PharmaSUG China 2025 - Paper AA-172

ACIRA: Advancing ADaM Code Intelligence Through Intuitive Interfaces for Human-AI Collaboration

Tingting Zeng, Yaohui Zhu, Yanan Sui, Mingliang Fan, Nana Yang and You Li, BeOne Medicines

ABSTRACT

The traditional ADaM (Analysis Data Model) coding process is often time-intensive, laborious, and constrained by its complexity and study-specific demands. To overcome these challenges, we developed ACIRA (ADaM Code Intelligence for Rapid Analysis), an AI-driven (Artificial Intelligence-driven) system that automates code generation, streamlines workflows, and ensures high-quality, submission-ready deliverables with human oversight.

This paper firstly provides an overview of ACIRA's product positioning, architecture, core framework and key features. It then focuses on the design and implementation of its intuitive user interfaces (UI), which foster seamless human-AI cooperation, enhancing efficiency and ensuring quality and oversight.

Key components include the Developer Console, which provides an efficient workspace to get AI-generated codes, visually comparing them against specs and recommended examples, and track and sync upstream changes. It enables users to monitor metadata status and quality while integrating ADaM codes seamlessly. Continuous improvement is supported by collecting user feedback on AI results and facilitating regular reviews of example libraries, enhancing code accuracy and adaptability over time.

Complementing this is the Risk-Based Review module, which leverages risk assessments to help users prioritize review efforts on critical areas. It facilitates cross-reviews of integrated code against spec methods and allows quick navigation to metadata workspaces for enhanced traceability. Automated features such as code formatting and debugging reduce manual efforts, while a centralized dashboard aggregates project progress, productivity, quality metrics, and change logs, providing real-time oversight.

In the final section, we outline plans to enhance ACIRA with multi-agent approaches and stronger cross-module integration. UI improvements will focus on smarter intervention processes, core functionality encapsulation, and intelligent cross-module interactions.

INTRODUCTION

ADaM datasets are essential for regulatory submissions because they provide a traceable bridge between raw clinical data and statistical outputs. Yet producing these datasets remains a critical bottleneck, facing hurdles in the following aspects:

- **Complex derivations and time-consuming:** Creating ADaM datasets demands the integration of up-and-downstream information like SDTM (Study Data Tabulation Model) datasets, SAP (Statistical Analysis Plan), TFL (Tables, Figures, Listings) shells, and multi-level standards with complex logics. Manual checks here are not efficient and prone to error.
- **Frequent changes lack version control:** It's not a one-time effort and often faces changing requirements, leading to heavy communication and risk of misalignment between specs, codes and outputs. Without clear version control, it's not easy to track and audit each modification.
- **Limited reusability:** Across studies, it's difficult to reuse scattered specs and codes due to diverse needs, free styles, lack of connection, and no efficient way to compare across studies or against standards.

To tackle these issues, ACIRA is designed to transform the creation of analysis-ready datasets and optimize clinical programming workflows within a harmonized ecosystem, with a focus on three strategic pillars:

- **AI and Digital-Driven Efficiency:** ACIRA leverages AI technology and reusable digital libraries to maximize efficiency. Its interactive interface enables seamless collaboration between metadata, AI, and human expertise for complex analysis needs.
- **End-to-End ADaM Solution:** By cooperating with SpecMaster (an intelligent platform to manage ADaM specifications), ACIRA aims to create an intelligent platform for the entire ADaM lifecycle, boosting efficiency, quality and compliance.
- **Connected Data Ecosystem:** With ADaM serving as a central connector linking workflows across SDTM and TFL, it targets to establish an interconnected ecosystem with real-time alert and response features.

This paper provides an overview of the system's workflow, key features including the centralized metadata, AI-powered collaboration, and automated code integration. It details the design and implementation of system intuitive interfaces, especially about how to develop and utilize the UIs to foster seamless human-AI cooperation, enhance efficiency while ensuring quality and oversight.

SYSTEM WORKFLOW: HOW ACIRA WORKS

ACIRA combines AI-driven code generation with human oversight to produce high-quality SAS programs and datasets. Its dynamic metadata repositories, automated code integration features, and user-friendly interfaces ensure maximum efficiency while maintaining accuracy. Below is the flowchart for the system.

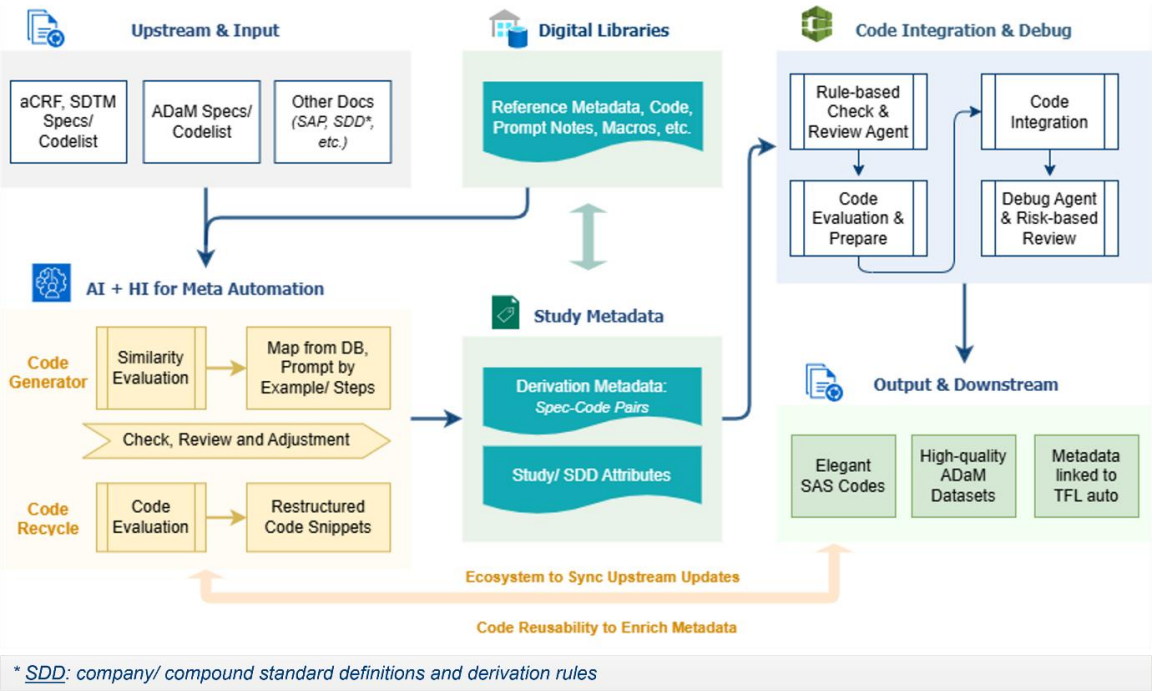

Figure 1. Flowchart of ACIRA

SYSTEM KEY FEATURES

To address the key challenges inherent in ADaM coding, ACIRA is designed with a focused approach to simultaneously enhance efficiency and ensure high-quality outputs. Below, we introduce the key features, each contributing to a streamlined and robust coding process.

1. Centralized Metadata Repository

The foundation of ACIRA system is a centralized metadata repository, a relational architecture designed to host validated code snippets which are meticulously linked to various derivation methods and study designs. It also stores system prompt notes which are dynamically provided to AI to improve accuracy during the code generation and review process. In addition, through its built-in version control and cross-module association capabilities, it enables the monitoring of upstream and downstream changes, provides change notifications, and supports real-time synchronization.

2. AI-Powered Collaboration

Harnessing the RICCE framework (a mnemonic device for the five key elements of role, instruction, context, constraints, and examples to consider when crafting prompts), ACIRA provides context-specific code suggestions that evolve with continuous user feedback. Additionally, we're developing AI review, debug and reflection agents that together identify code defects, classify and resolve errors, and capture manual interventions to refine prompts and enhance overall code quality.

3. Automated Code Integration

Our system goes beyond merely creating and storing code snippets; it meticulously evaluates and integrates them into comprehensive, submission-ready ADaM programs. Through a sophisticated evaluation of dependencies and logical relationships, the system assesses the source data and proper sequency in which snippets should be combined, determines the possibility of unifying them into a single step, considers whether pre-/post- data handling is necessary, and flexibility sets dataset names to optimize integration. This process ensures that the final code is not just an accumulation of fragments but is designed with a clear structure, high readability, and maintainability.

4. User-Friendly Interfaces

Featuring an interactive interface, our system facilitates real-time monitoring, specification/code reviews, change tracking, and progress visualization. It provides an intelligent workspace where humans and AI collaborate efficiently, ensuring that metadata is accurately generated and integrated. The interface also includes a quality control center for integrated code, assessing risks and assisting in review and edits. The centralized dashboard enables users to track project progress, efficiency, changes, and quality indicators.

This overview sets the stage for a more detailed examination of the user interface in the following sections. We will delve deeper into how these features are designed to foster seamless human-AI cooperation, enhancing efficiency while ensuring quality and oversight.

INTELLIGENT WORKSPACE FOR CODE GENERATION

The intelligent workspace serves as a dynamic platform for AI-human collaboration, enabling efficient generation and management of ADaM programs. The workspace is composed of various modules designed to streamline processes and improve user interaction with AI components.

OVERALL REVIEW AND BULK OPERATIONS

Upon accessing the homepage, users are presented with a list of projects they are authorized to view. By clicking on a specific project, they can review the overall status of all ADaM datasets and enter the interface for one ADaM dataset (as illustrated in Figure 2). This serves as an important component, encompassing several key functions automated code generation, specification and code reviews, metadata real-time checks, and review/ synchronization of upstream changes.

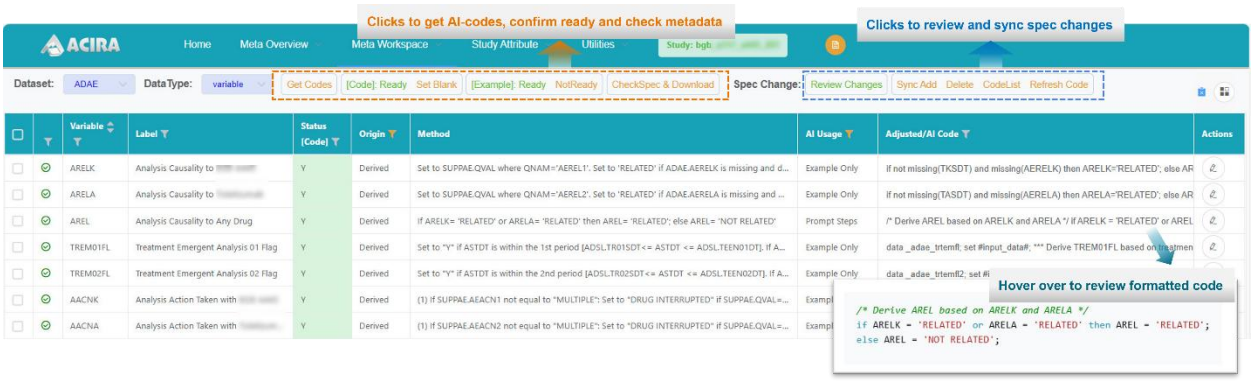

Figure 2. Overall Review and Bulk Operations Interface

1. Automated Code Generation

Users can easily obtain AI-generated code by clicking the 'Get Codes' button. The system efficiently processes batch AI tasks for multiple variables at once, leveraging SDTM/ADaM specifications and the centralized digital library to choose the optimal code generation strategy. Users can track the progress of the code generation on the platform.

2. Specification and Code Reviews

Users can review the AI-generated codes together with key elements of ADaM specification, to decide if it is ready for integration or promotion as examples for other projects. To facilitate the review, users can choose to show or hide certain columns and hover over cells for better formatting, such as syntax-highlighted and well-formatted codes.

3. Metadata and Code Real-time Checks

The interface enables real-time checks on metadata and code for compliance and potential issues, such as cross-dataset dependency errors (e.g., variables or parameters referenced but absent in source SDTM or ADaM datasets). Results from these checks are presented with highlighted alerts and can be downloaded for user review, aiding in the assessment of derivation metadata's maturity.

4. Review and Synchronize Specification Changes

Additionally, users can review upstream specification changes by spotlighting newly added or deleted variables and any specific modifications. With the database linking every version of code and specification, the system allows real-time tracking even with both are under progress. Users can also synchronize upstream changes by clicking one button to prompt AI components to reevaluate and generate updated code.

VARIABLE/ PARAMETER-SPECIFIC REVIEW AND EDITING

While the previous interface is geared towards bulk review and processing, this key interface assists users dive deeper into reviewing and adjusting of AI-generated code for specific variables/ parameters, as shown in Figure 3.

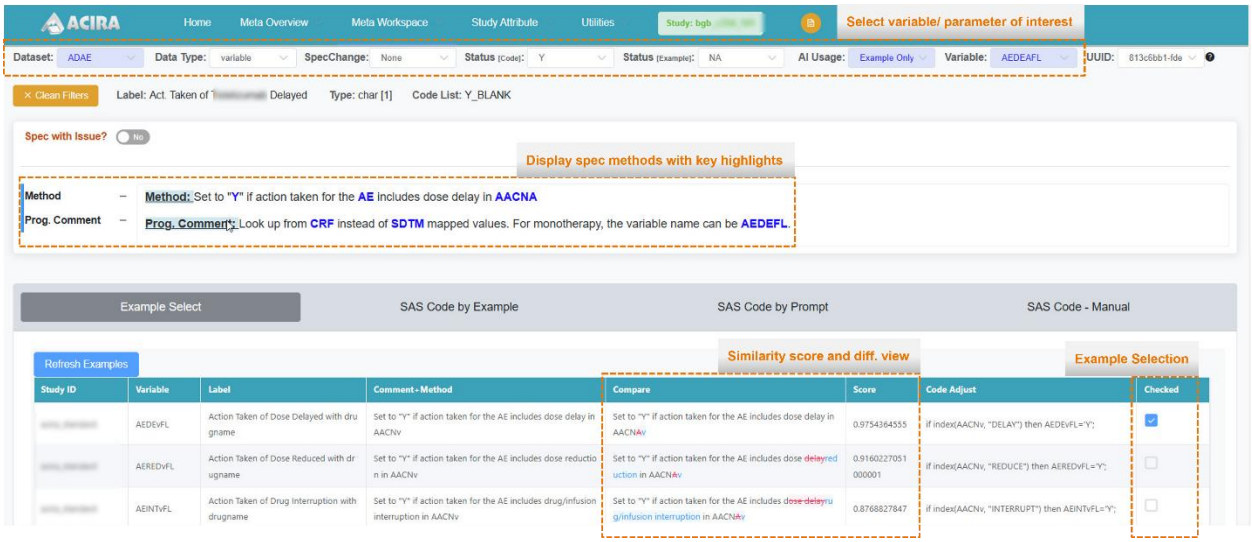

Figure 3. Interface for Variable/Parameter-Specific Review and Example Selection

1. Variable/Parameter Selection and Specification Review

Users can select variables/parameters of interest using filters and review their derivation methods, with key information highlighted. If there's un-synced changes, the system visualizes these differences for user reference. Additionally, users can identify potential issues within the specification itself and communicate them to the upstream SpecMaster platform for revisions.

2. AI Code Generation Process and Results Review

Users can navigate subpages to review the AI code generation process and outcomes. For example, subpage 1 (see Example Select in Figure 3) presents AI-recommended examples based on semantic similarity assessments, highlighting differences with the target variable and allowing users to adjust example selections accordingly.

Three additional subpages provide detailed insights into code generation. For example-modified code (see Figure 4), the UI highlights revisions in AI-generated codes, assisting users in determining the necessity for further modifications. Users can rate, provide feedback, and interact with the AI to enhance code quality. They can also make direct code modifications in the editing interface, where before-and-after comparisons are clearly displayed.

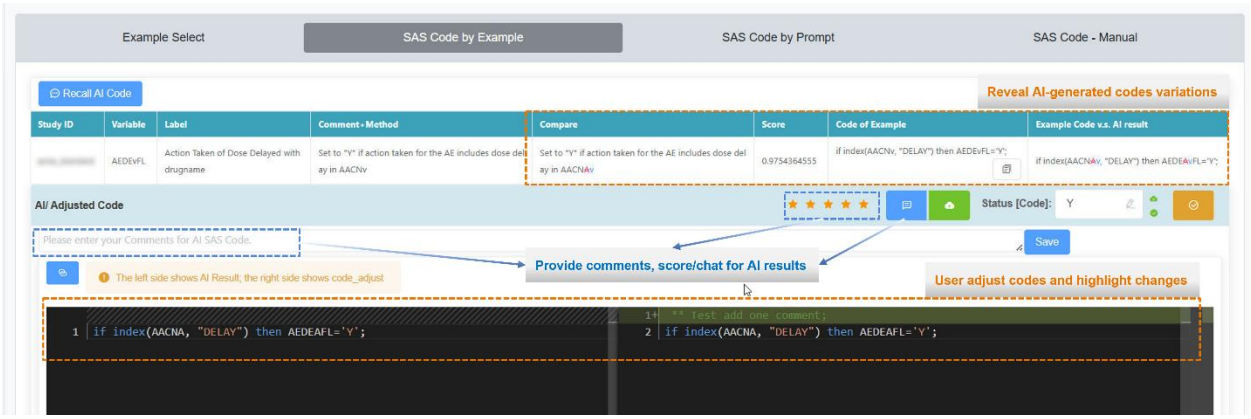

Figure 4. Example-Modified AI-Generated Code Interface

SUPPLEMENTARY INTERFACES

Additional interfaces include the 'Study Attribute UI', which allows for overarching project design settings that affect multiple code generations, such as configurations for randomized/non-randomized studies, single/combo treatments, and central/local lab tests. The 'SDD Attribute UI' addresses the implementation of specific derivation guidelines outlined in compound or study documents, such as for the derivation rules of PFS (progression free survival) endpoints.

The system utilizes document parsing alongside AI assistance to automate code generation, reflecting project design and SDD specifications. The interfaces also aid review through visual contrast and rendering effects, such as the impact of study attributes on multiple variables.

Overall, the UI platform embodies effective AI-user interaction, streamlining code generation and updates while enhancing feedback mechanisms for quality improvement. Visual highlights and comparative displays empower users to evaluate code reliability and make informed decisions with ease.

COMPREHENSIVE CODE INTEGRATION AND REVIEW

This section introduces key interfaces dedicated to integrating code snippets into complete programs, ensuring quality control through evaluation and review mechanisms. It encompasses several core functions as described below and will integrate AI debug and reflection agents which are under development.

STATUS REVIEW AND ONE-CLICK CODE INTEGRATION

After reviewing AI-generated code snippets and checking metadata maturity in the previous section, users can view the overall completion status of ADaM datasets on the interface. With the click of a button, they can initiate the backend process for evaluating and integrating code snippets into completed ADaM programs, transitioning to the review platform, as shown in Figure 5.

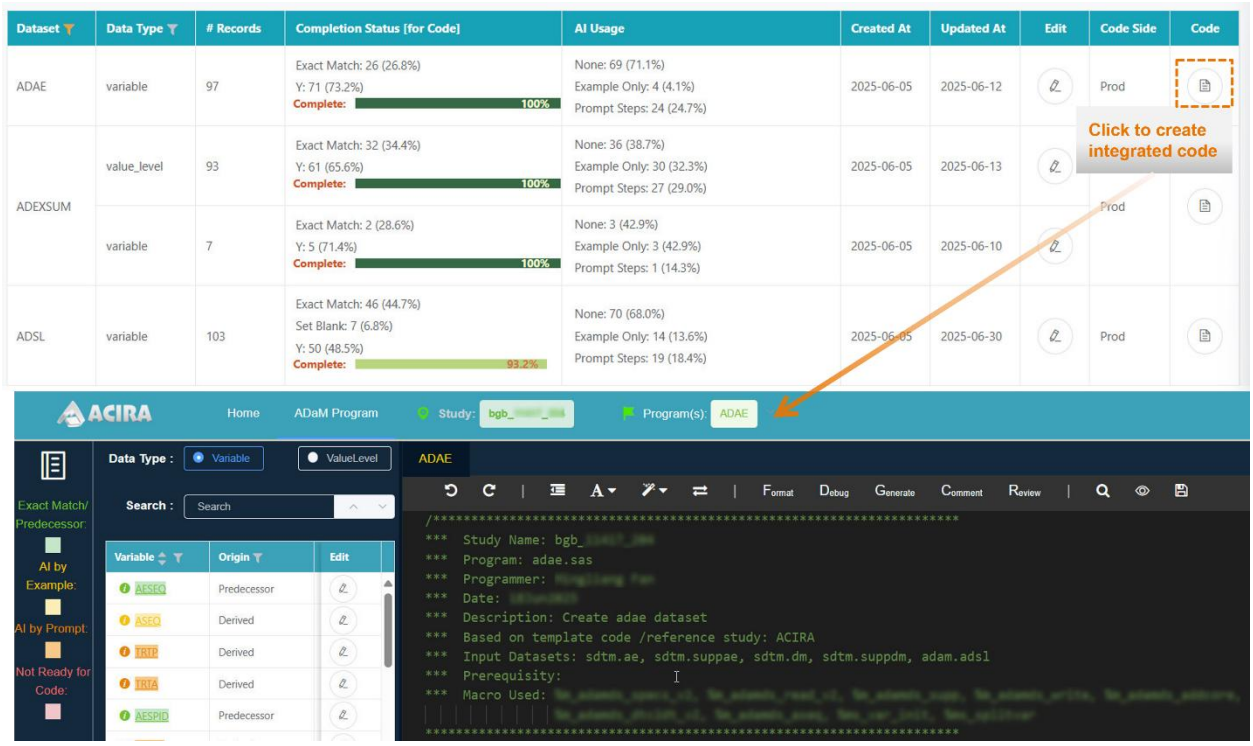

Figure 5. Status Review and Code Integration Interface

CROSS REVIEW OF SPECIFICATION AND INTEGRATED CODE

The interface presents a list of variables and parameters, including risk assessment results based on AI code generation strategies and check/review issues, facilitating focused review by users. Users can click on specific variables to navigate directly to corresponding code positions and utilize the information button for a cross-review with the specific derivation method of the variable/parameter, as illustrated in Figure 6.

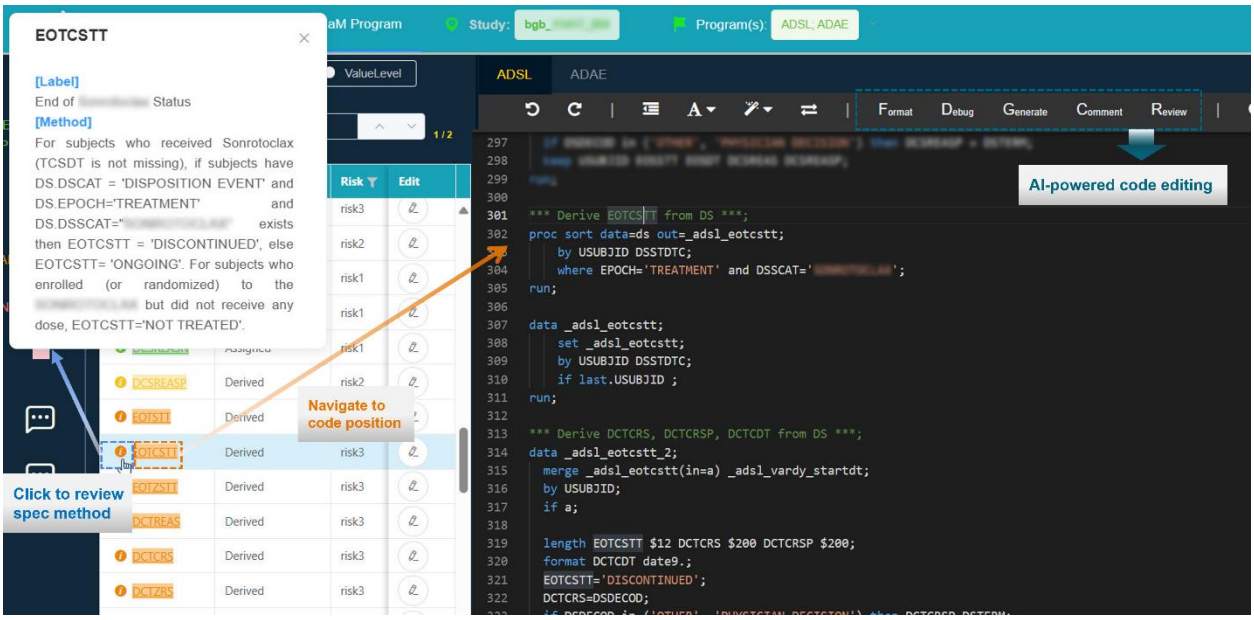

Figure 6. Interface for Risk-Based Review and AI-Assisted Code Editing

AI-ASSISTED EDITING INTERFACE

ACIRA integrates functionalities from the CodeGenius system (an AI-powered SAS copilot with internal resource integrated), offering AI-driven features for code formatting, review, and debugging (see Figure 6). The developing AI reflection agent will continuously monitor, evaluate, and incorporate user modifications back to the centralized metadata, ensuring consistency and optimizing quality. During the review process, if users encounter uncertainties or need to adjust code snippets, they can seamlessly return to the workspace page for the corresponding variable/parameter with a single click.

CENTRALIZED PROJECT DASHBOARD

This section presents a centralized dashboard designed to oversee multiple projects utilizing the ACIRA platform. It offers comprehensive insights into cross-therapeutic area (TA) project applications, progress, efficiency and quality, along with analytical insights for optimizing systems or processes.

OVERVIEW OF ACIRA SYSTEM PERFORMANCE

In this part, users can review all or selected projects to obtain a summary of overall statuses and key metrics. These include project progress, efficiency metrics, quality insights, and upstream-downstream interactions. Users can also download comprehensive reports that provide both overall and module-specific analyses, as illustrated in Figure 7.

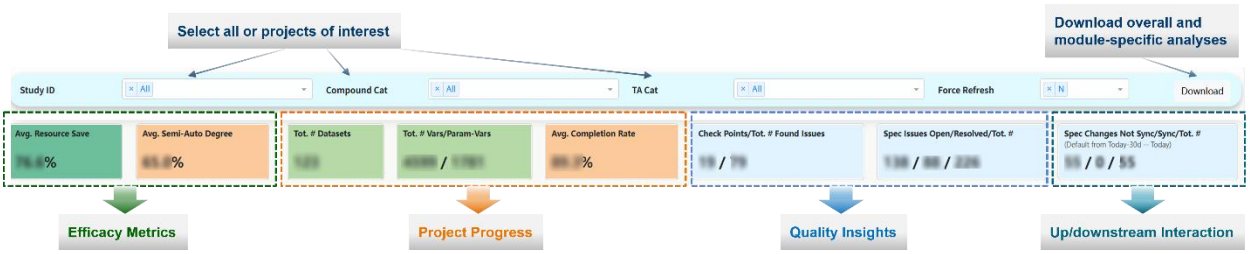

Figure 7. Summary of ACIRA System Performance Metrics

DETAILED MODULE ANALYSIS AND KEY INDICATORS

By selecting various cards shown in Figure 7, users can access in-depth graphical analyses for each component, offering valuable insights for overall management and optimization. This includes:

1. Project Coverage and Progress Analysis

Users can view projects across TAs supported by ACIRA, with usage distribution by quarter. The interface visually represents the number of system-supported datasets, variables, and parameters, along with their completion status, as shown in Figure 8.

Figure 8. Project Coverage and Progress

2. Efficiency Metrics and Comparative Analysis

As illustrated in Figure 9, users can examine resource consumption by datasets, compare the saving rate against the planned time, and analyze inter-project metrics. The interface also provides reference data for further analysis, such as automation levels and time consumption across various workflow stages.

Figure 9. Efficiency Metrics and Comparative Analysis

3. Quality Check Insights

As shown in Figure 10, users can view key rules and results of the system's automated checks, alongside issues recorded during the code generation and review stages—such as specification issues or comments on AI-generated codes. The interface presents an overview of how different issues distributed across studies, also facilitates users to explore details or link to a comments-tracker for easy issue recording and resolution tracking.

Study ID	Dataset	Variable	Spec Issue Details	Issue Status
bg_	ADCM	CMCLASCD		Open
bg_	ADSL	EFFFL		Open
bg_	ADTR	VISIT		Open
bg_	ADTR	VISITNUM		Open
bgb	ADAE	AACNC		Resolved
bgb	ADAE	AACNC		Open
bgb	ADAE	AACNOTH		Resolved

Figure 10. Tracking of Quality Issues Across Projects

4. Upstream and Downstream Interaction Analysis

Depicted in Figure 11, users can monitor the overall upstream (e.g. ADaM specifications) changes during the selected time-period across studies, alongside the downstream synchronization status within ACIRA. The interface allows for one-click navigation to review specific changes and sync details.

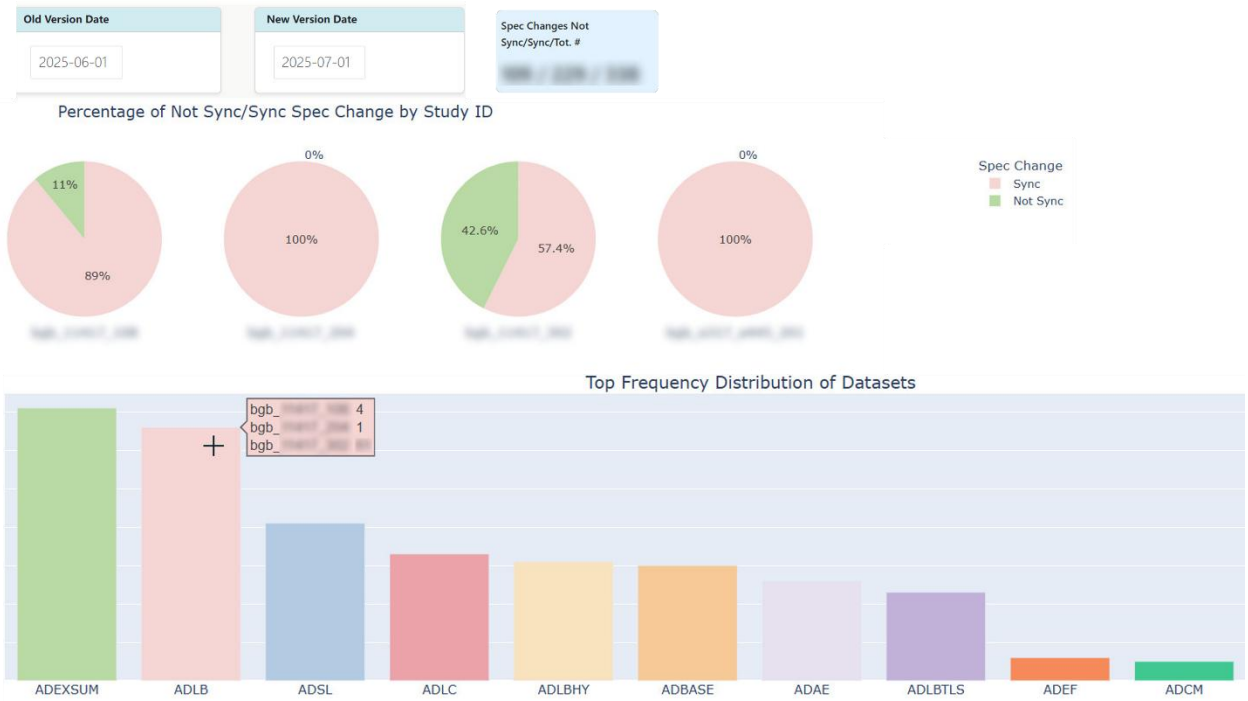

Figure 11. Monitoring and Synchronization of Upstream and Downstream Changes

In summary, these modules provide visual insights from macro to micro perspectives, ensuring that project progress, quality, and consistency align with requirements at critical milestones. They also offer valuable insights for future optimizations in platforms and processes.

TECHNICAL FRAMEWORK

This section provides an overview of the technical framework that underlies the ACIRA system. Developed within the Integrated SAS Programming Platform (ISPP), ACIRA shares a unified technology infrastructure with other products. This integration enhances resource sharing, including relational databases and server infrastructure, ensuring seamless data flow and cohesive workflows across various components.

FRONTEND LAYER

The frontend of ACIRA is crafted using JavaScript, incorporating multiple powerful frameworks and tools to enhance user experience, such as:

- **Vue.js:** A JavaScript framework for building user interfaces. It is built on top of standard HTML, CSS, and JavaScript and provides a declarative, component-based programming model for developing UIs efficiently. In ACIRA, Vue.js serves forms the foundation of the web framework, facilitating dynamic data binding and efficient state management to enhance interactivity.
- **Element-UI:** A Vue-based component library that simplifies the design of intuitive interfaces. In ACIRA, Element-UI supports the rapid development of user-friendly interfaces through its extensive suite of pre-built components such as forms, tabs, and data tables.
- **VXETable:** A high-performance table/grid component for Vue.js. ACIRA leverages VXETable to manage complex data operations and visualization tasks, ensuring data integrity and readability across large datasets.

- **Monaco Editor:** A powerful code editor component used in various programming environments. It enhances code editing functionalities within the ACIRA interface, offering features like real-time syntax highlighting, comprehensive editing capabilities, and effective code comparison.

BACKEND AND DATABASE LAYERS

The backend of ACIRA is developed in Python, incorporating several frameworks and tools:

- **Flask and SQLAlchemy:** Leveraging Flask as a lightweight and modular web framework, paired with SQLAlchemy for comprehensive database interaction, the ACIRA backend efficiently manages data exchange and ensures traceable data operations with features like audit trails and version control.
- **Celery and Redis:** These tools enable efficient task queuing and execution, particularly for AI-related tasks such as code generation. Celery handles task distribution, while Redis manages task queues.
- **Dash:** An original low-code framework for rapidly building data apps in Python. ACIRA utilizes Dash to develop its centralized dashboard, capitalizing on its capability to rapidly prototype sophisticated data visualizations and interactions, offering a comprehensive and dynamic view of data analytics.
- **Diff-Match-Patch:** The libraries offer robust algorithms to perform the operations required for synchronizing plain text. ACIRA utilizes it to efficiently detect and highlight changes between different versions of ADaM specifications, enabling clear visualization in user interfaces.

AI ENGINE LAYER

- **Databricks-LLMs:** ACIRA's AI applications are built based on Databricks, a platform known for its seamless integration with a wide array of large language models (LLMs). This allows ACIRA to efficiently harness powerful AI capabilities, ensuring robust performance and flexibility in AI-driven processes.
- **LangGraph:** As a low-level orchestration framework associated with LangChain, LangGraph specializes in creating, managing, and deploying long-running, stateful agents. ACIRA uses this cutting-edge framework to develop AI agents focused on code review and debugging, enabling sophisticated automation and intelligent decision-making within the system.

CONCLUSION

ACIRA offers a transformative solution to the challenges of ADaM coding by streamlining workflows and enhancing the efficiency and quality of the process. By integrating centralized data management with AI-fueled intelligence, the system delivers comprehensive and adaptable code generation that meets diverse study demands with precision.

The user interfaces are thoughtfully designed to optimize AI-human collaboration, enabling seamless integration, real-time monitoring, and strategic project oversight. Through them, users gain clear insights and maintain control over project trajectories and quality benchmarks.

Looking ahead, ACIRA's commitment to evolving its capabilities, particularly through enhanced multi-agent systems and deeper cross-module synergies. This evolution not only streamlines workflows but also elevates the potential for innovation and excellence throughout the entire clinical data analysis lifecycle.

REFERENCES

Vue.js. "Vue.js - The Progressive JavaScript Framework | Vue.js." https://vuejs.org/.

Element-UI. " Element - A Desktop UI Toolkit for Web” https://element.eleme.io/.

Element Plus. "A Vue 3 UI Framework | Element Plus” https://element-plus.org/.

VXETable. "VXETable GitHub Repository." https://github.com/x-extends/vxe-table/.

Monaco Editor. "Monaco Editor GitHub Repository." https://github.com/Microsoft/monaco-editor.

Flask. "Welcome to Flask — Flask Documentation.” https://flask.palletsprojects.com/.

SQLAlchemy. "SQLAlchemy - The Database Toolkit for Python." https://www.sqlalchemy.org/.

Dash. "Dash Documentation & User Guide | Plotly" https://dash.plotly.com/.

Diff-Match-Patch. "Diff Match Patch GitHub Repository." https://github.com/google/diff-match-patch.

Celery. "Celery - Distributed Task Queue." https://docs.celeryproject.org/.

Redis. "Redis: The Real-Time Data Platform." https://redis.io/.

Databricks. "Databricks: Leading Data and AI Solutions for Enterprises." https://databricks.com.

LangGraph. "LangGraph." https://langchain-ai.github.io/langgraph/.

ACKNOWLEDGMENTS

We would like to express our sincere gratitude to the members of the ACIRA Collaborative Committee, including Jia He, Aide Zhou, Zhijuan Yu, Meng Qu, and Li Zhou, for their invaluable suggestions on project development strategy, platform optimization, and especially for their coordination and guidance in the practical implementation within clinical projects. Furthermore, we extend our appreciation to the leadership team for their strategic guidance and continuous support throughout the project.

CONTACT INFORMATION

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

Tingting Zeng
BeOne Medicines (formerly BeiGene)
tingting.zeng@beigene.com