



CDISC India Day

A cloud-native, containerized clinical programming platform that provides on-demand SAS/R environments, integrated with GIT-based source control and governed clinical data access

Presented by, Prasanna Parthasarathy, Senior Principal Stat Programmer, Syneos Health



Agenda

- ✓ Synopsis
- ✓ Hybrid Model - SAS-Centric to Flexible, Reproducible Analytics.
- ✓ High-level architecture of Containerized Development Environment (CDE)
- ✓ Kubernetes Made Simple: The Hotel Analogy
- ✓ Workbench Launch Flow
- ✓ Docker Image: A Preconfigured Software Environment
- ✓ Open-Source Pharamaverse to Enterprise R Workbench
- ✓ End-to-End SAS & R Development Workflow
- ✓ Role of Python in CDE
- ✓ Key Takeaways



Speakers



Syneos Health

Prasanna Parthasarathy is a Senior Principal Statistical Programmer at Syneos Health, with over 16 years of experience in the CRO industry. Currently, he serves as a Study Lead Programmer and has significant experience working with next-generation tools, particularly R. He played a pivotal role in assisting the client with setting up the pilot version of the R transformation and Hybrid model and its associated tools. Within Syneos Health, Prasanna has been recognized for his innovation, proactiveness, and leadership qualities. In the past, he has presented at multiple conferences:

- ✓ 2017, Delhi – Future of Clinical Trials and SDTM Standards
- ✓ 2021, December (Virtual Event) – Robotic Process Automation
- ✓ 2022, Virtual Summer Event – Considerations from the FDA and the CTTI for Performing a Decentralized Trial (DCT)
- ✓ 2023, ISCR_Admiral_workshop
- ✓ 2025, CDISC India Day - Modular programming of SDTM in R using {sdm.oak}



Disclaimer and Disclosures

The views and opinions expressed in this presentation are those of the author(s) and do not necessarily reflect the official policy or position of CDISC

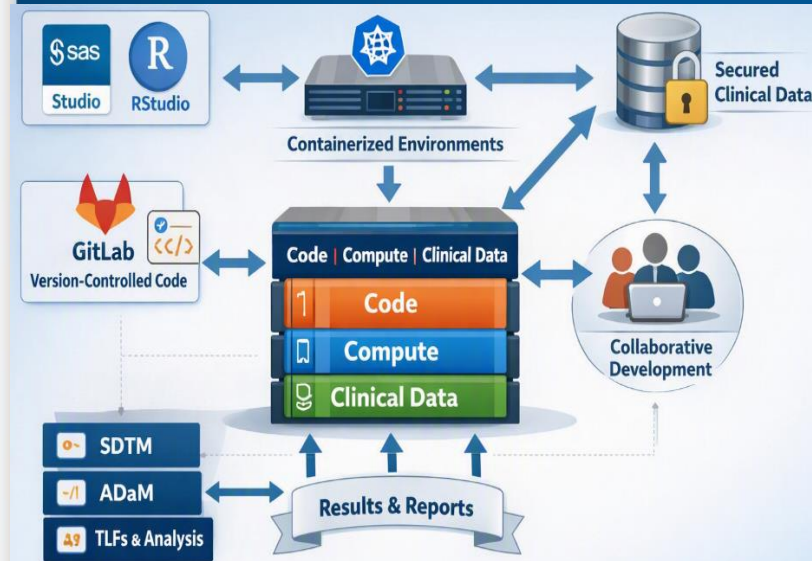




Synopsis

This presentation describes a modern clinical programming environment that combines SAS and R within a containerized platform, enabling programmers to leverage the strengths of both technologies while maintaining regulatory compliance, reproducibility, security, and collaborative development practices. The ecosystem separates code, compute environments, and clinical data into distinct managed components.

Programmers can launch SAS Studio or RStudio on demand through containerized environments, access version-controlled code through GitLab, and connect to secured clinical data repositories through governed access mechanisms. This architecture supports efficient development of SDTM, ADaM, Tables, Listings, Figures (TLFs), and exploratory analyses while ensuring consistency across users and projects





Hybrid Model - SAS-Centric to Flexible, Reproducible Analytics

Historically, clinical programming environments were heavily SAS-centric.

However, modern clinical development increasingly requires:

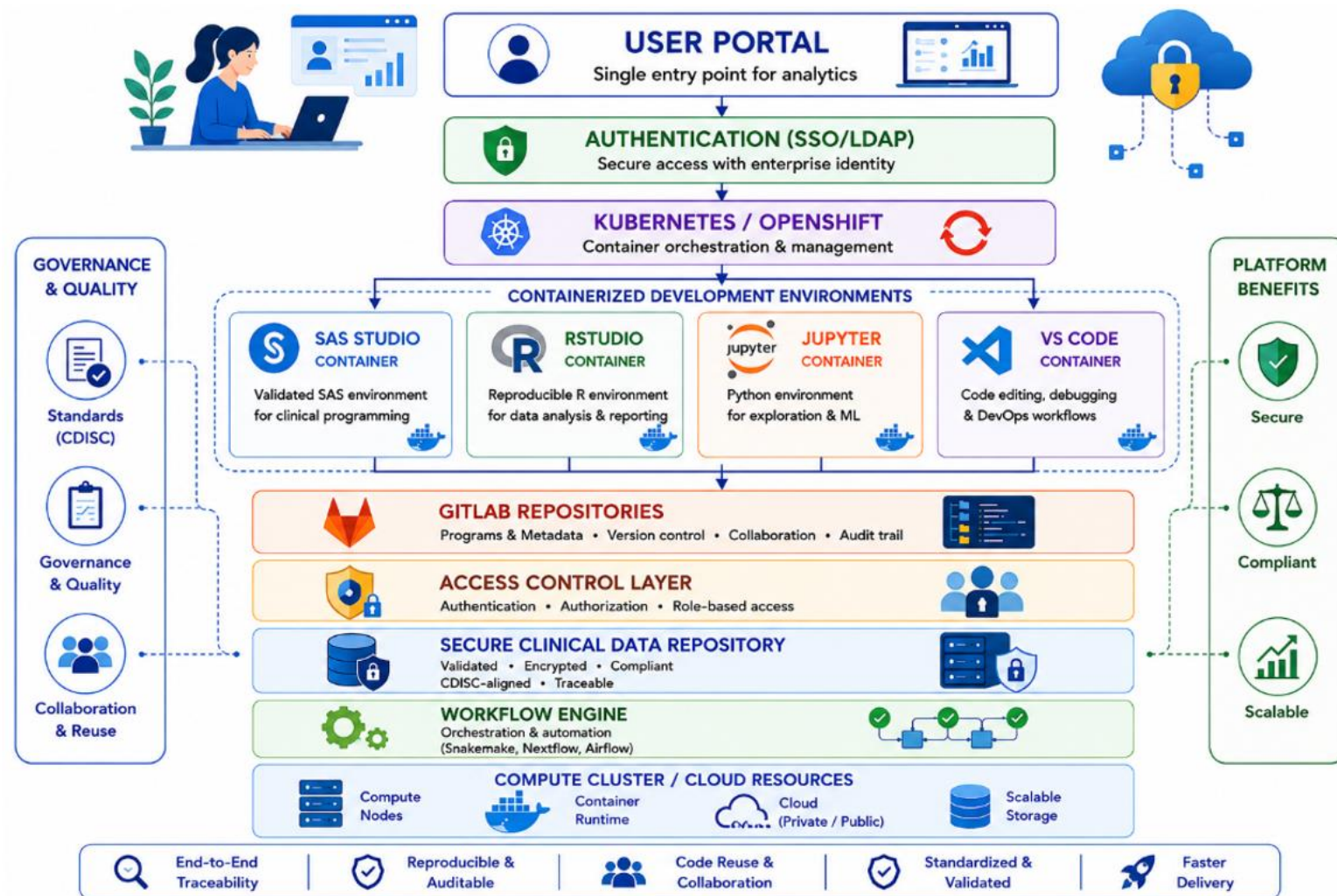
- Advanced visualization
- Interactive reporting
- Open-source innovation
- Reproducible research
- Scalable automation
- Collaborative software development practices

Rather than replacing SAS, the hybrid model allows organizations to leverage:

- Open-source innovation
- Advanced graphics
- Rich package ecosystem
- Reproducible analytics
- Interactive outputs



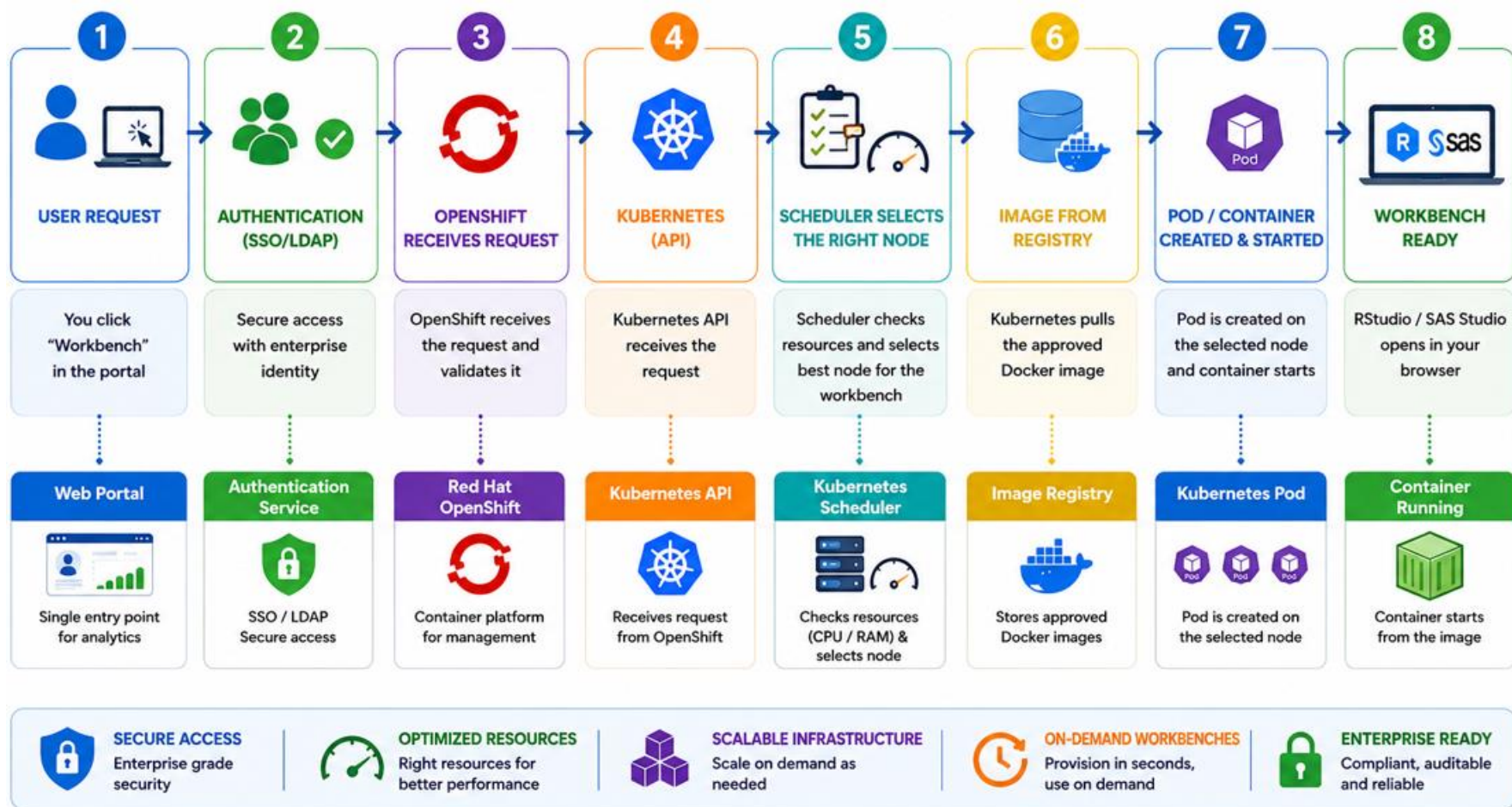
High-Level Architecture of Containerized Development Environment (CDE)



Kubernetes Made Simple: The Hotel Analogy

TECHNOLOGY	HOTEL ANALOGY
 Kubernetes	 Hotel manager receives your booking and decides where you'll stay.
 Node	 The manager selects a hotel building with an available room.
 Docker Registry	 If the building doesn't have the standard room setup, it retrieves the room design/furniture from the central warehouse (Docker Registry).
 Docker Image	 The standard room design used to prepare every room. It defines exactly how the room should look.
 Pod	 The manager assigns you a hotel room (Pod) where you'll stay.
 Container	 You enter the fully furnished room (the running application).

Workbench Launch Flow

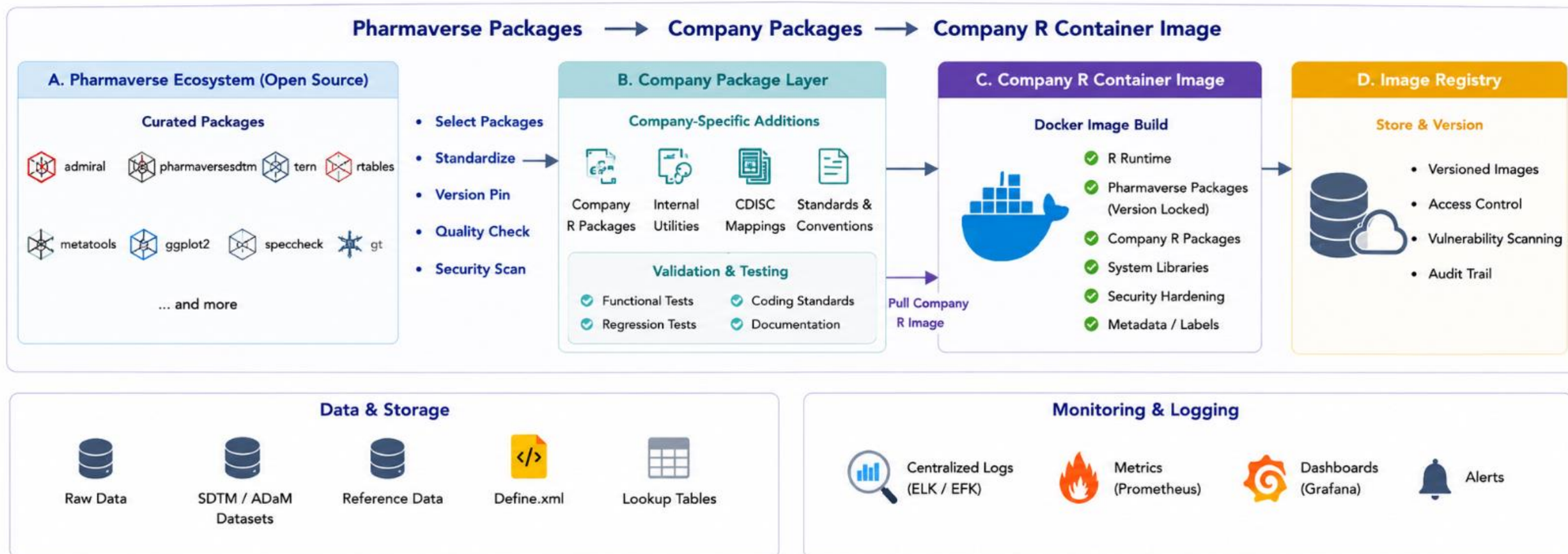


Docker Image: A Preconfigured Software Environment

	A Docker image is a read-only template that contains everything needed to run an application in a container.	
	Ubuntu Linux	Base operating system providing the core environment.
	R 4.4	The R programming language and runtime environment.
	RStudio Server	Integrated development environment for R.
	Python	Python runtime and libraries for scripting and automation.
	Git	Version control system for code management.
	Validated R Packages	Pre-installed and tested R packages for analytics.
	Company Libraries	Internal libraries, modules, and business logic.
	Configuration Files	Environment settings and application configurations.
	Startup Scripts	Scripts to initialize services and start applications.

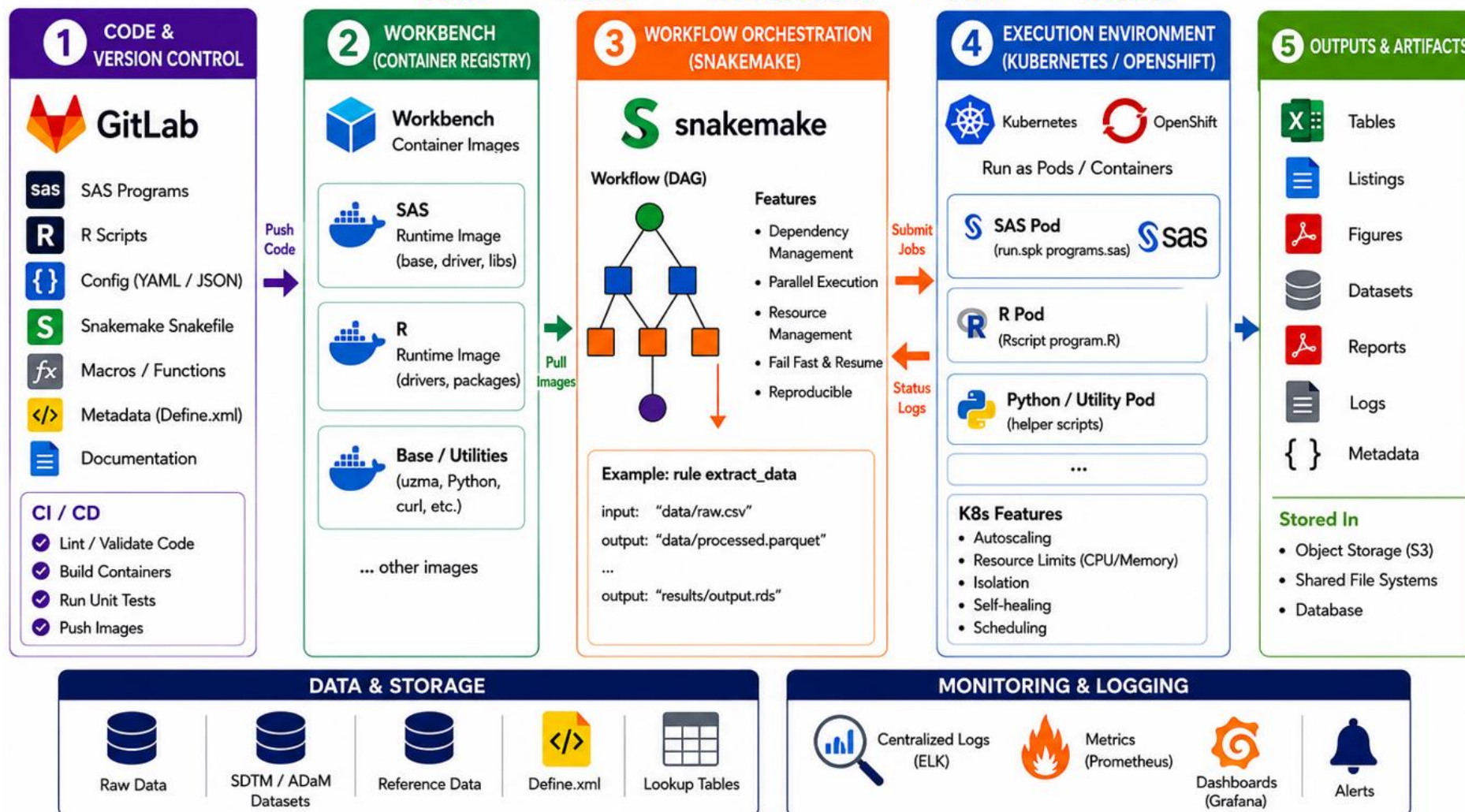


Open-Source Pharmaverse to Enterprise R Workbench

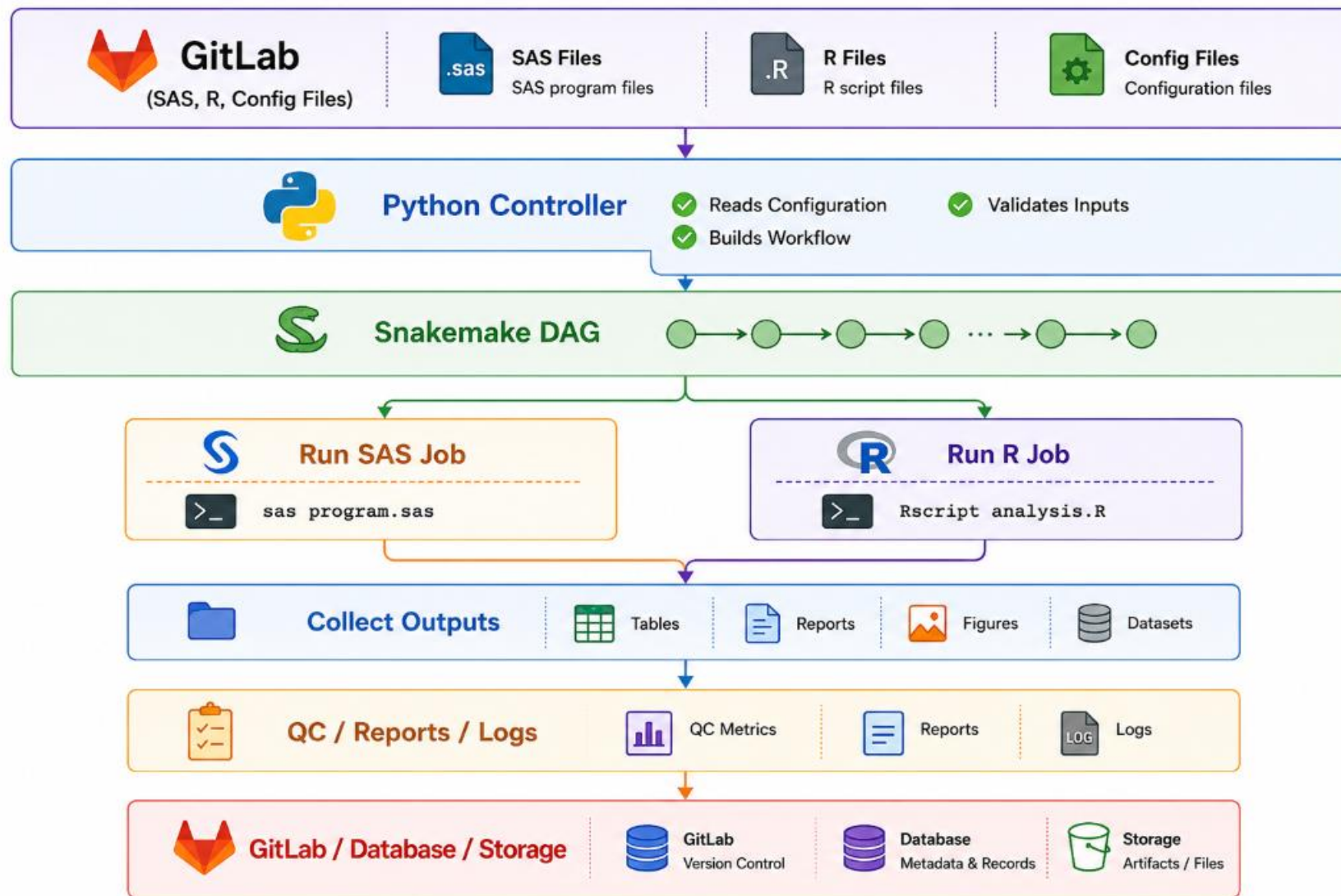


End-to-End SAS & R Development Workflow

Code → Build → Orchestrate → Run → Results



Role of Python Programming in CDE





Key Takeaways

✓ Hybrid Development Model

- Leverages the strengths of both SAS (validated production programming) and R (advanced analytics and visualization).

✓ Containerized Development Environment (CDE)

- Provides consistent, isolated, and reproducible development environments across users.

✓ Cloud-Native Technologies

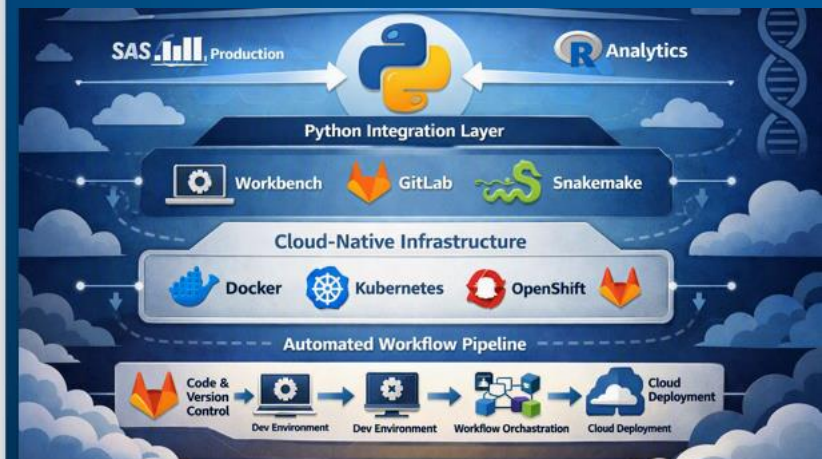
- Docker, Kubernetes, OpenShift, GitLab, and Snakemake enable scalable, automated, and collaborative workflows.

✓ Integrated Development Workflow

- GitLab manages code, Workbench provides standardized development environments, and Snakemake orchestrates end-to-end execution.

✓ Python as the Integration Layer

- Connects SAS, R, workflow orchestration, and cloud infrastructure into a seamless automated pipeline.



Appendix I

KEYWORD	ONE-LINE EXPLANATION	KEYWORD	ONE-LINE EXPLANATION
 Cloud Computing	Delivery of computing services—such as servers, storage, databases, networking, software, and analytics—over the internet (“the cloud”).	 Node	A physical or virtual server where Pods and containers run.
 Cloud-Native	Applications designed to run efficiently in cloud environments using containers, microservices, and automation.	 Pod	The smallest deployable unit in Kubernetes that contains one or more containers.
 Container	An isolated, lightweight environment that runs an application with all its dependencies.	 Image Pull	Downloading a Docker image from a registry to a node before starting a container.
 Docker Image	A reusable package containing the application, libraries, configuration, and runtime needed to create a container.	 Git	A version control system for tracking code changes and collaboration.
 Docker Container	A running instance of a Docker image.	 GitLab	A platform for Git repositories, code reviews, and CI/CD pipelines.
 Docker Registry	A repository where Docker images are stored and retrieved.	 CI/CD	Automated process for building, testing, and deploying applications.
 Kubernetes	A platform that automatically deploys, manages, and scales containerized applications.	 Workflow Orchestration	Automating the execution of tasks in a predefined sequence (e.g., Snakemake).
 OpenShift	Red Hat’s enterprise platform built on Kubernetes, providing security, monitoring, and user management.	 RStudio / SAS Studio / Jupyter	Browser-based development environments launched inside containers.
		 Workspace (Workbench)	A personalized development environment created for each user to perform programming and analysis.

Thank You!



Questions?

