Study Workspace is the primary interface for recording study-specific implementation changes. Consistent with the delta-only data model, programmers record only variables whose implementation differs from the applicable standard, rather than explicitly processing every standard variable.

To submit a study-specific change, the programmer selects the study, IG version, domain, and variable; specifies the implementation type; and provides a study-specific programmer rule and justification as applicable. Submitting the form creates a record in the Study_Spec_Implementation list and may initiate the configured downstream workflow.

The workspace displays the study-specific changes recorded for the selected study, grouped by domain and annotated with their workflow status. The prototype also demonstrates an export pattern that combines the applicable standard records with approved study-specific changes to produce a consolidated tabular specification for a selected study and domain.

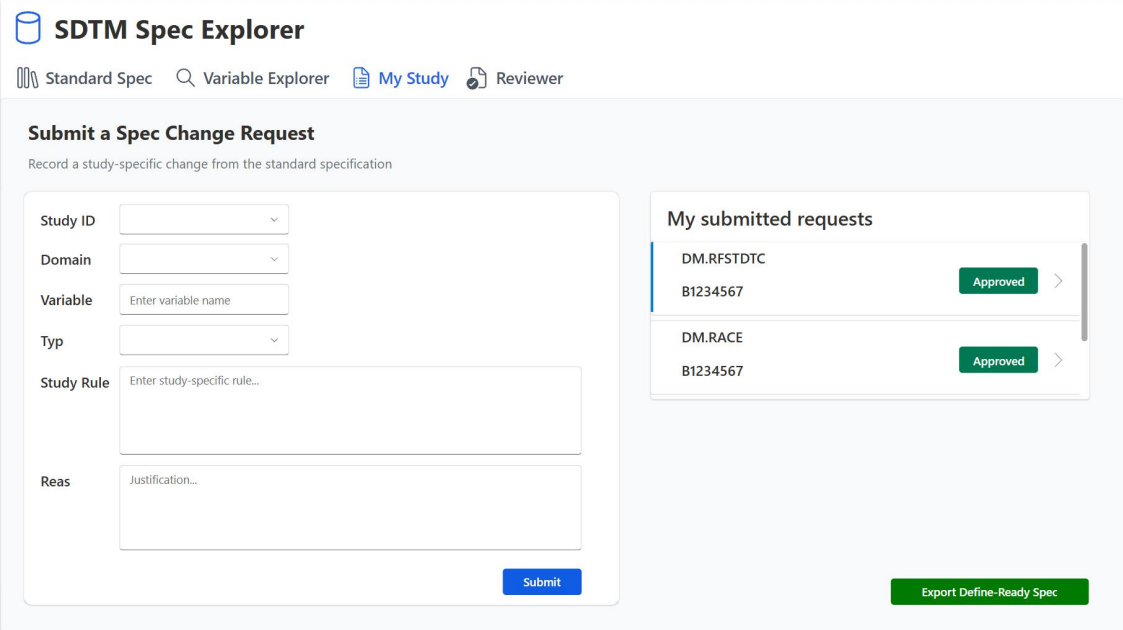

Figure 7. My Study Workspace

AUTOMATION LAYER: POWER AUTOMATE

The automation layer provides the workflow capabilities that Excel-based processes lack. Three core flows underpin the framework, with additional flows easily added as needed.

NOTIFICATION FLOW

Trigger: When an item is created or modified in the SDTM_Standard_Spec list.

Logic:

1. Capture the modified item and identify which fields changed.
2. Identify the list of users who have referenced this variable in their study implementations.
3. Send notifications via email and Microsoft Teams to those users.

This ensures that any change to the standard specification is immediately surfaced to programmers whose studies might be affected — without the need for manual email cascades.

APPROVAL WORKFLOW

Trigger: A study-specific change is submitted for review.

Logic:

1. Retrieve the submitted records for the applicable study and domain.
2. Create a Microsoft 365 Approval task assigned to the designated reviewer.
3. Notify the reviewer through Teams or Outlook.
4. On approval, mark all records as Approved.
5. On rejection (Return for Revision), notify the programmer with comments.

This formalizes the previously informal email-based review process and ensures all reviews are traceable.

DEFINE-READY EXCEL WORKFLOW

Trigger: When a study programmer clicks export button in Power Apps

Logic:

1. Identify Study ID and Domain
2. Extract items from both standard library and study specific implementation
3. Merge two lists and generate define-ready CSV table.
4. Create file and save to SharePoint site or OneDrive

EXTENSIBILITY

Additional flows can be added incrementally without disrupting existing functionality. Examples include:

- **Daily standard sync reminder** — alerting programmers when the standard has been updated since their last study activity
- **Quarterly review summary** — aggregating spec change patterns across studies for governance review
- **Integration with study tracking systems** — automatic status updates pushed to external dashboards

ADDITIONAL CAPABILITIES

Beyond the core functionality, the framework supports several extensions that further demonstrate the advantages of a platform-based approach.

CROSS-STUDY PATTERN DETECTION

When multiple studies make similar spec changes from a standard variable, this signals a potential gap in the standard itself. The platform can periodically aggregate spec changes across studies and flag patterns (e.g., "4 studies have demoted AE.AETRTEM") for governance review. In Excel-based workflows, this analysis requires essentially significant manual effort.

EXTENSION TO ADAM AND OTHER SPECIFICATIONS

The same architectural pattern can be applied to ADaM analysis dataset specifications, controlled terminology, study tracking sheets, and other documents currently maintained in Excel. The SharePoint List schema and Power Apps interface can be replicated or extended with minimal incremental effort.

INTEGRATION WITH OTHER MICROSOFT 365 COMPONENTS

Because all data lives in SharePoint, it is natively accessible to other Microsoft 365 tools — including Power BI for dashboarding, Microsoft Lists mobile app for on-the-go access, and Microsoft Graph for custom integrations. This creates compound value beyond the SDTM Specification Explorer itself.

SHAREPOINT AS A LANGUAGE-AGNOSTIC DATA BACKBONE

While Power Apps provides the primary user interface for this framework, the underlying SharePoint data layer is not limited to Power Apps consumption. The centrally maintained specification data can be accessed by a variety of tools and workflows, making the framework valuable even for teams whose members prefer other technologies.

Three complementary access patterns are supported:

- **Direct API access.** SharePoint Lists are exposed through the Microsoft Graph API and the SharePoint REST API, allowing R (e.g., via httr), Python (e.g., via requests or the Office365-REST-Python-Client library), or SAS to query the data programmatically in real time.
- **Scheduled export to a shared drive.** A recurring Power Automate flow can export the SharePoint Lists to CSV or Excel files on a network drive at a defined cadence (e.g., nightly).

Downstream tools such as R Shiny apps or Python scripts can then read these files without needing any Microsoft 365 credentials or API configuration — often the simplest and most robust option in practice.

- **Structured operational repository**. Even without a custom application, SharePoint Lists can provide typed columns, permissions, filtering, and version history for lightweight operational metadata management.

This decoupling of the data governance layer from the consumption layer means each user can work in their preferred environment — Power Apps, R Shiny, Python, SAS, or Power BI — while still drawing from a single, consistently maintained source of truth.

DISCUSSION

IMPLEMENTATION FEASIBILITY

The prototype was implemented using standard Power Apps controls and Power Fx functions. The demonstrated interface includes domain galleries, a table-style variable browser, search-driven variable exploration, IG-version filtering, comparison views, and a form for recording study-specific changes.

Implementation requires careful design around Power Apps delegation behavior. Non-delegable operations may be evaluated only against the configured client-side row limit, which can produce incomplete results when the underlying list is larger. In the prototype, filtering by IG version and domain reduces the active data scope; however, a production design should verify the delegation behavior of each formula rather than relying only on the expected number of displayed records. Delegable filters, indexed SharePoint columns, pre-aggregated fields, or controlled local collections may be used according to the query pattern and data volume.

Visual mappings such as Core-status colors are maintained explicitly in Power Fx because SharePoint column-formatting settings are not automatically inherited by Power Apps controls. Aggregations across studies are better handled through scheduled flows, precomputed summary records, or a separate analytical layer rather than repeated client-side queries.

These observations demonstrate that the proposed interface is technically feasible, but they do not constitute a formal scalability or performance evaluation.

SCALABILITY CONSIDERATIONS

While SharePoint Lists theoretically support up to 30 million items, performance considerations become relevant beyond the 5,000-item view threshold. For typical SDTM specification management — even covering all domains across multiple IG versions within a single study — the data volume (~2,000–3,000 rows) remains well within optimal performance range. For organizations managing specifications across many studies, the framework can be extended through indexed columns, filtered views, or by partitioning specifications across multiple lists by study or IG version.

LIMITATIONS

The framework relies on the Microsoft 365 ecosystem and assumes organizational availability of SharePoint, Power Apps, and Power Automate. Teams without access to this platform — or organizations with strict governance policies that restrict citizen development — may not be able to adopt the framework directly. The learning curve for Power Fx and SharePoint List configuration, while moderate, is non-trivial; some upfront investment in training or template development is required.

The framework does not currently include native support for offline editing, mobile-optimized data entry, or programmatic API access for SAS or R consumers. These can be added through complementary tools (e.g., Power Apps mobile, Microsoft Graph API) but require additional implementation effort.

CHOOSING THE RIGHT TOOL: LOW-CODE VS. CODE

A natural question is why this framework should be implemented on a low-code platform rather than in a more flexible environment such as R Shiny, Python, or a custom web application. This question has

become increasingly relevant as AI-assisted development reduces some of the initial effort required to create application interfaces and code.

For browse-only applications, many technology stacks can provide effective interfaces for specification data. The architectural distinction becomes more important when an application must support authenticated multi-user updates, persistent workflow states, role-based access, review routing, notifications, and centrally managed data.

A custom solution can support these capabilities, but it may require additional infrastructure such as an application service, persistent data store, authentication and authorization integration, workflow logic, deployment processes, monitoring, and technical ownership. The required effort depends heavily on the organization's existing platforms and engineering support.

The Microsoft 365 approach provides several of these services within an ecosystem that may already be governed and available within an organization. SharePoint provides structured storage, permissions, and version history; Power Apps provides the workflow-oriented interface; and Power Automate provides event-driven integration and approval patterns. This reduces the need for a separately managed application server, but it does not remove the need for application design, testing, maintenance, governance, or licensing.

The choice is therefore not simply between code and no-code. It depends on the primary requirements of the application. A custom development framework may be preferred when computational flexibility, custom visualization, performance, or software-engineering control is the dominant requirement. A managed low-code platform may be attractive when integration with enterprise identity, permissions, collaboration tools, and workflow services is more important.

RELATIONSHIP TO AI-ASSISTED APPROACHES

AI-assisted development and generative AI can accelerate parts of the application lifecycle, including the creation of initial mappings, draft specifications, interface components, formulas, and workflow logic. However, content generation alone does not provide the governance mechanisms required for controlled multi-user maintenance, review, approval, access management, and traceability.

The architecture described in this paper addresses a complementary problem. By storing standard and study-specific specification metadata as structured records, the framework provides a governed data foundation on which AI-assisted capabilities could later be evaluated.

Potential extensions include summarizing differences between standard and study-specific programmer rules, retrieving similar implementation patterns from other studies, identifying records that may merit governance review, or drafting reviewer-facing comparisons. Such outputs would remain subject to human review and to organizational requirements for privacy, security, validation, and appropriate use.

AI integration was not implemented or evaluated as part of the prototype. It is presented as a future extension rather than as a current capability of the SDTM Specification Explorer.

CONCLUSION

This paper presented a framework for modernizing SDTM specification management through an integrated Microsoft 365 architecture based on SharePoint Lists, Power Apps, and Power Automate. The central design principle is to separate reusable standard metadata from study-specific implementation changes and to manage those records through a structured interface and workflow layer.

A functional prototype demonstrated domain and variable browsing, IG-version filtering, search across specification metadata, comparison of study-specific implementations, and submission of study-specific changes. The architecture also illustrates workflow patterns for notification, approval, centralized change logging, and consolidated specification export.

The work establishes technical feasibility but does not provide a formal comparison with Excel-based processes or evidence of production-scale performance, regulatory validation, or enterprise adoption. These areas require further evaluation using representative studies, users, data volumes, and organizational controls.

The same architectural pattern may be adaptable to ADaM specifications, controlled terminology, study-tracking metadata, and other structured documents currently maintained in spreadsheets. Rather than replacing every spreadsheet or custom application, the framework provides one practical option for teams whose primary requirements involve centralized metadata, multi-user workflow, enterprise integration, and governed change management.

REFERENCES

Clinical Data Interchange Standards Consortium. (2021). *Study data tabulation model implementation guide: Human clinical trials* (Version 3.4). https://www.cdisc.org/standards/foundational/sdtmig

Microsoft Corporation. (2026a). *Format a column to change how it looks (SharePoint list formatting)*. Microsoft Learn. https://learn.microsoft.com/en-us/sharepoint/dev/declarative-customization/column-formatting

Microsoft Corporation. (2026b). *Microsoft lists and SharePoint lists documentation*. Microsoft Learn. https://learn.microsoft.com/en-us/sharepoint/lists-overview

Microsoft Corporation. (2026c). *Power Apps documentation*. Microsoft Learn. https://learn.microsoft.com/en-us/power-apps/

Microsoft Corporation. (2026d). *Power Automate documentation*. Microsoft Learn. https://learn.microsoft.com/en-us/power-automate/

Microsoft Corporation. (2026e). *Understand delegation in a canvas app*. Microsoft Learn. https://learn.microsoft.com/en-us/power-apps/maker/canvas-apps/delegation-overview

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Lou, Shuting
Pfizer (China) Research and Development Co., Ltd.
shuting.lou@pfizer.com