Paper PP06

Small Application for Big Change: Data Integrity and Compliance through Automated Decision Tracking with Shiny (R)

Olena Olishevska, EstiMates, Espoo, Finland
Sami Virtanen, Orion, Turku, Finland
Mika Makinen, EstiMates, Rauma, Finland

ABSTRACT

Ensuring data integrity and compliance in clinical and nonclinical studies is challenging due to the diverse systems and processes employed by companies. To address this, we introduce a prototype of web application designed to track derivation and formatting decisions, thereby enhancing transparency and accountability in data flow.

Proposed solution accounts for harmonizing input data, logging of user actions and versioning of standardization documents, providing a full traceability of data flow. Using real-world example, we show how our implementation of the proposed design using Shiny (R framework) creates a decision database in accordance with CDISC Controlled Terminology, which is later used for automated production of SEND-compliant domains. Early user evaluation indicates major reduction in data processing time and improved transparency of processing steps for further automation. Join us to explore how this innovative tool can revolutionize the handling and oversight of data in nonclinical studies, ensuring both integrity and compliance.

INTRODUCTION

With the wish to ease comparison of study data across a couple dozens of pre-clinical studies came the need to structure, harmonize and standardize the data that these studies hold. We had to create a mechanism to explore the raw data files and keep track of all transformation steps (that also require consensus within teams and decision made) and different cases of variable/value use across many projects. To compile all these aspects of standardization in a well-organized manner, we created a **decision database (DDB)** that stores processing rules. In this paper, we refer to "database" meaning any structured and queryable data storage without implying any particular technology.

For the purposes of collecting data processing rules and applying them to raw data in real-time to create result domains, we develop a web application using Shiny (R) [1]. Our goal is to harmonize and transform study-related data files into SEND domains with ability to map raw variables to SEND domain variables, format raw values to Controlled Terminology, and export final SEND files in a desired destination. The developed application implements this pipeline while interacting with the decision database through all the steps **(Fig. 1)**. We also perform extensive logging of user activity and versioning of the state of the database, providing a clear trail of transformation of raw data into its final form – standardized domains. While currently used for pre-clinical studies only, such approach can be extended and applied to clinical studies for maintaining structured, transparent and traceable data transformation pipelines.

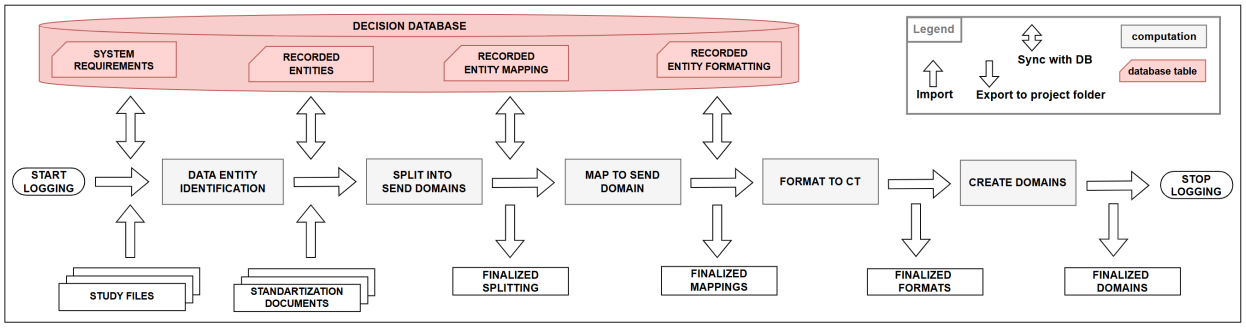
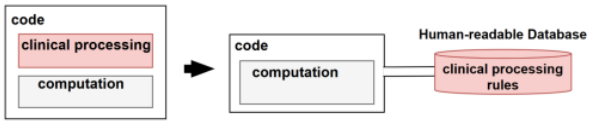


Figure 1. Integration of decision database into the data processing pipeline

The paper is organized in the following manner. In the first section, we outline the key concepts of our approach, such as decisions and decision database, and empathize the importance of user interaction. In the second section, we demonstrate the implementation of the suggested approach for nonclinical studies; each subsection draws a comparison between a specific aspect of the approach and its implementation in the developed application. In the third section, we discuss technical aspects of the solution, such as UI design, modularity, logging, and exporting.

APPROACH: UI-DRIVEN DECISION DATABASE

We treat each processing rule as a distinct **decision** and store all decisions in a centralized **decision database (DDB)**. Since the processing rules only relate to the clinical requirements, by storing them in a centralized database we separate the *clinical* aspect of processing from *technical*, making the data transformation process more transparent and enhancing reusability of the code:



Within the database, we collect information about raw data file types, key variables for each file type, their mapping and formatting rules to SEND domain. This way, we can keep track of any structural changes in files, having versioned history of changes in any aspect of processing.

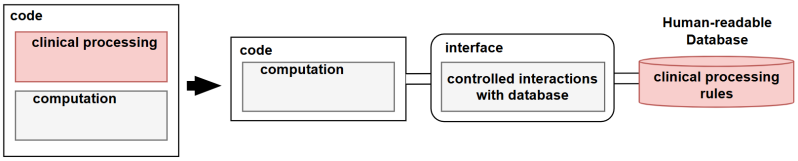
When new raw data files are introduced, the collected rules from the database are used to identify new structures and values or automatically applied to 'recognized' structures.

BENEFITS

In addition to making the data processing code cleaner, the use of a decision database makes it easier to maintain consistent processing strategies across teams and evaluate data readiness for automatic SDTM/SEND domain creation. It also allows decision-makers to evaluate the quality of data collection approaches, provide feedback upstream, and improve the practices accordingly.

FILLING THE DATABASE: UI OR NOT UI?

Although it is possible to populate the database manually at the initial stage, the effort required for it quickly skyrockets if the decisions are directed by different people within different teams and relate to several data files within each project. Building a custom interface for interactive entry provides more structured and guided input, extensive validation and automation possibilities, and creates a controlled environment where user actions can be logged and/or limited by access control:



Ideally, with strict data collection rules, limited free-text use, consistent and unchanging data structures, the database can be filled with a 'full and final' list of rules once and then be used for fully scheduled automated script runs without extra human interactions. But in reality, the decision database should be something that evolves together with the progression of standards, systems, and practices. With constant need to add/update/remove processing rules, the solution should provide a clear and user-friendly interface for interacting with the database.

FURTHER USE: SEND DOMAIN CREATION

Our goal is not only to provide a User Interface for gathering processing rules and uploading them to decision database, but also to see the practical application of those decisions in use – creation of standard-compliant datasets. For that, the decision database should be designed so that the decisions line up in a consistent chain of step-by-step transformations. In the following section, we provide an example of implementing such design for SEND domains.

APPLIED USE: AGGREGATION OF DECISIONS FOR NONCLINICAL STUDIES

KEY CONCEPTS: DATA ENTITIES AND DECISIONS

Attribution of raw data records to SEND domains (automatic or manual) requires identification of some key variables, the values of which distinguish one group of records from another, e.g. body weight measurements from organ weight measurements. We will further refer to such logical groups of records as ***data entities***.

Data entity is defined by a combination of values in the identification variables and describes unique measurements, procedures, findings, events etc. Data entity identifiers may vary depending on file type and system while being related to the same domain **(Fig. 2)**. Having these identifiers within the decision database allows the user to unify data from different files and systems.

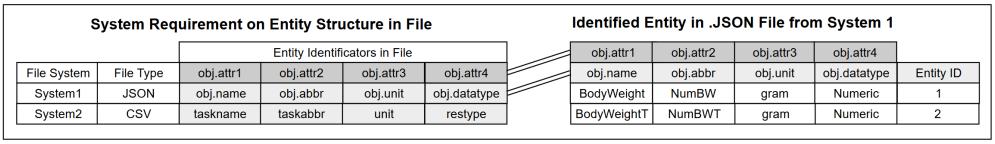


Figure 2. Example of data entity identifiers depending on their source system.

After entities are identified in the given file(s), processing decisions can be made for each separate entity. Since data entities have some correspondence to SEND domain records, entity-related decisions can be classified as *cross-domain* and *in-domain*.

Entity-wise *(cross-domain)* ***decisions*** include rules on harmonizing data from different systems. Simplest example of entity-wise decision can be **domain attribution**: which domain should the given data entity be mapped to.

Within-entity *(in-domain)* ***decisions*** include rules on how variables and values within entity should be transformed to fit SDTM and SEND standards. Main *within-entity decisions* we focus on are following:

- **mapping** to SEND domain metadata: what variables in raw data correspond to what SEND domain variables and how those should be derived;
- **formatting** to Controlled Terminology or custom format: what are the standardized definitions that correspond to each raw value.

The following example illustrates how three types of processing decisions (attribution to domain, mapping rules, and formatting rules) are collected and then saved to the decision database **(Fig. 3)**.

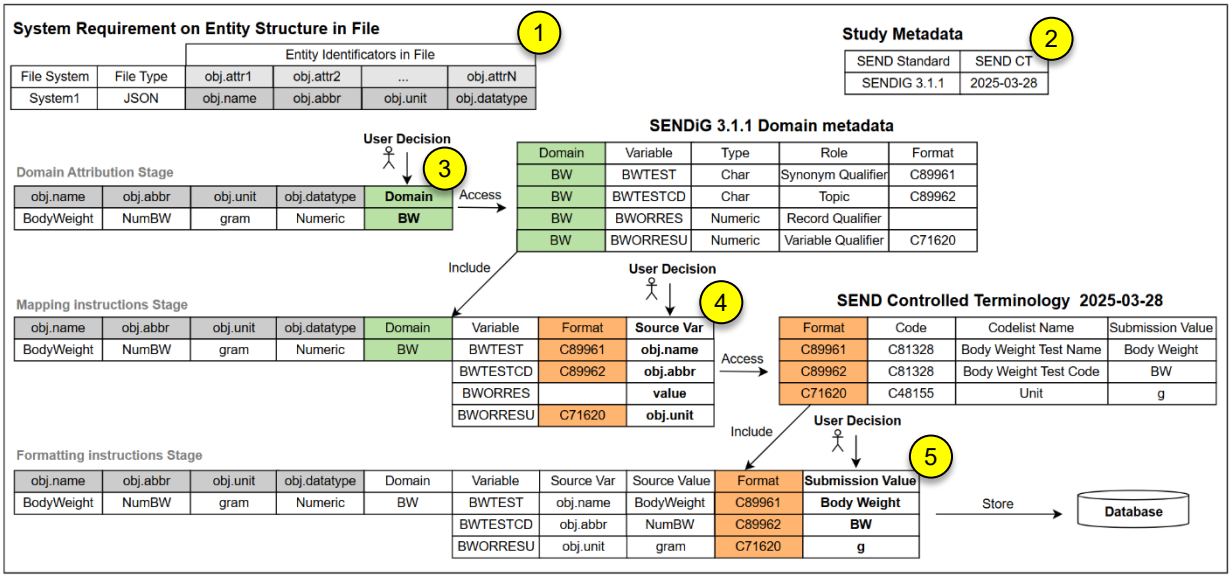


Figure 3. The creation of processing decisions for a new entity with step-by-step user input. Before any data is imported, the requirements for the file structure and the standards relevant to the study must be defined (1,2). When the raw data is imported, it is attributed to a specific domain (3). The domain metadata postulates specific variables which must be defined in the data; on the following step, the variables from raw data are matched to SEND variables (4). The standard also defines specific formats for its variables, and the formats of the raw data are matched in the last step (5).

The order of decisions (domain attribution → mapping → formatting) should be consistent because there are many restrictions and dependencies involved when applying those decisions to transform data. For example, *cross-domain decisions* should be populated before *in-domain* decisions, because in-domain decisions rely on domain metadata that the entity is attributed to.

FILLING THE DATABASE: GUIDING USERS

By populating the database through the developed application, we collect the decisions in strict order, accommodating necessary restrictions **(Fig. 4)**. We also provide a friendly UI for informing users about any inconsistency in the input they provided.

Within the application, the decisions are further classified based on project attribution. Processing decisions that are applicable regardless of study type are considered **global**, while decisions that are unique to a particular study are considered **project level.**

After selecting the data files, imported raw data is first compared to existing information in the database *(global rules)*, and the relevant rules are automatically applied. If there is no information on how to process a given data structure, users can see 'unrecognized' structures in a separate table in UI (**Fig. 5**, left). With the provided UI, they can fill the missing information and complete a decision by confirming and saving it as a *project-level rule* (**Fig. 5**, right). After all the required rules are defined, users will be able to see the resulting dataset, which is presented in more detail in the next subsection.

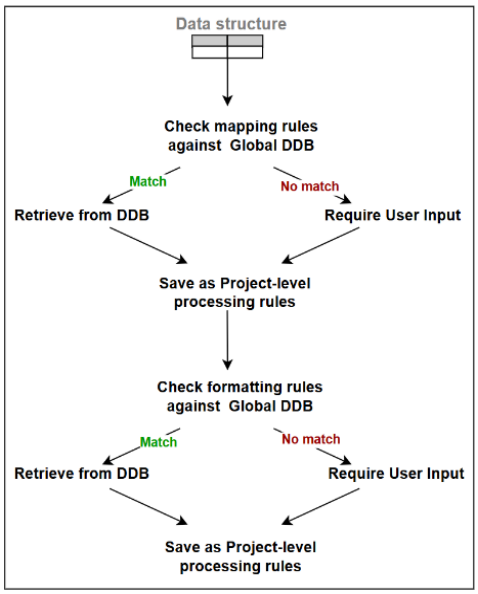


Figure 4. Order of decision collection.

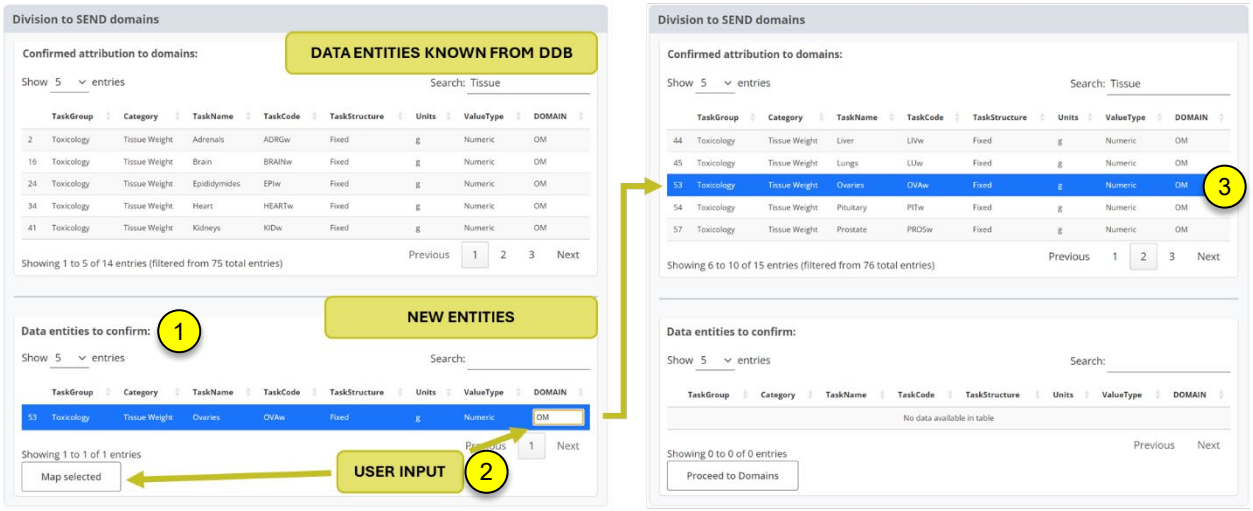


Figure 5. Domain Attribution of a new entity. When the entity is first introduced, it appears in "To confirm" table, separate from the confirmed mappings (1). After the user assigns a domain and confirms the mapping (2), the DDB is updated and the new rule shows up in "Confirmed" (3).

After the entity is attributed to a corresponding SEND domain, the domain's metadata is pulled to later provide a link between entity variables and SEND domain variables. To optimize derivation, entities are grouped according to a domain they have been mapped to **(Fig. 6)**. Transformation of raw data to domain happens in stages similarly to manual processing: starting with direct/formatted copies of raw variables and continuing with several stages of derivations. It is important to note that beyond the simple example **(Fig. 6)** entity derivation instructions can be expanded with more configurations for advanced derivations.

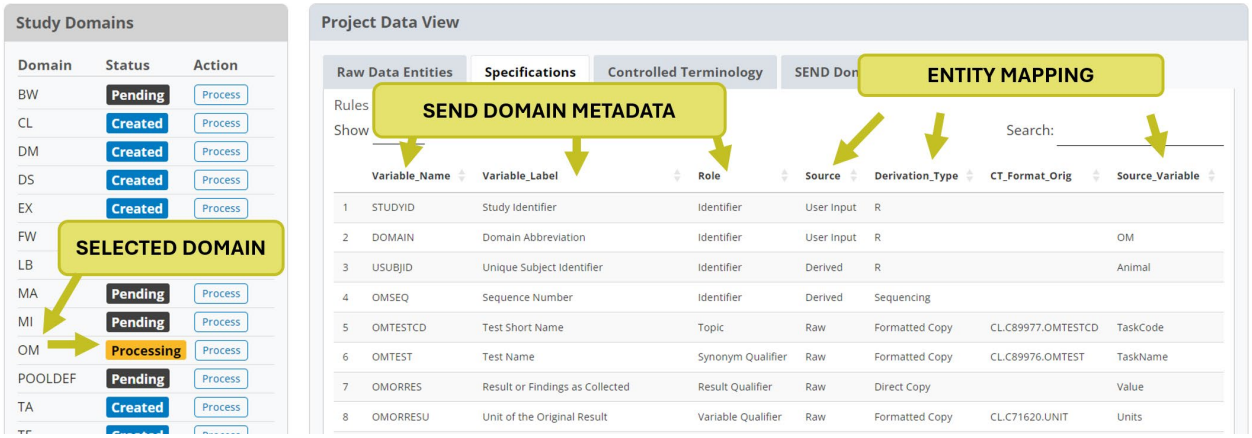


Figure 6. Example of selected group of data entities mapping to SEND OM domain. SEND variables (Variable_Name) are derived from raw data variables (Source_Variable) using a set of derivation instructions (Source, Derivation_Type).

When the mapping is confirmed, unique values of Source Variables can be formatted to SEND Controlled Terminology or any custom format list **(Fig. 7)**.

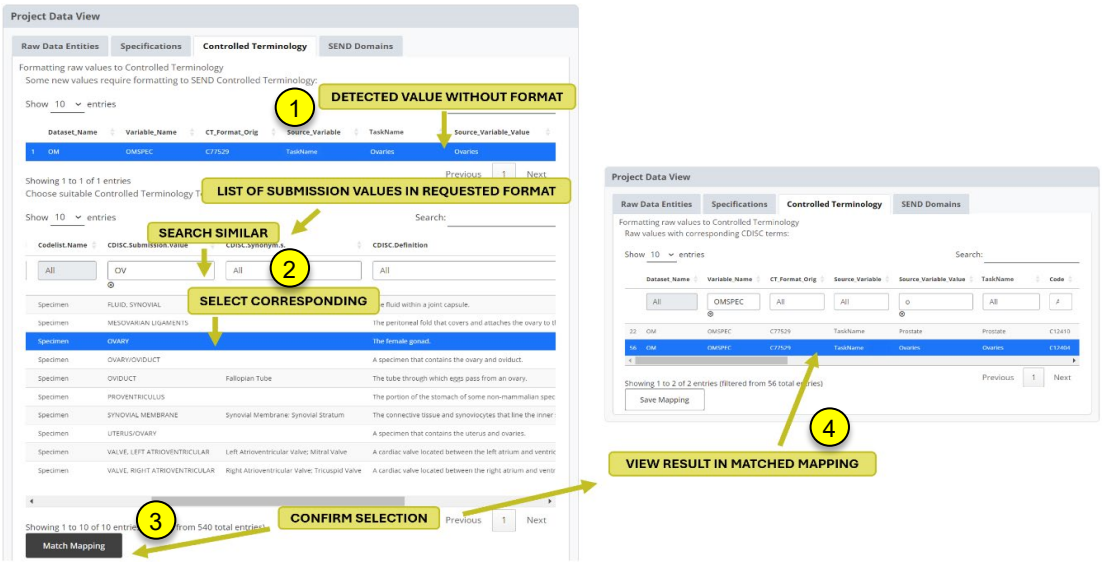


Figure 7. Example of formatting a raw value to SEND Controlled Terminology. A new value in TaskName that requires mapping to Specimen codelist in Controlled Terminology appears in a separate table (1). Upon selecting the value from the table, Specimen codelist is displayed, prompting the user to find a corresponding term (2). After the user confirms chosen matching with button click (3), the new matched value can be found in the list of all formatted values (4).

After both specifications and formatting rules are completed, user can view generated domain in 'SEND Domain View' tab **(Fig. 8)**. This way processing rules can be applied in real-time and, if needed, can be amended, and result domains can be re-generated.

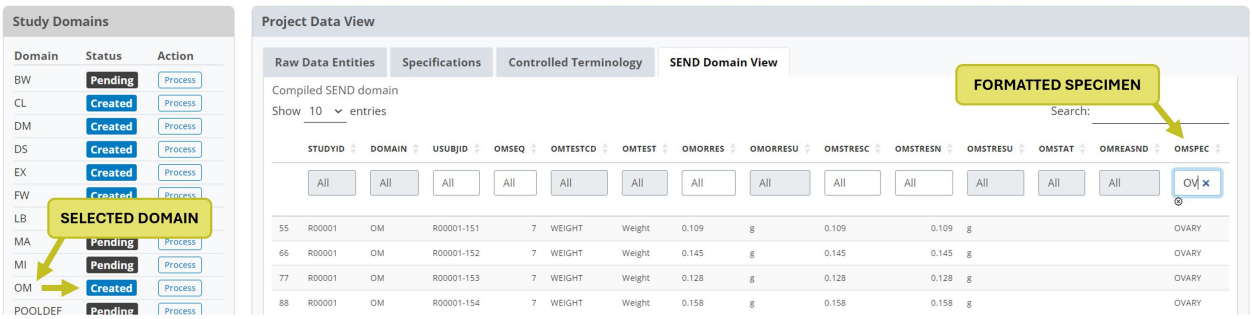


Figure 8. SEND domain created in real-time considering entiy mapping and formatting instructions. User-confirmed formatting of Specimen (see Fig. 7) is applied in result domian.

VIEWING RESULTS: SUMMARY PAGE

As the processing of each study may involve tens (if not hundreds) of decisions, we provide users with ability to cross-check and compare approaches across projects, teams and departments. Summary page below provides overview of all the documents, decisions, standards and custom rules that contributed to domain creation **(Fig. 9,10)**, as well as final domains themselves **(Fig. 11)**. Thus, the final datasets can be created, viewed and exported to desired location within one app.

We color-code processing rules to differentiate project-level decisions from global-level decisions that can be re-used in other projects. **Orange** indicates that these rules are project-level and are not yet confirmed for use in other studies. **Green** color indicates that these rules are confirmed for use in other studies. A project-level rule can be promoted to global or, if needed, removed from the database.

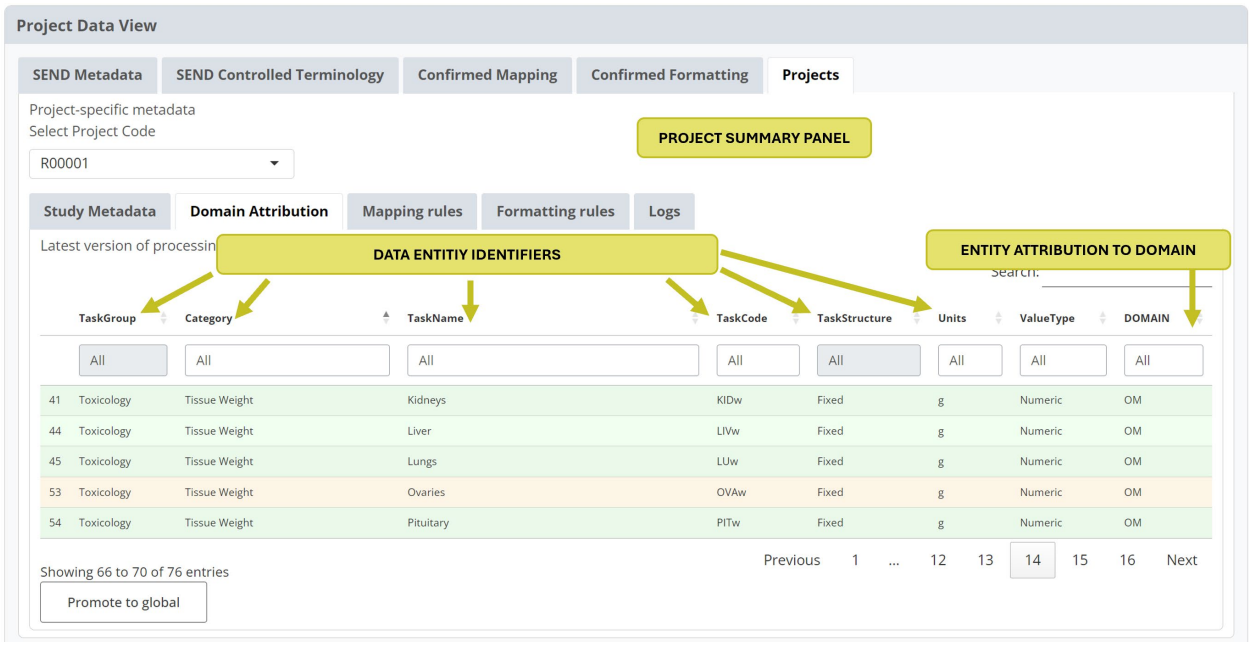


Figure 9. List of data attribution rules on Project Summary page.

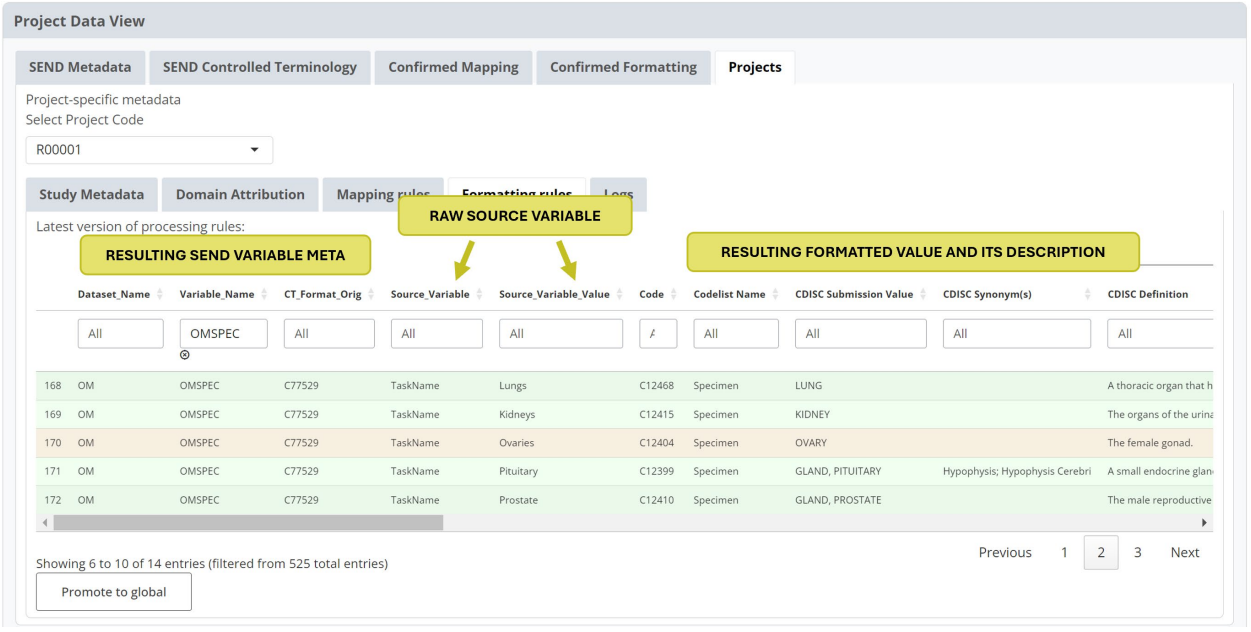


Figure 10. List of data formatting rules on Project Summary page.

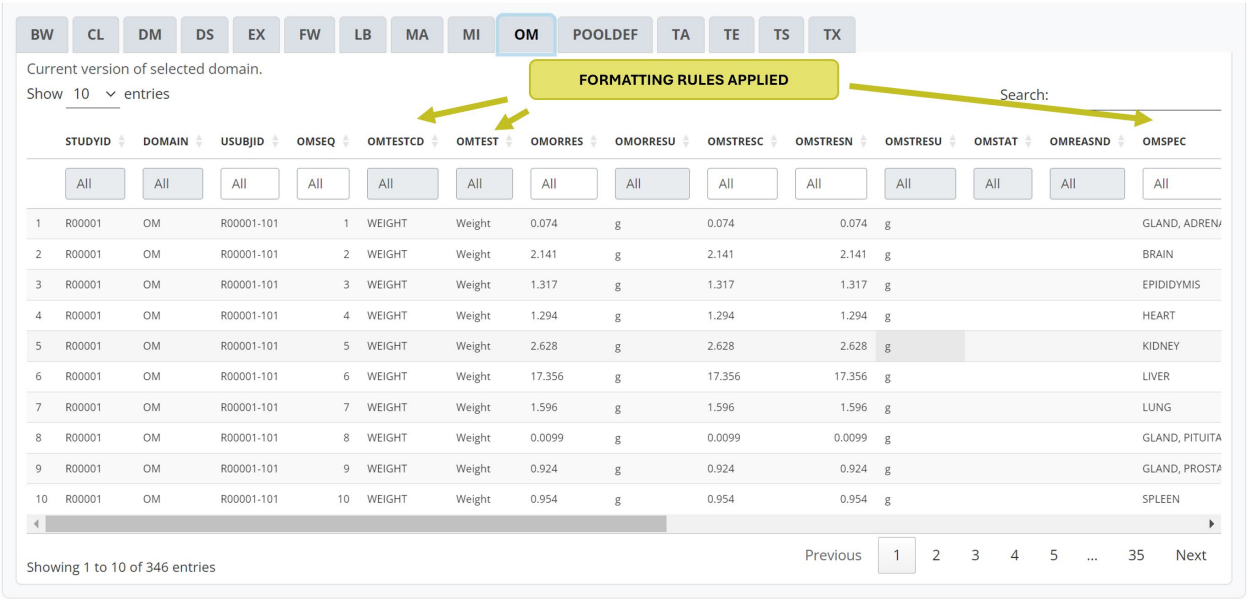


Figure 11. Application of previously confirmed mapping (see Fig. 8) and formatting (see Fig. 9) in the finalized OM domain.

COMPLIANCE REQUIREMENTS: LOGGING

When working with clinical and non-clinical data, full traceability of data transformation pipelines is usually required. The fact that the end goal is to create SEND datasets brings additional traceability and reliability requirements to the application. While for fully automated pipelines the code itself provides the necessary trace, in a UI-based solution it becomes important to record the order and content of user actions, versions of standards, and the state of the database at a time of domain creation. Therefore, we are required to provide audit-level logs of every user session. Our system is complex enough to require several types of logs involved:

- Database logs (to keep a history of decision database evolution over time: use of new systems and file formats, changes in structure of raw data, changes in regulatory or company practices)
- Study logs (to track any amendments to study: adding a new study file, finalizing a set of decisions, exporting a new domain)
- Domain logs (ordered list of decisions and transformations that led to domain creation)
- User Activity logs (page navigation, use and reuse of page elements, decision-making process)
- Technical logs (user activity, application performance, errors and crashes)

The implementation of logging in our developed application is discussed, among other aspects, in the following section.

CHALLENGES AND TECHNICAL ASPECTS OF BUILDING A RELIABLE SOLUTION WITH R AND SHINY

DESIGN CONSIDERATIONS

Designing an understandable UI is key. For that, we divide the application into pages by their scope **(Fig. 12)**. Each page then is divided into several blocks by their functionality. If there are blocks of similar functionality on several pages, it is more consistent to keep them in the same area among pages.

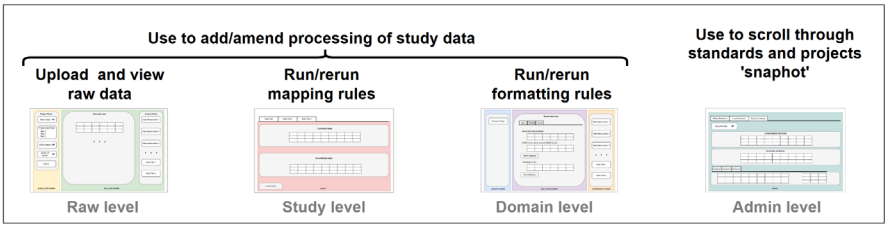


Figure 12. Simplified design schema.

In our case, there are blocks of Raw level and Domain level pages that define what data user sees in the middle section of the page. For Raw level page, it's a place for user inputs of project configuration and study data (**Fig. 13,** a), while for Domain level page it is a place to select a domain for further processing. Similarly, Raw level and Domain level pages both have a functional block for real-time interaction with data (raw or domain) on the right side of page. For the Raw level page, the intent of this block is to provide advanced search options through raw data (**Fig. 13,** c), while for Domain level page the block is intended for tuning the derivation parameters.

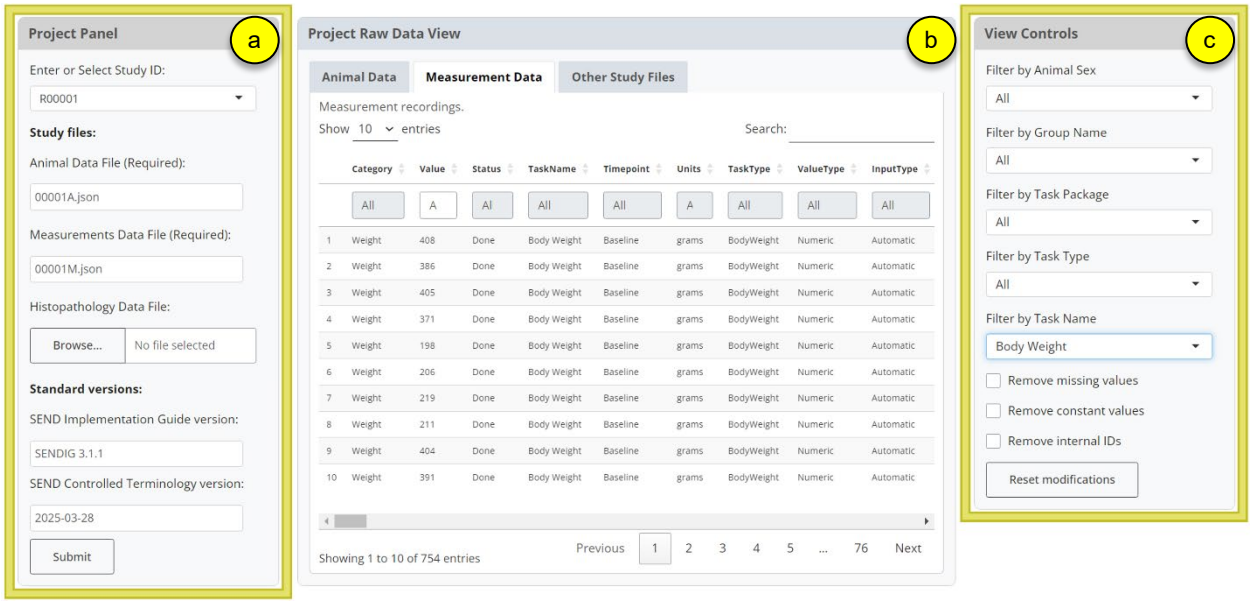


Figure 13. Raw data page. The page consists of project panel on the left (a), raw data view in the center (b), and view control panel on the right (c). The highlighted blocks (a,c) have similar functionality to the blocks on Domain level page.

In a scope of one project, it is sufficient to work with three pages: raw, study, and domain level. But since we had a need to compare processing decisions across several projects, it was convenient to keep project summaries on a separate page for faster access.

DATA FLOW

With R, designed functional blocks can be translated into Shiny modules. When working with Shiny modules, there should be a clear communication strategy between modules to avoid redundant operations (such as importing the same file several times). By designing modules to share processing milestones, we ensure that each task is performed only once and its output is passed forward efficiently (Fig. 14). This not only improves performance but also keeps the codebase cleaner and more maintainable.

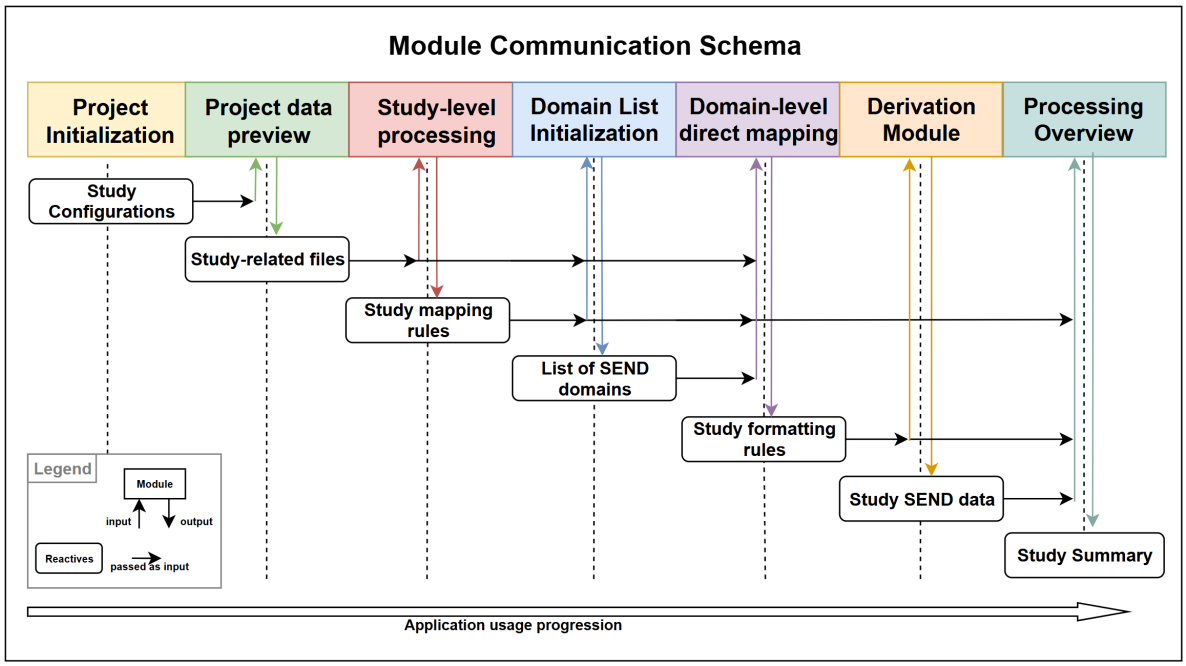


Figure 14. Communication schema between Shiny modules. Each module has a set of required reactives to execute module functionality. During application use, each module successively obtains expected reactives; with further processing and/or user input, it returns resulting reactives that are shared further until all study-related information is obtained.

USEFUL LIBRARIES

To create and export application usage logs we utilize `futile.logger` library [2]. It provides several levels of logging messages (trace, debug, info, warn, error, fatal) and ability to append records to existing logs.

For creating snapshots of study decisions and saving them to a corresponding study folder we use `pins` library [3]. This way, we keep a versioned history of changes in decisions for a given study.

CONCLUSION

To define and track data processing principles across different studies, we identified and organized processing instructions in a decision database. For controlled communication with the decision database we developed an interactive application with Shiny, minimizing human error by limiting user actions in UI and enforcing confirmed processing decisions prior to user input. With our UI-driven approach, the creation of processing instructions became more transparent and traceable, leaving us with a logged history of study decisions and results of their real-time application.

While our application was only a prototype, we successfully tested it on 50+ animal studies and:

- gathered hundreds of processing rules to decision database;
- developed a set of validated macros for metadata-driven SEND domains;
- automatically created sets of SEND domains in .XPT format;
- created audit-trail level log files.

We ensure that created SEND domains follow CDISC guidelines by supporting versioning of SENDiG and Controlled Terminology and storing only validated processing rules in decision database.

While the use of a decision database unifies data processing principles across studies and teams and makes data transformation path more transparent, communicating with it through an interactive application enables safe, consistent and traceable data processing pipelines, enabling automated domain creation. Although our current decision database structure is sufficient to work with small non-clinical studies, we plan to extend our database to include more standards and complex derivations so that the application can be also used in clinical setting.

REFERENCES

1. W. Chang *et al*. “shiny: Web Application Framework for R”. URL: https://CRAN.R-project.org/package=shiny (2025-07-03).
2. B. Rowe. “futile.logger: A Logging Utility for R”. https://CRAN.R-project.org/package=futile.logger (2016-07-10).
3. J. Silge *et al*. “pins: Pin, Discover, and Share Resources”. https://CRAN.R-project.org/package=pins (2025-04-30).

CONTACT INFORMATION

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

Author Name: Olena Olishevska
Company: EstiMates
Work Phone: +358 40 04 225 25
Email: olena.olishevska@estimates.fi
Website: estimates.fi