20TH ANNIVERSARY EDITION

WSO2CON ASIA

PLATFORMLESS MODERNIZATION

Building a Robust Enterprise Integration Architecture

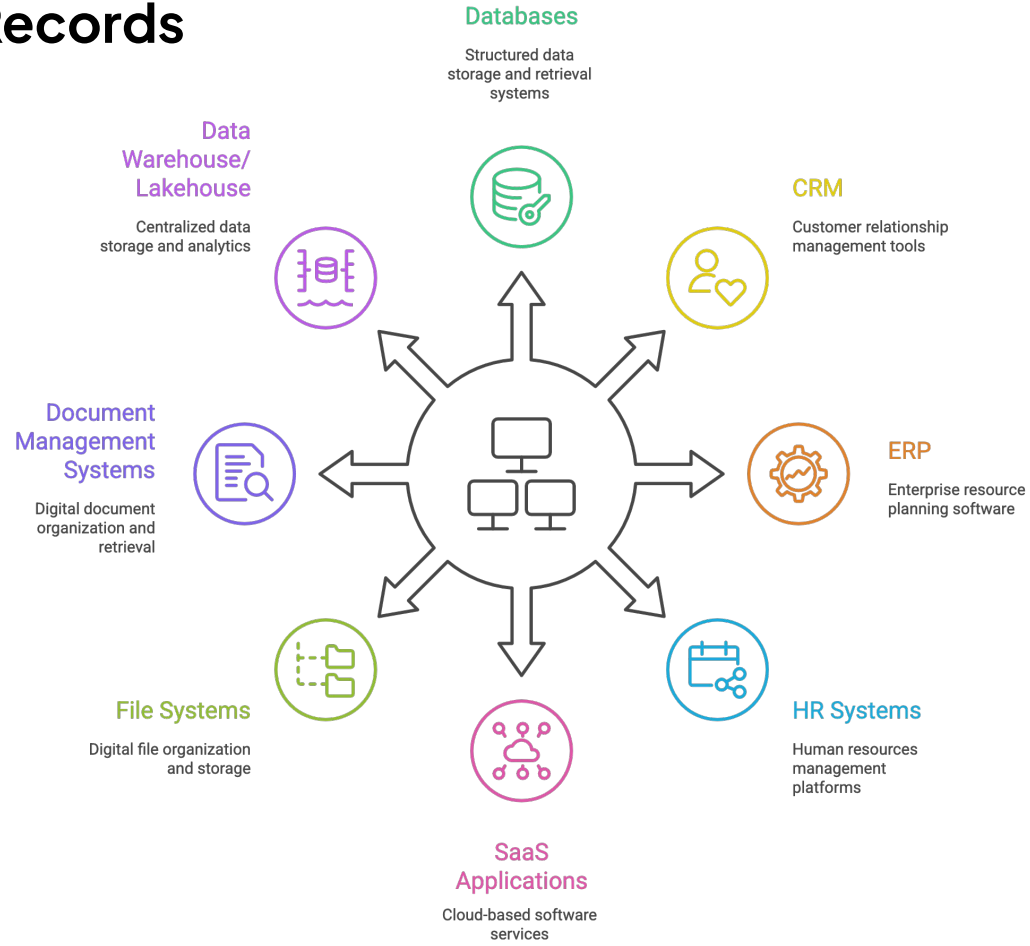


Shazni Nazeer
Director - Solutions Architecture
WSO2


Modern enterprises are distributed,
multi-cloud enabled and data-rich



Systems of Records

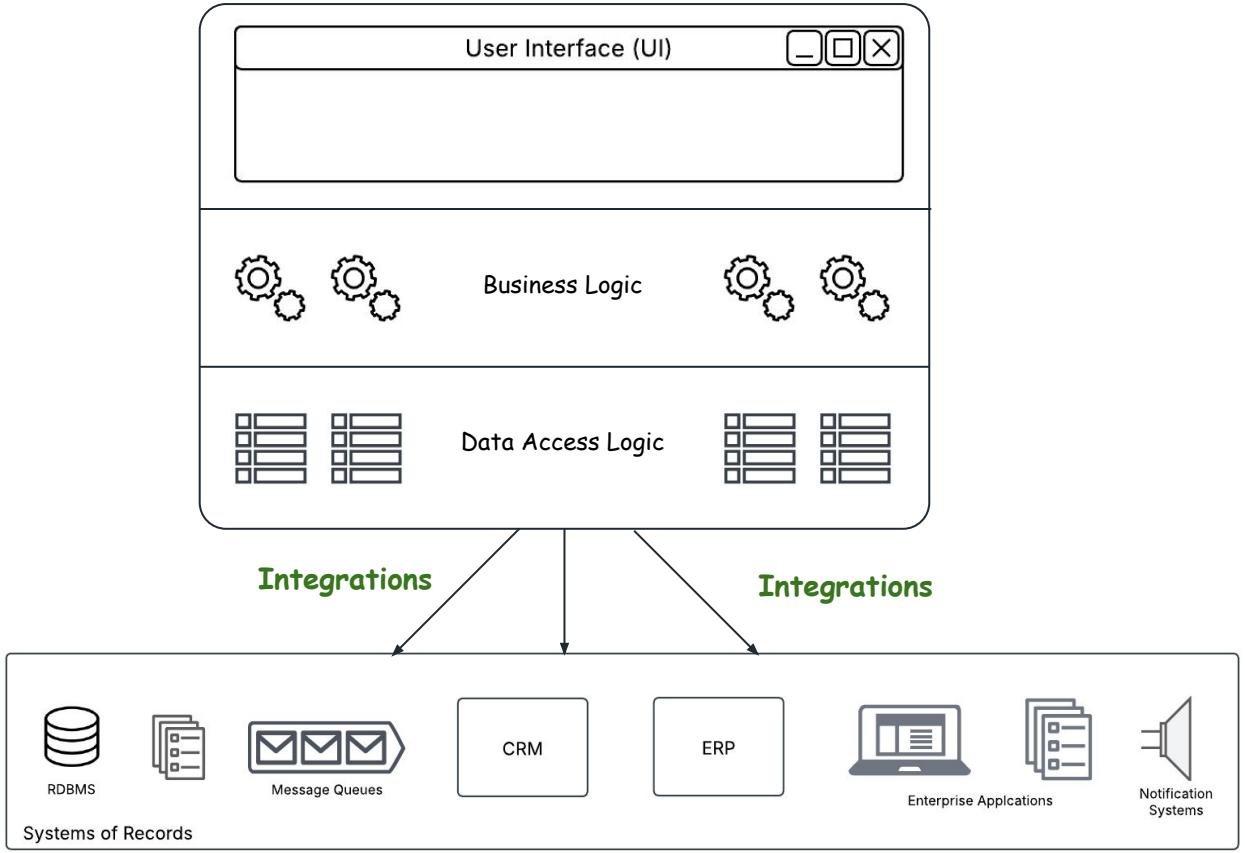


Enterprises create value with digital experiences

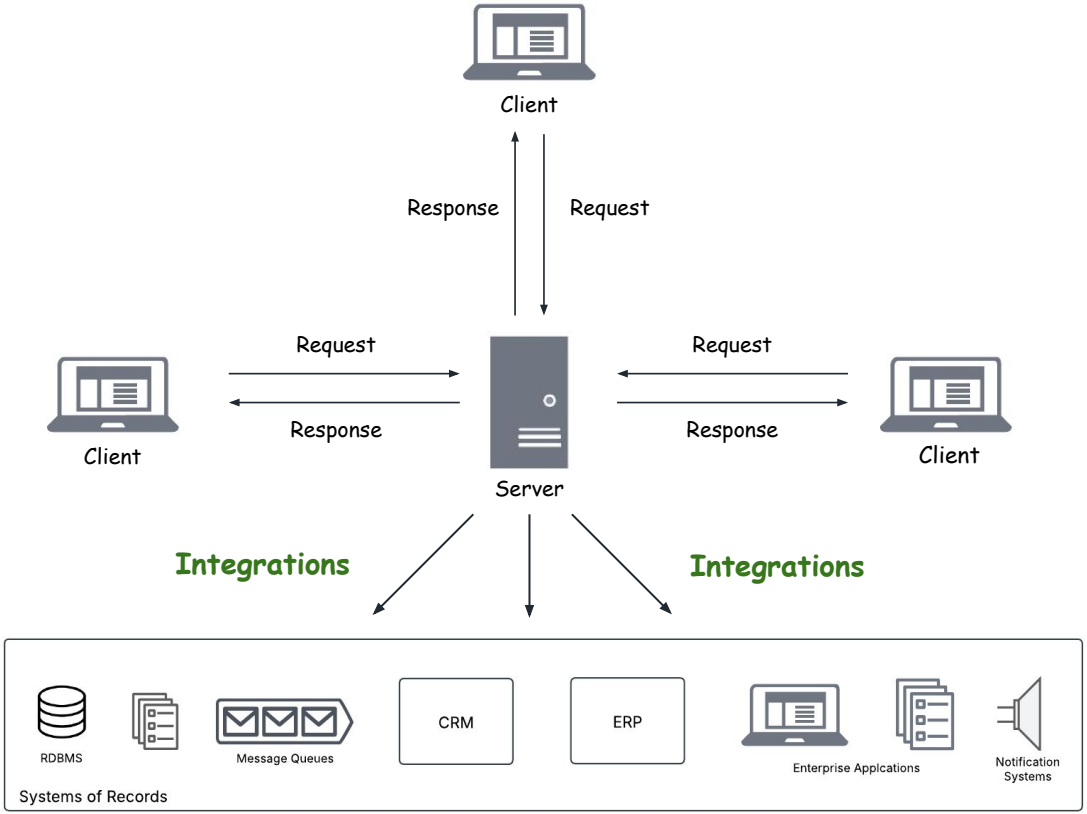
Digital Experience comes with Applications



Monolithic Applications



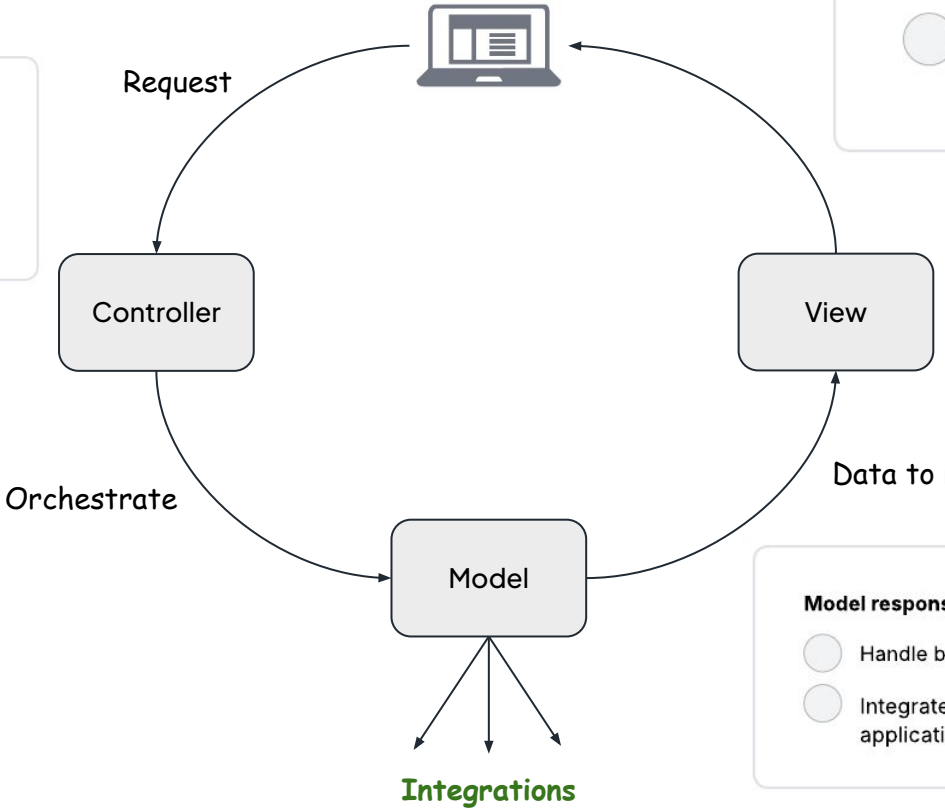
2-tier Client Server Architecture



3-tier Model-View-Controller (MVC)

Controller responsibility

- Instructs the model when to use data
- Instructs the model what to use



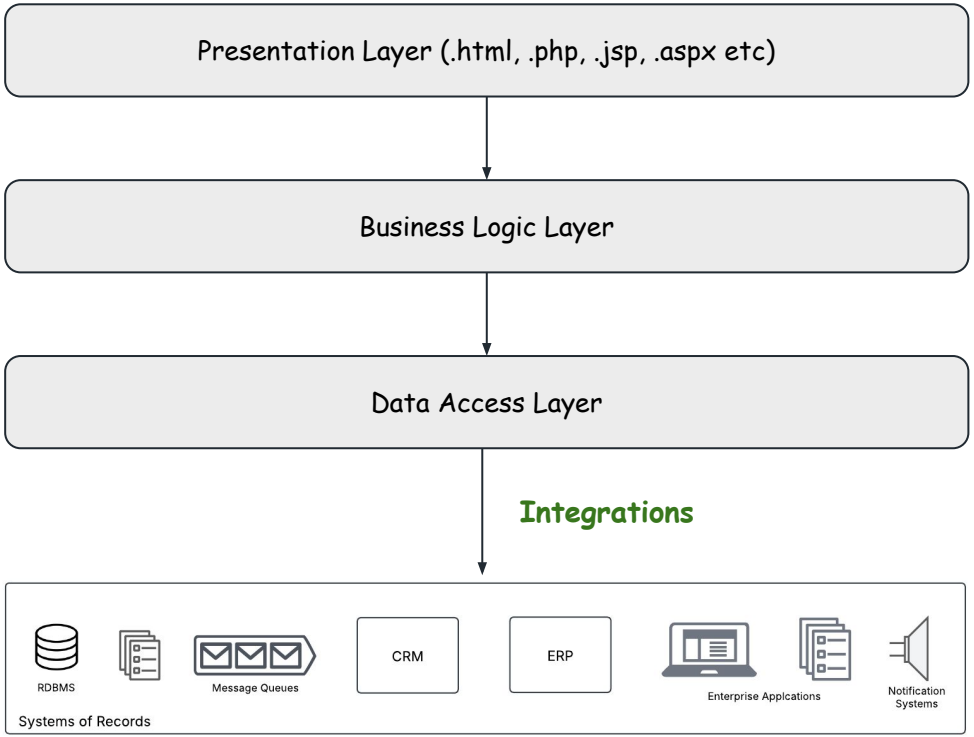
View responsibility

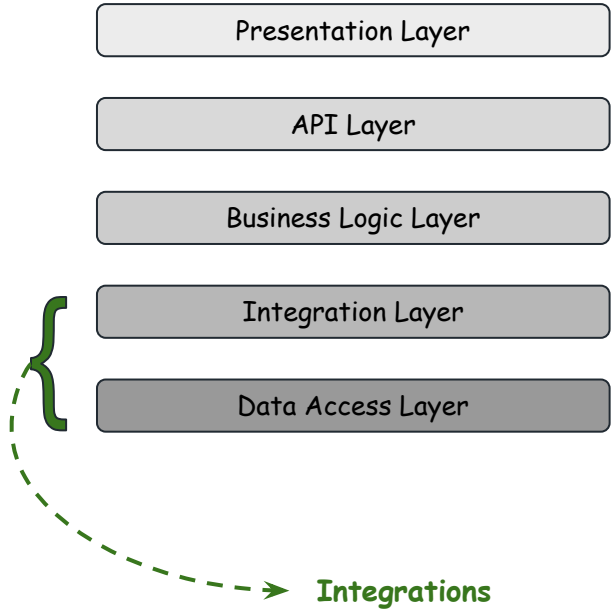
- Take data from model and format as needed for the client

Model responsibility

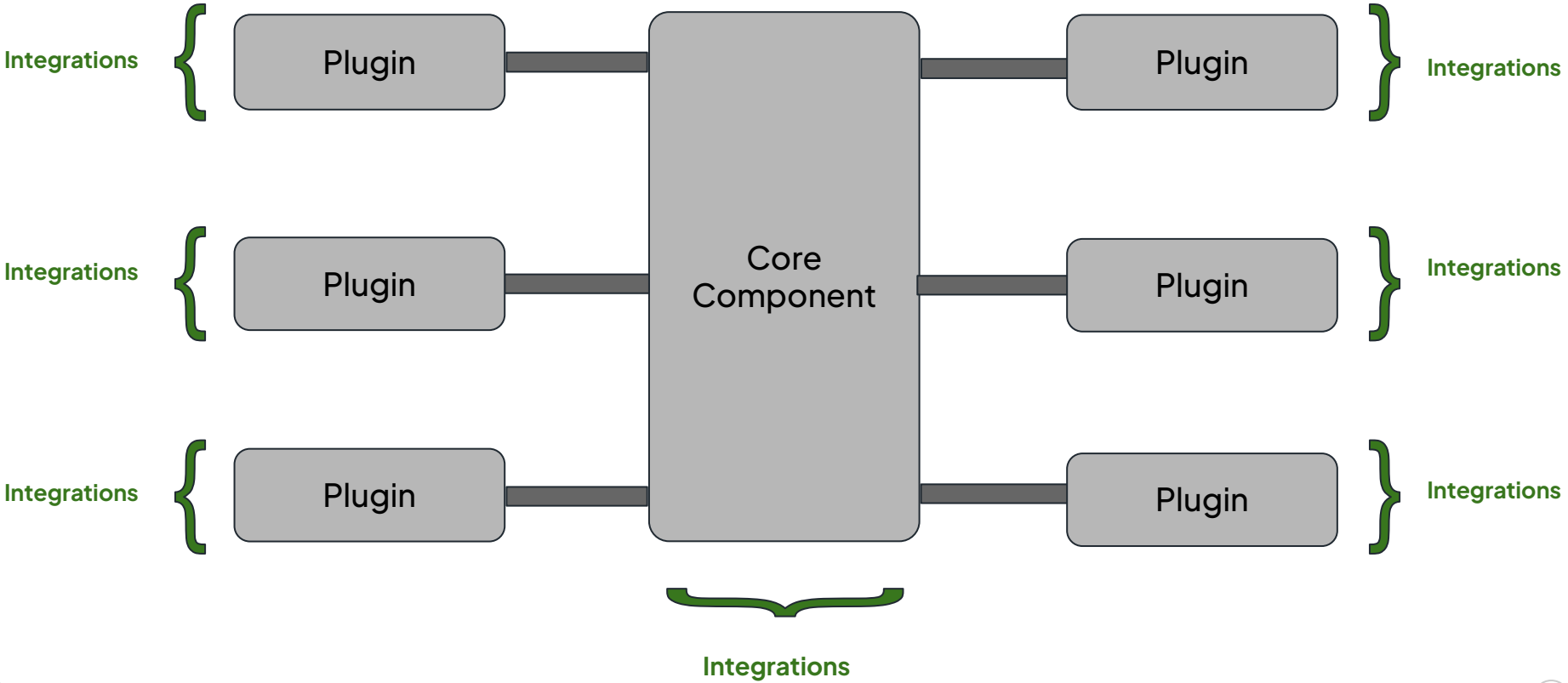
- Handle business logic
- Integrate with downstream applications

N-tier Layered Architecture





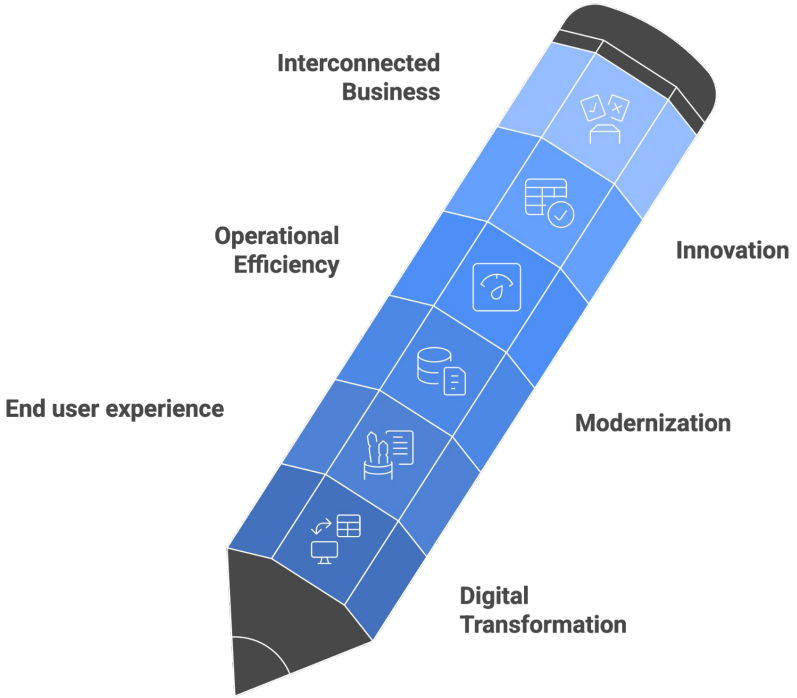
Microkernel Architecture



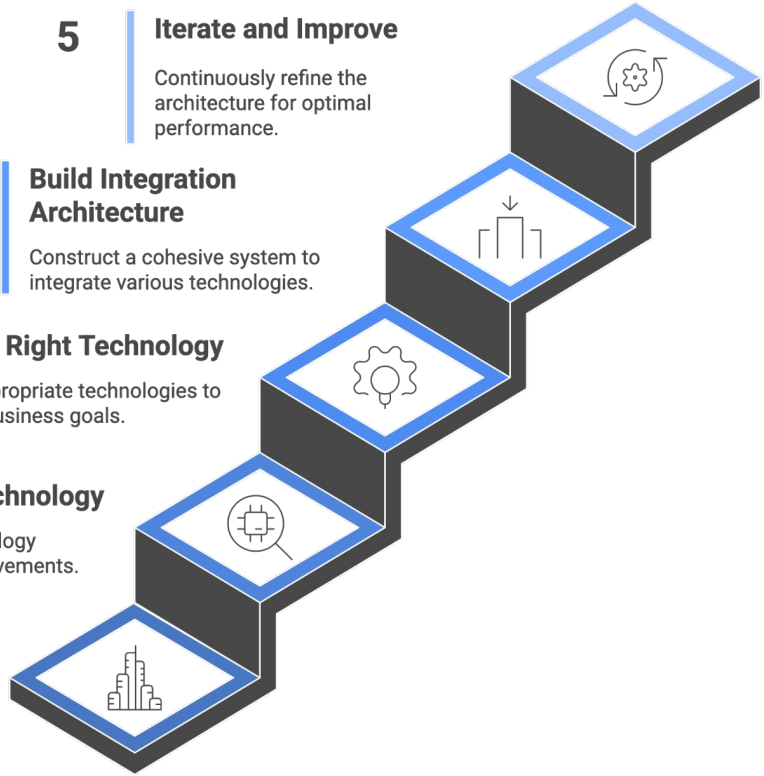
Why do we need Enterprise Integration Architecture (EIA)?

"EIA serves as a fundamental tenet enabling seamless communication between disparate systems"

- To expose shared functionality for all applications within the enterprise
- To enable enterprise as an integrated whole
  - To have a unified integration method



Approach to building an Enterprise Integration Architecture



1 **Create Business Architecture**

Develop a strategic framework for business operations.

2 **Audit Existing Technology**

Evaluate current technology infrastructure for improvements.

3 **Choose Right Technology**

Select appropriate technologies to support business goals.

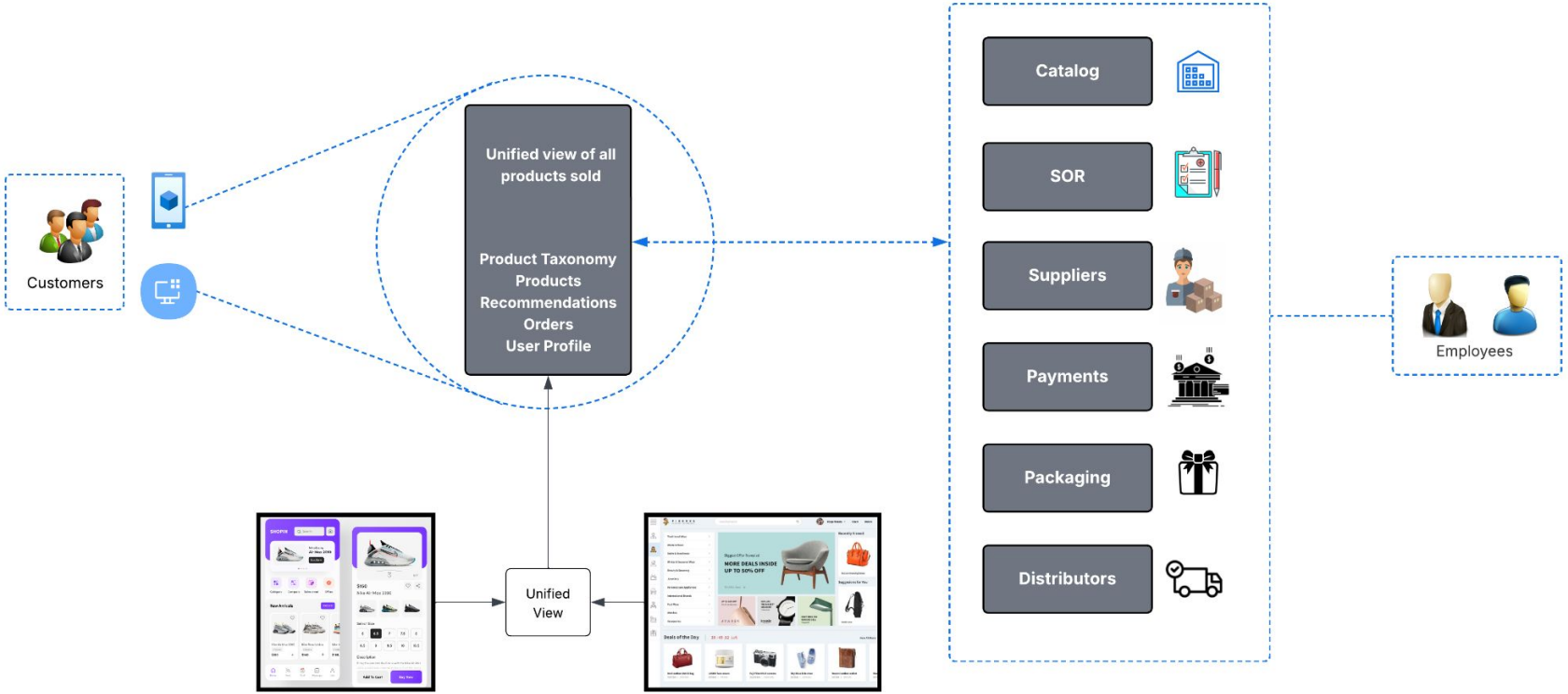
4 **Build Integration Architecture**

Construct a cohesive system to integrate various technologies.

5 **Iterate and Improve**

Continuously refine the architecture for optimal performance.

Business Architecture



Seamless UX to Consumers through multiple channel applications

Interconnected Business + Efficient Operations

Main Pillars of Enterprise Integration

Application Integration

Connects diverse software applications to ensure seamless data flow.

Process Integration

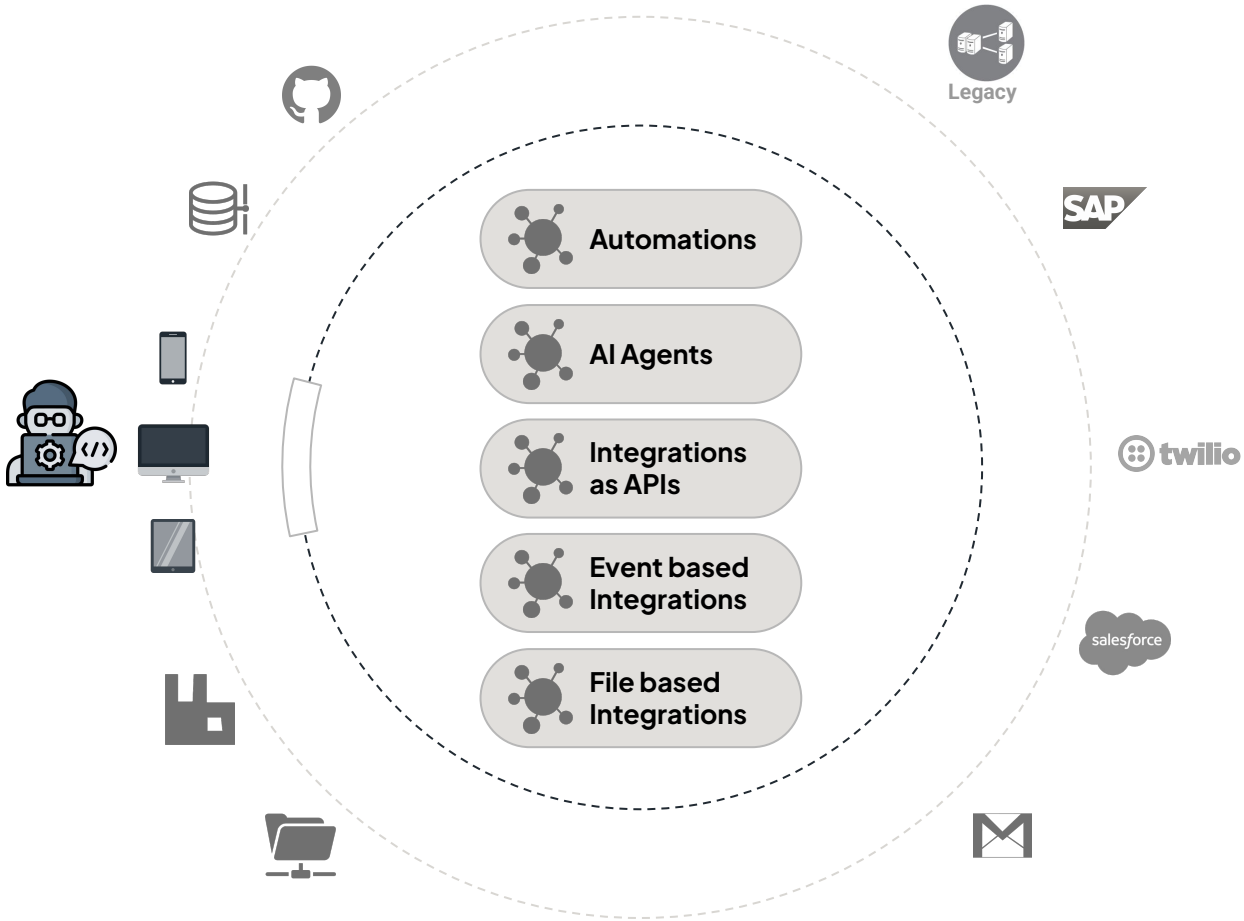
Aligns business processes across departments for efficiency and consistency.



Data Integration

Unifies data from various sources into a cohesive, accessible format.

Common Integration Scenarios in Enterprises



Common integration scenarios

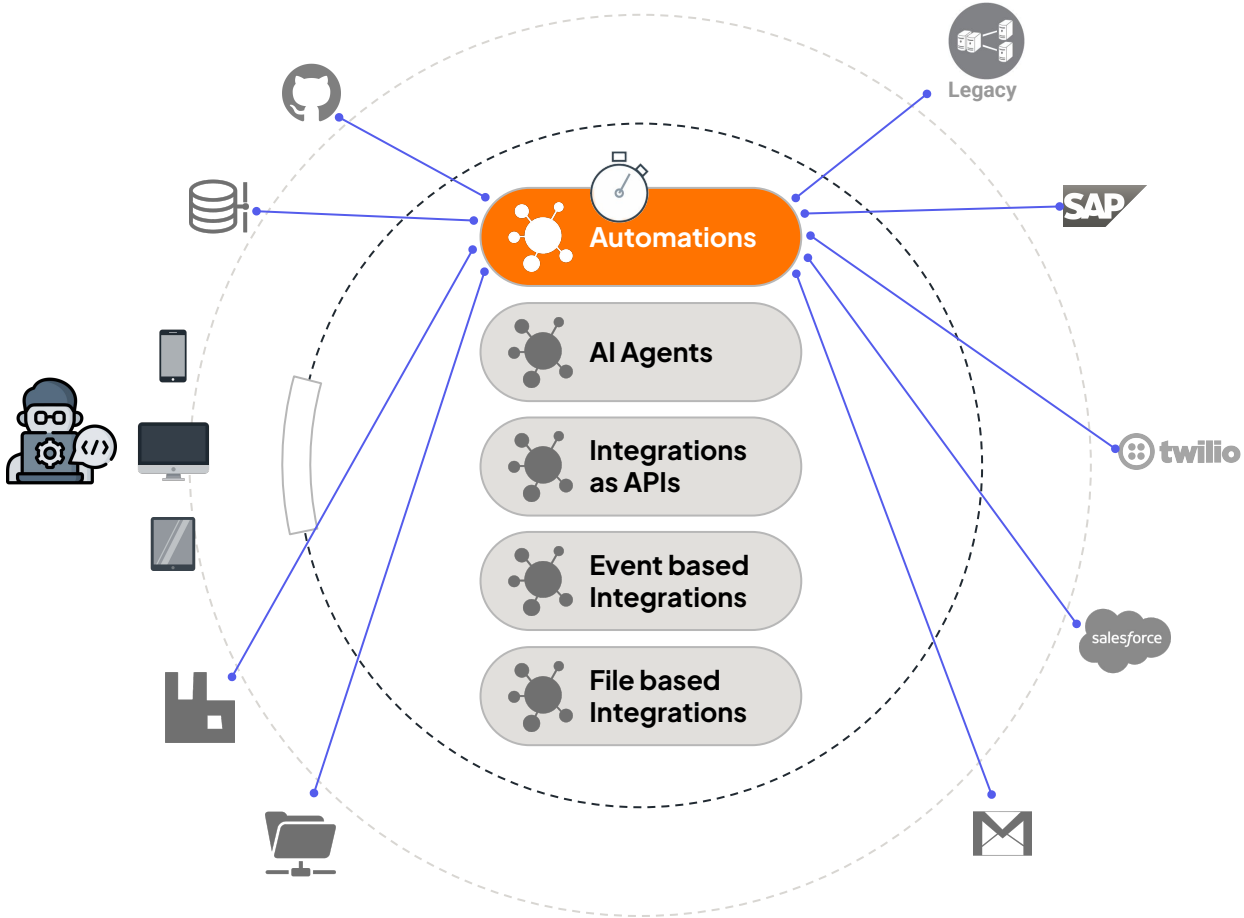
Automations - automatically trigger integrations based on a timer.

AI Agents - reason and plan with GenAI models, leverage available knowledge, and call available tools.

Integrations as APIs - use API requests to trigger integrations in real-time.

Events - trigger integrations based on events.

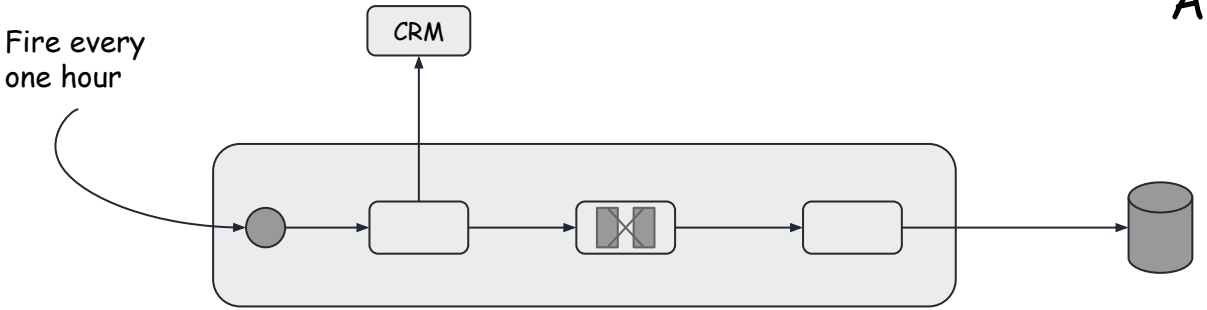
Files - trigger integrations based on the availability of files in a location.

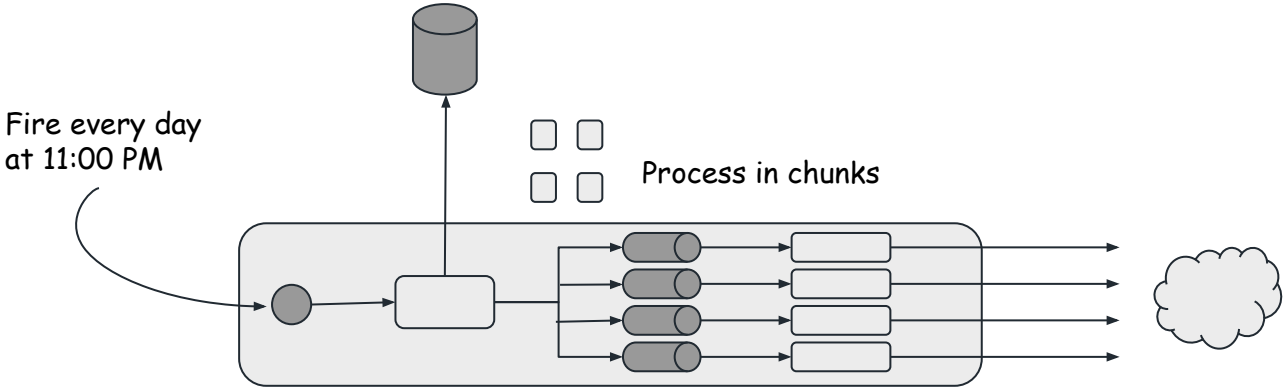


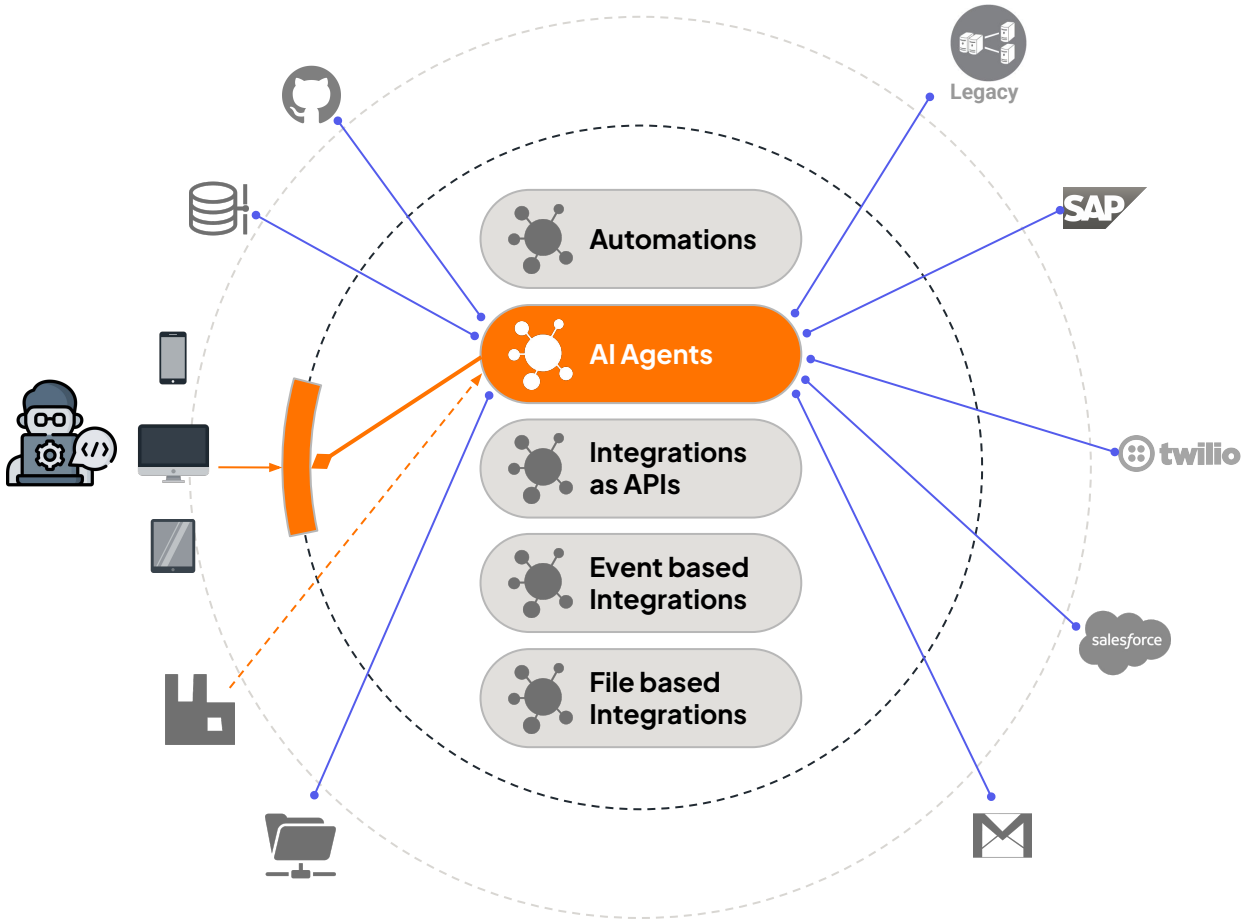
Use cases

- Periodic data synchronization
- Batch tasks / business processes
- Extract, Transform, Load (ETL)

Automations



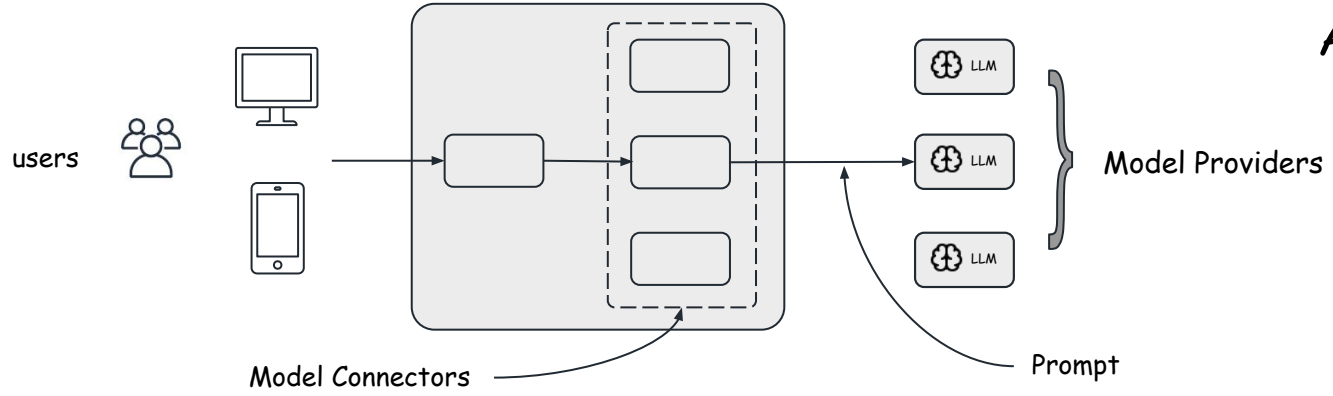


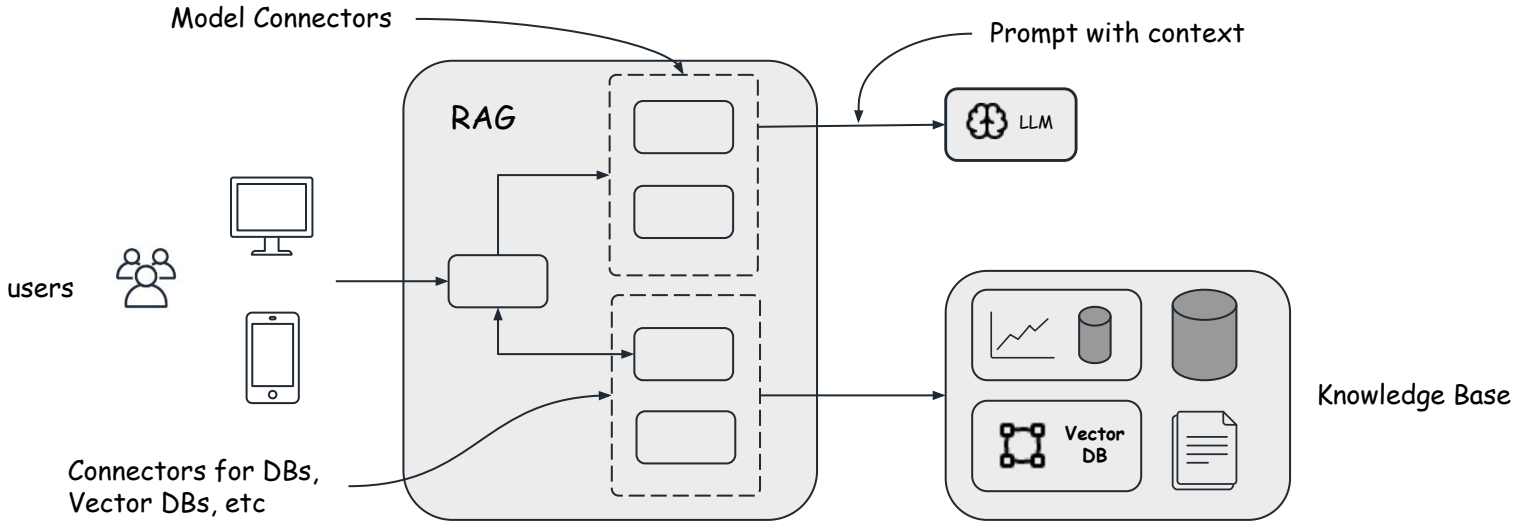


Use cases

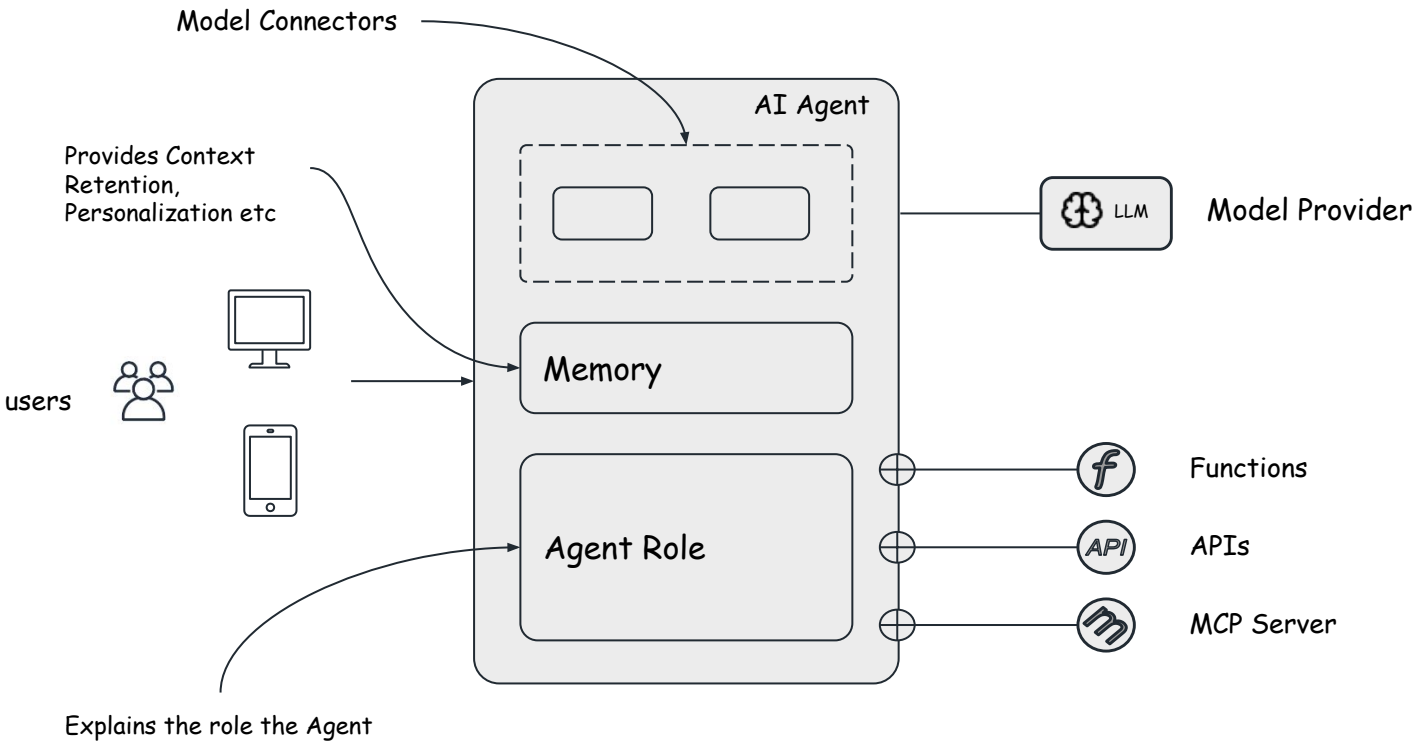
- Virtual assistants / chatbots
- Actions based on sentiment analysis
- Error detection, and resolution

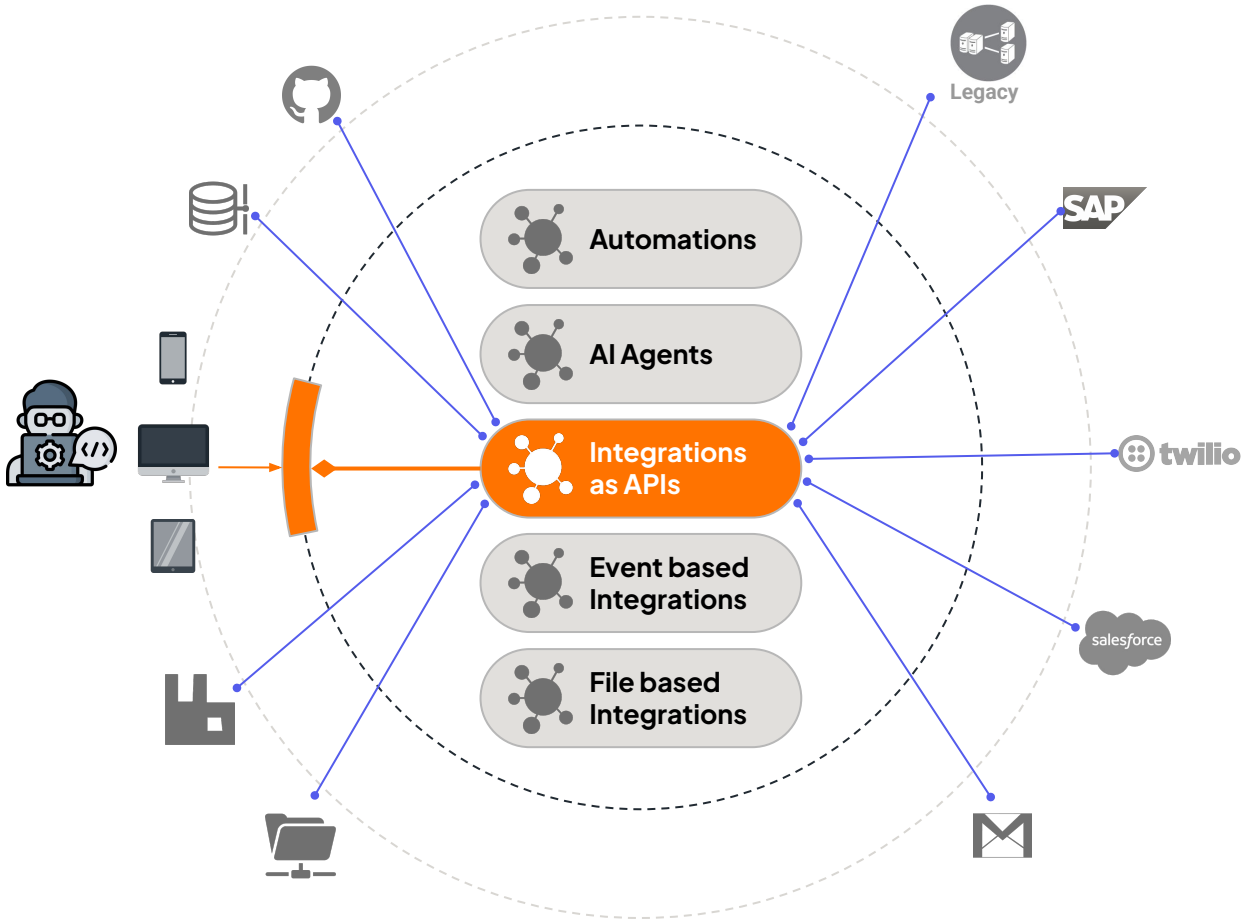
AI Integrations





AI Agents

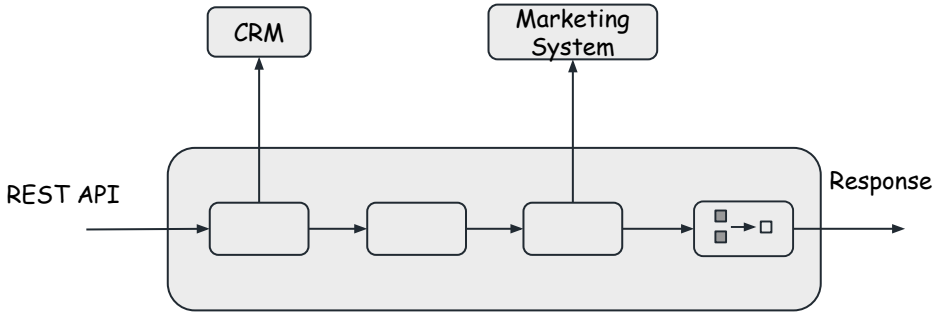

Model Connectors
AI Agent
Provides Context Retention, Personalization etc
LLM
Model Provider
users
Memory
Agent Role
Functions
APIs
MCP Server
Explains the role the Agent

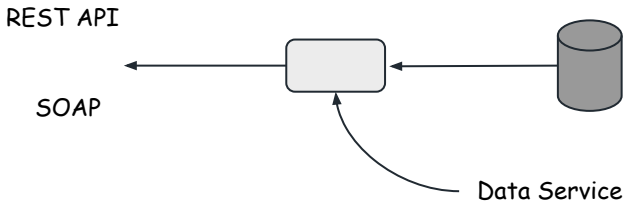


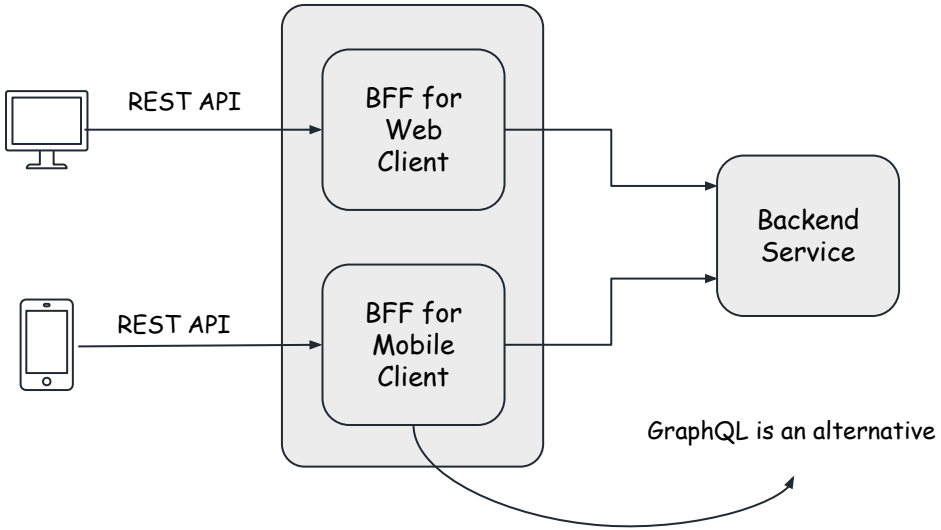
Use cases

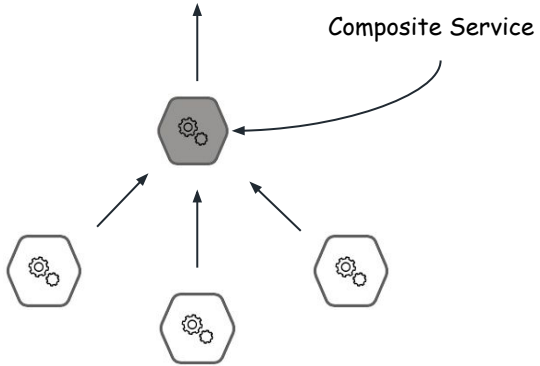
- Multi-step processes
- Composite services
- Backend for frontend (BFF)
- Data services

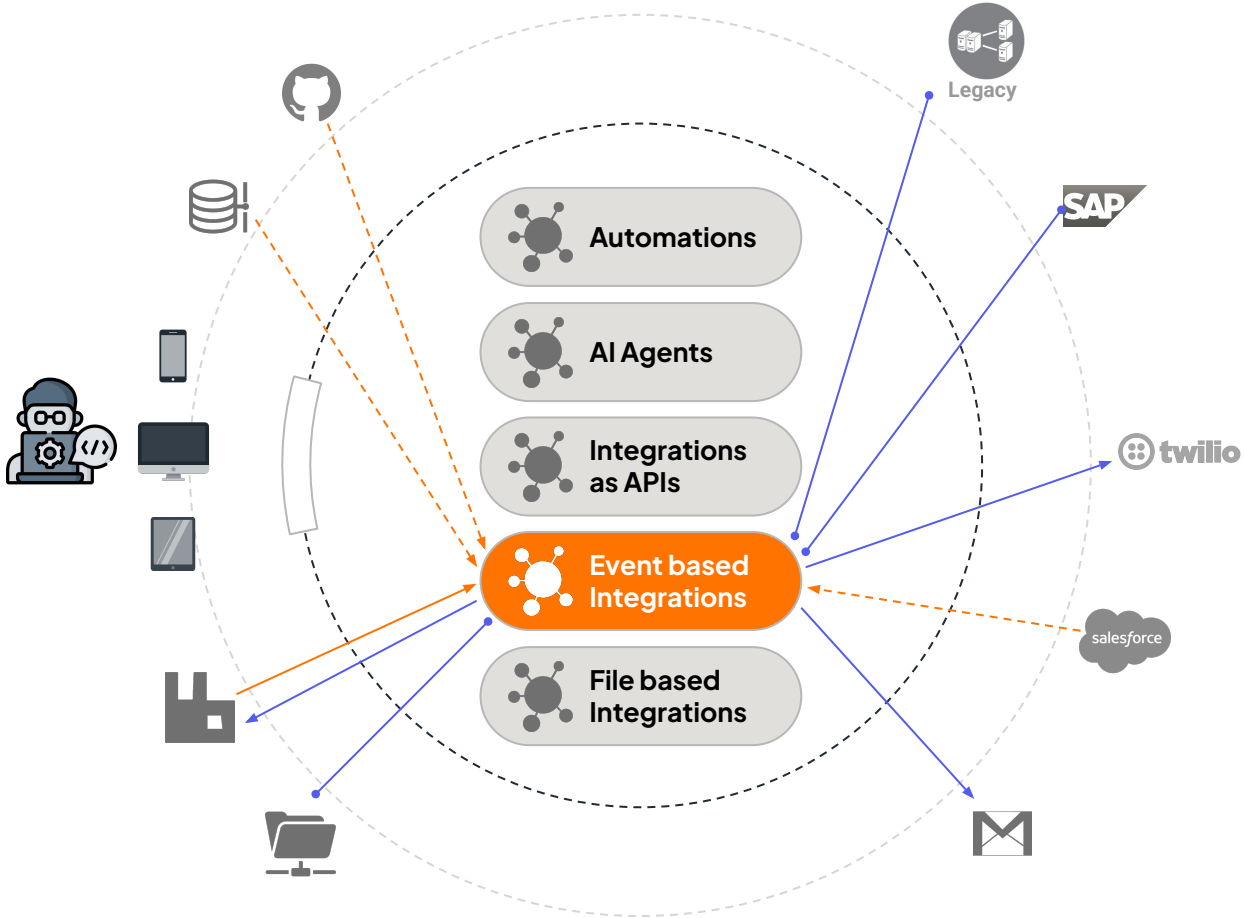
Integrations as APIs







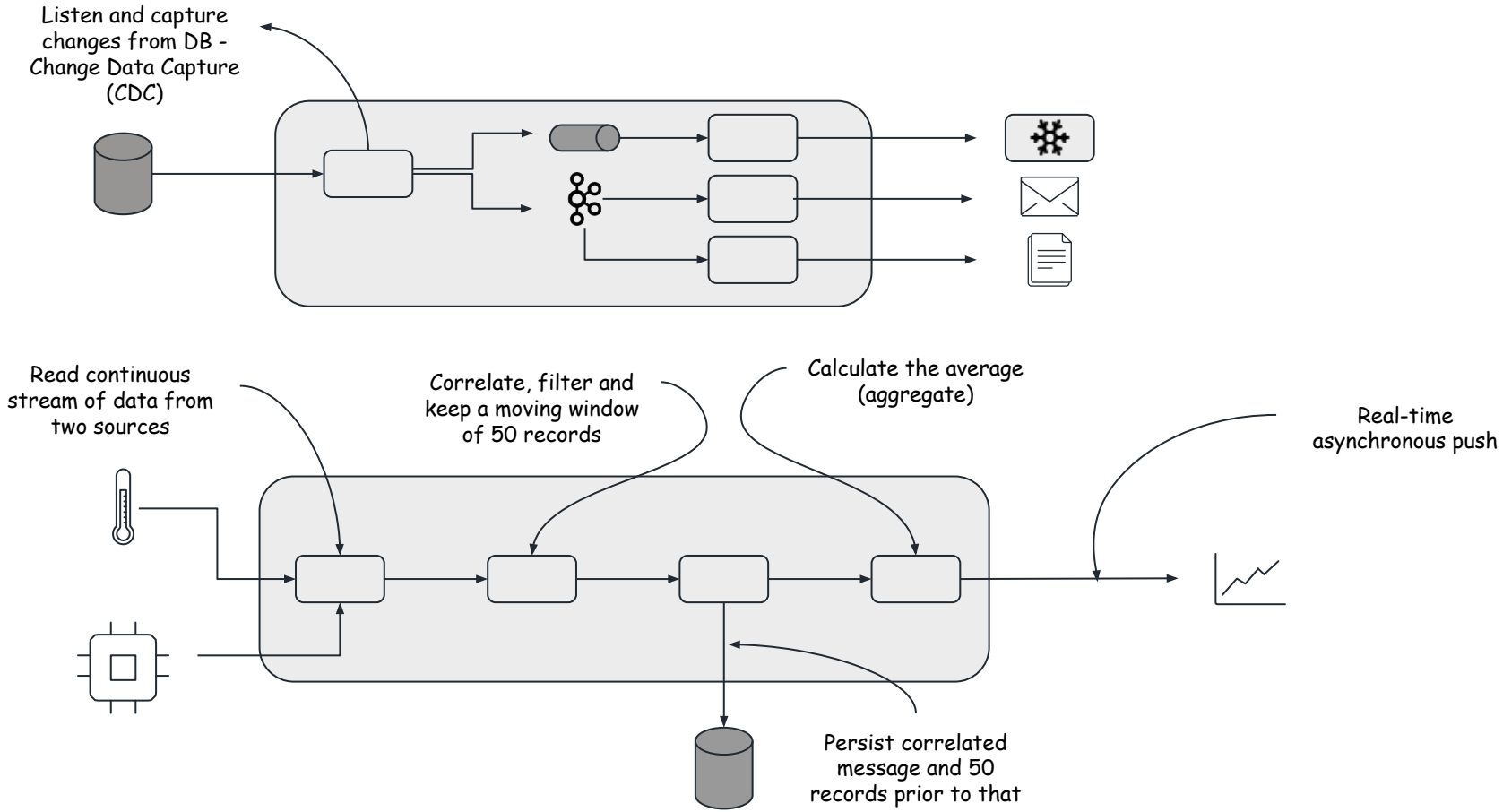


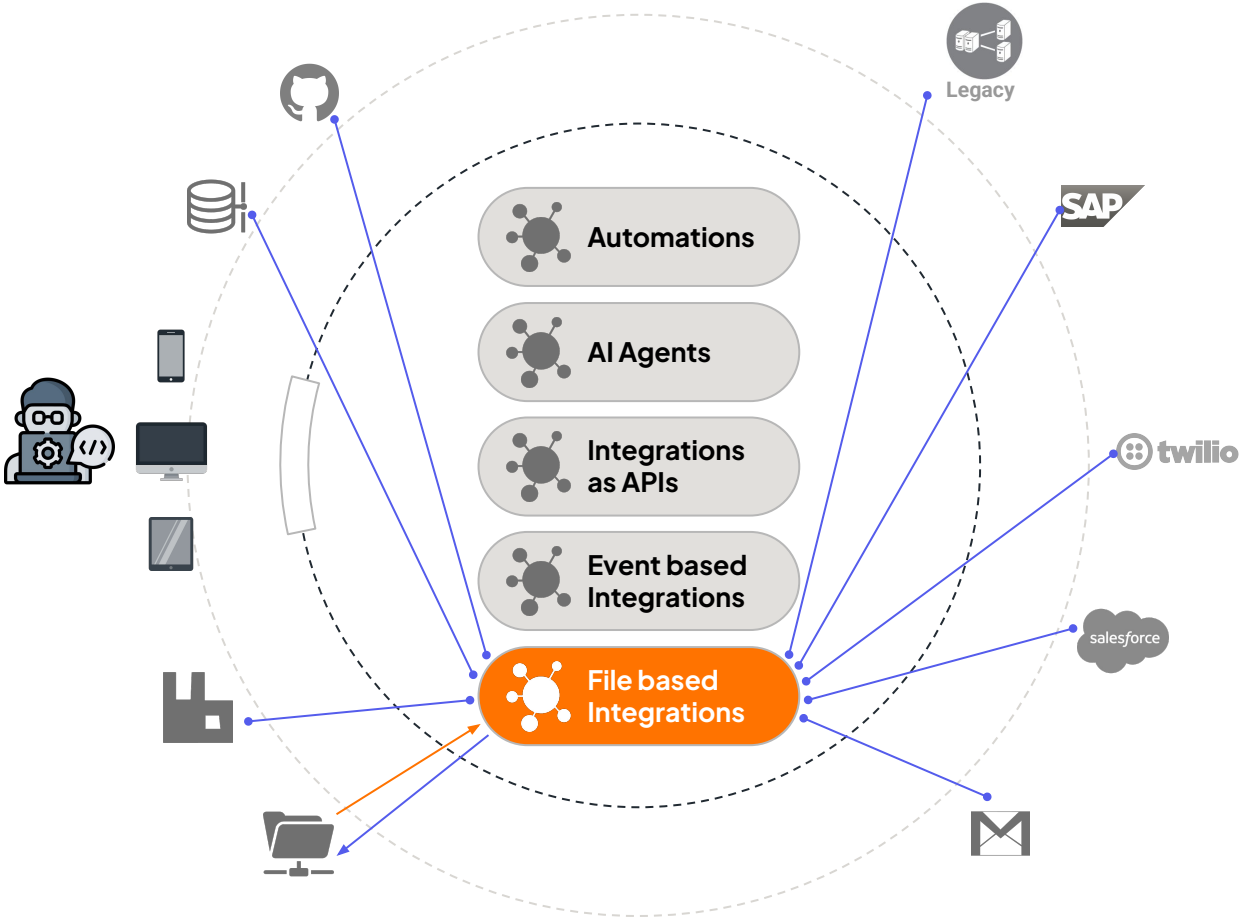


Use cases

- Data consistency by propagating state changes
- Trigger business processes based on state changes
- Change data capture (CDC)
- Streaming ETL
- Stream processing
- Complex event processing

Event-based integrations

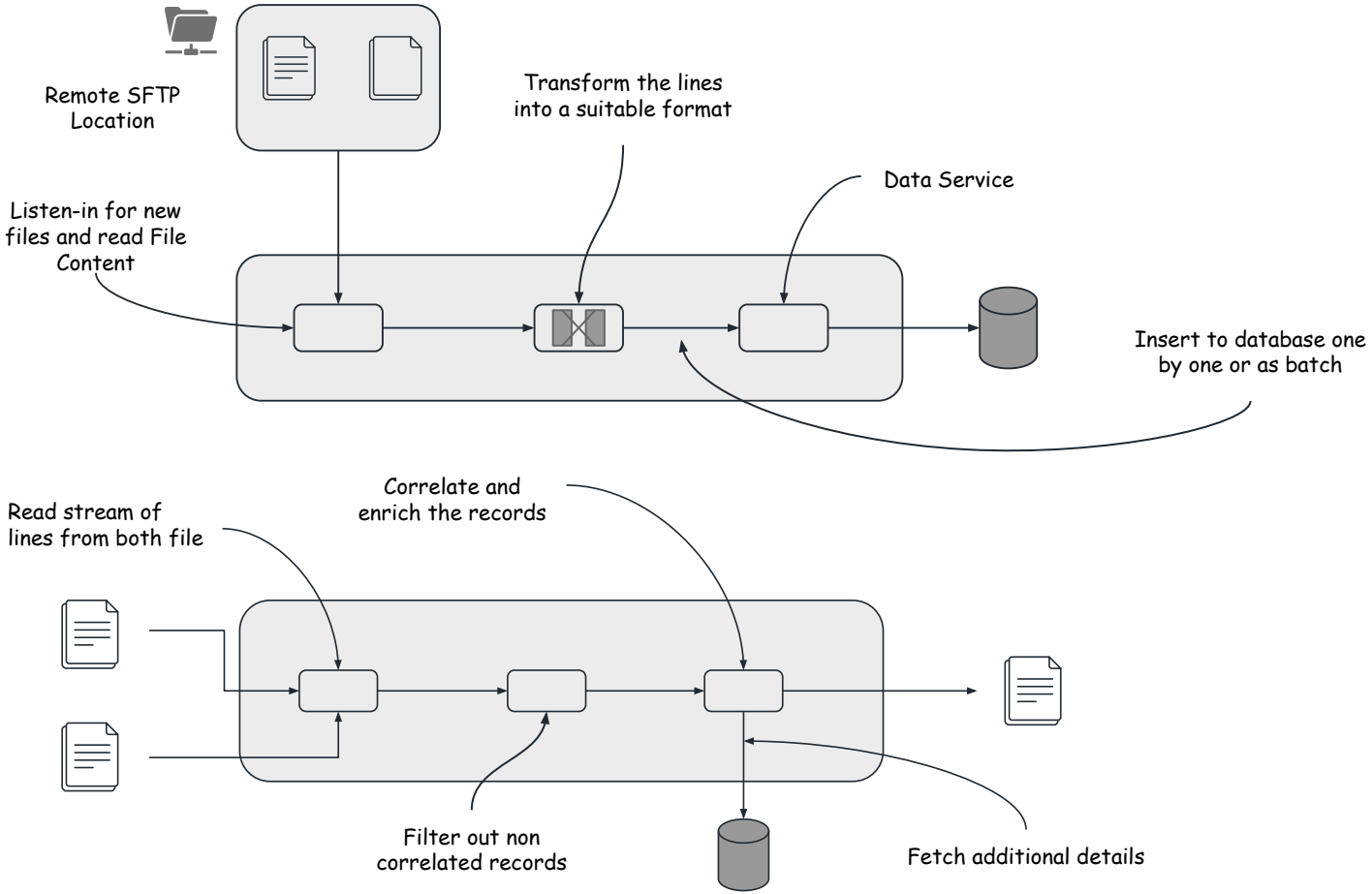




Use cases

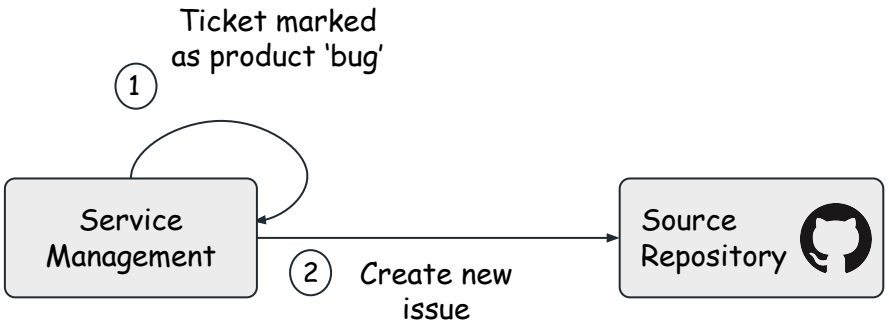
- Extract, Transform, Load (ETL)
- Trigger business processes for each entry in a file
- B2B integrations
- Legacy system integrations

File-based integrations



Integration Architectures/Topologies

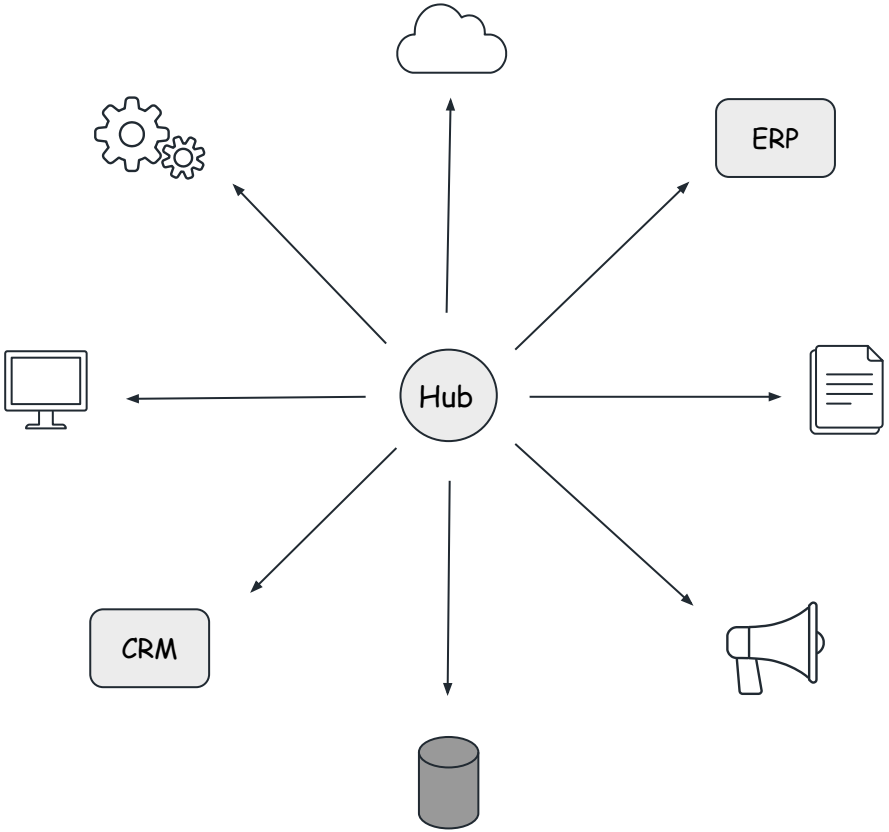
Point-to-Point Integration



Also known as "P2P Integration" and "Native Integration"

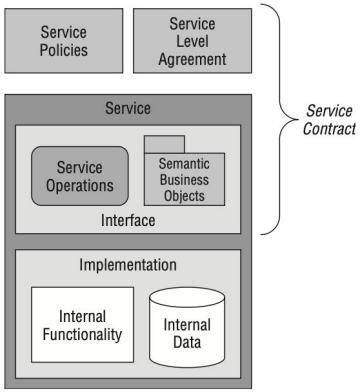
Pros
Simplicity
Faster to implement, deploy and run
Less dependent on third party systems
Cons
Tight coupling
Difficult to scale
Difficult to maintain in long term
Poor visibility and governance
Security risk for systems

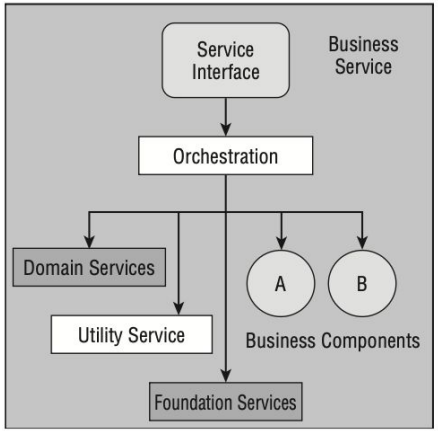
Hub and Spoke Architecture



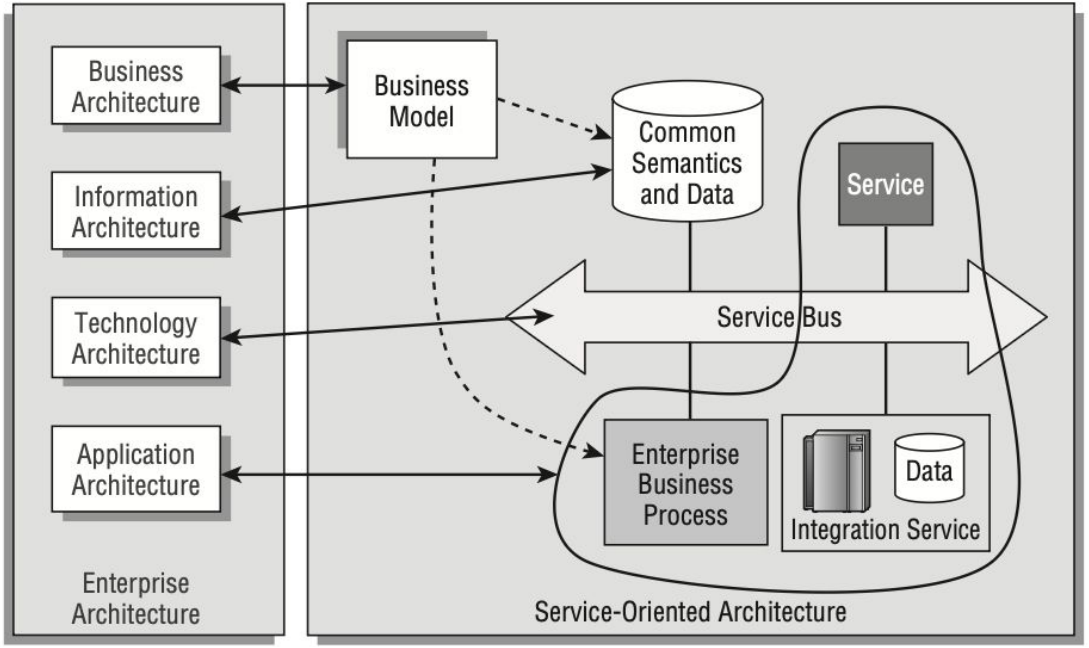
Pros
Avoids implementing logic in all systems
Decouples systems of records
Integrations managed by central hub
Cons
Difficult to scale
Single point of failure
Not cloud native

Service Oriented Architecture (SOA)



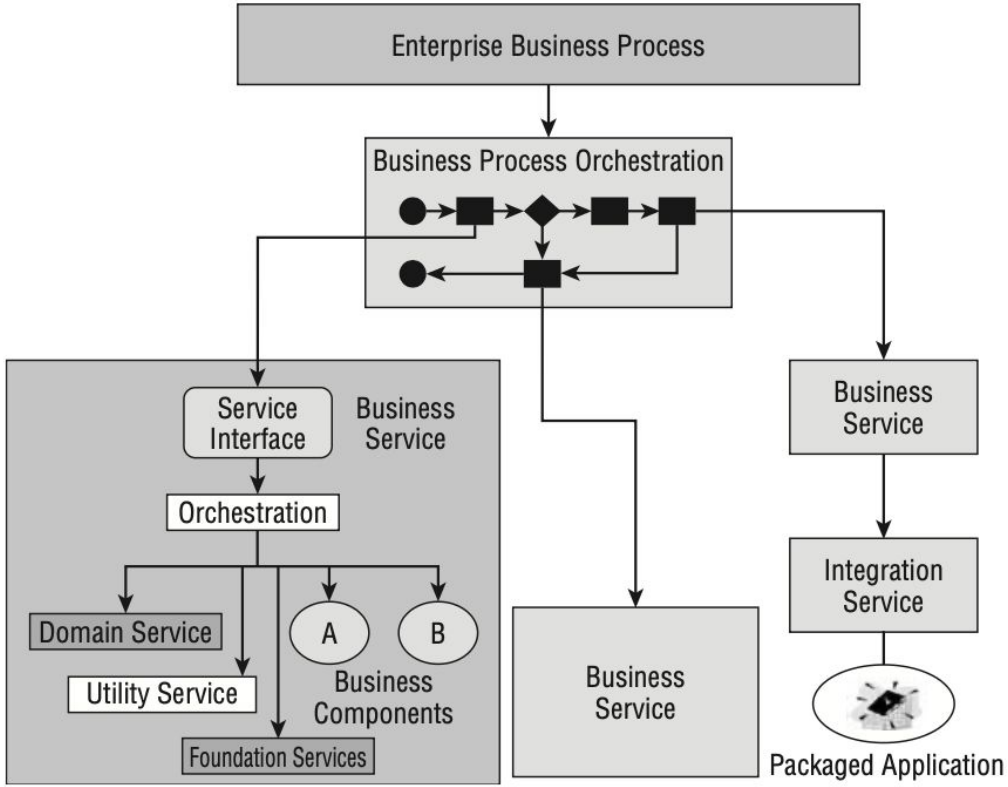


Breakdown application and integration logic to multiple reusable and composable services



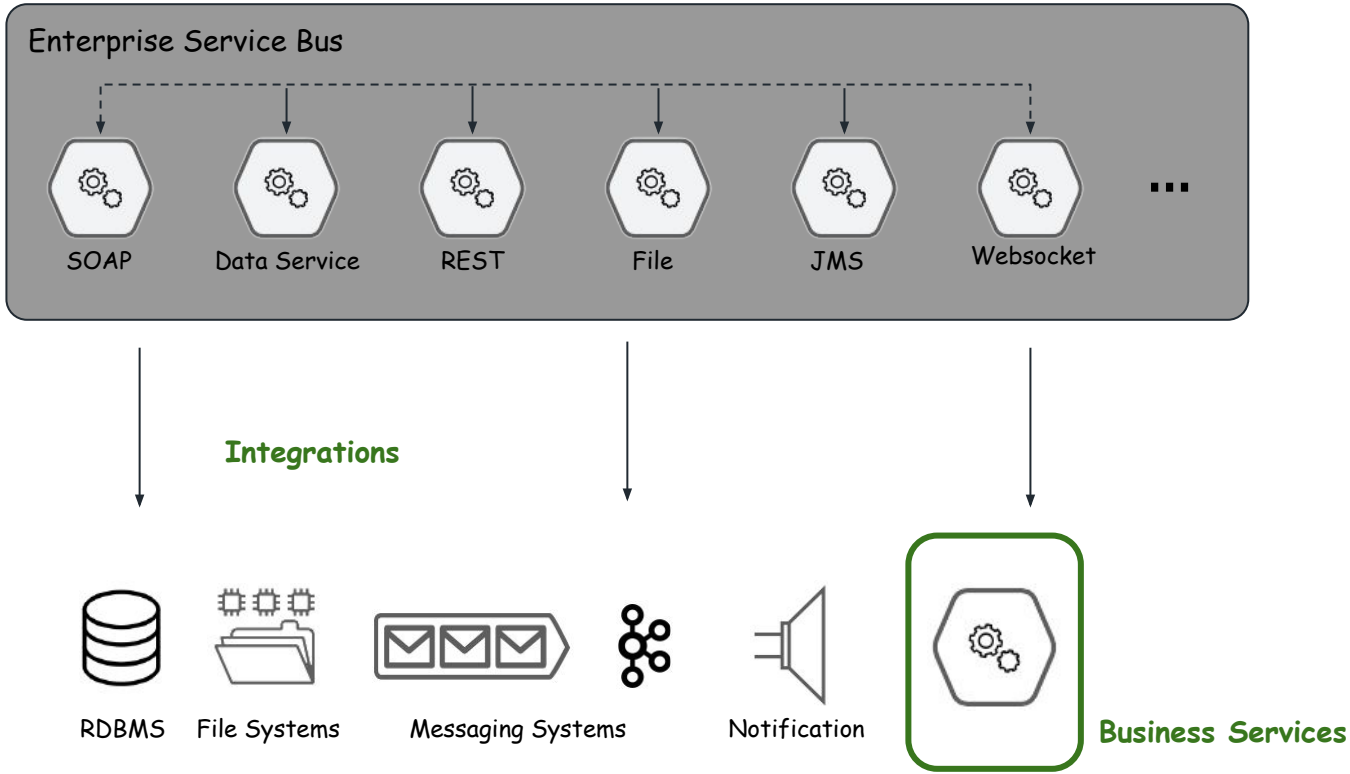
Reference - Applied SOA: Service-Oriented Architecture and Design Strategies by Michael Rosen et al.

Service Oriented Architecture (SOA)



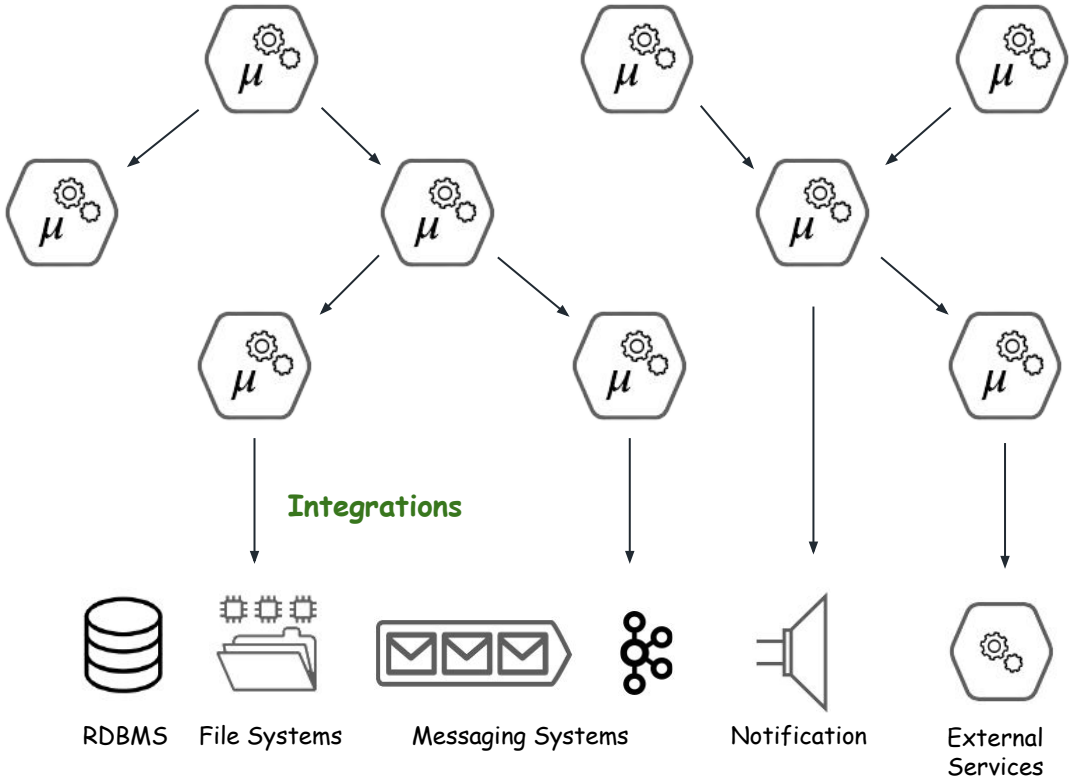
Reference - Applied SOA: Service-Oriented Architecture and Design Strategies by Michael Rosen et al.

Service Oriented Architecture (SOA)



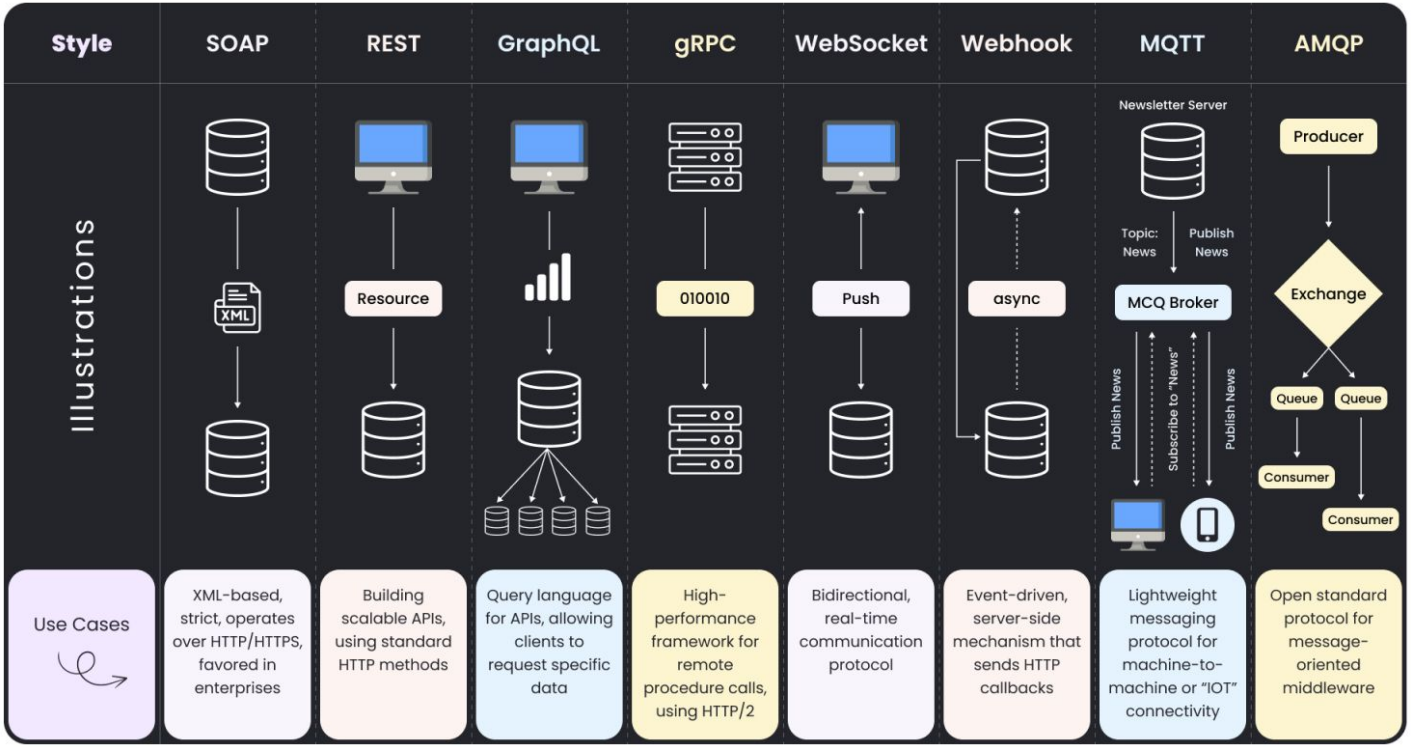
- Traditionally works along with ESBs
- Services are composable
- ESB orchestrates business services and provides wide protocol support for integration
- Provides plethora of connectors to heterogeneous systems

Microservices Architecture (MSA)



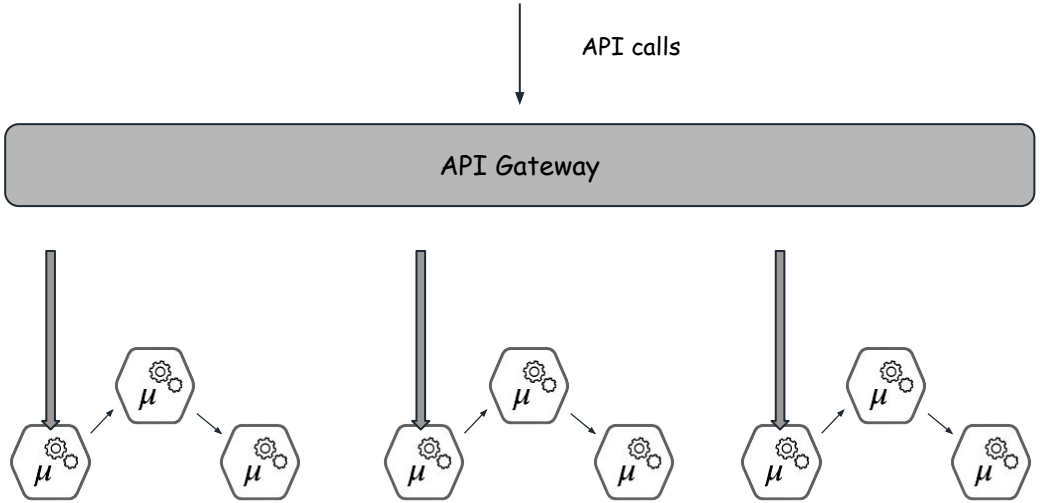
- An evolution of SOA
- A service is a small, individual, independently deployable unit that does one unit of work
- An application is a collection of such small loosely coupled services
- Some services can be integration focused
- Some business services can do integration as well

API Architectural Styles

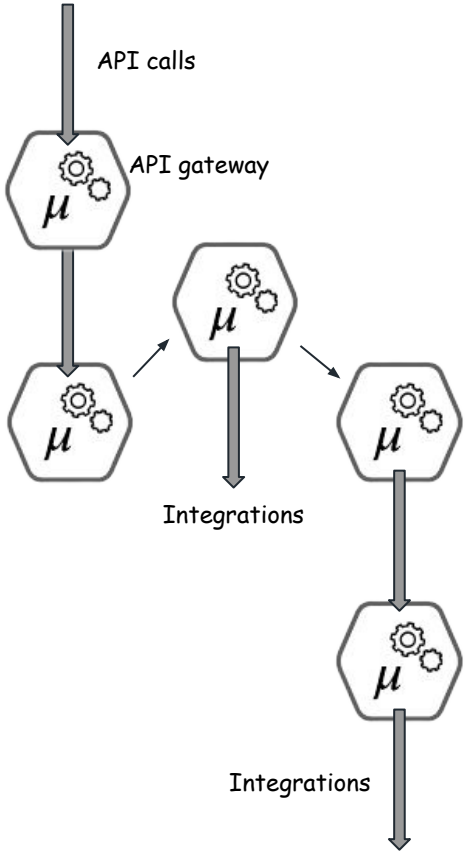


https://www.linkedin.com/pulse/navigating-api-landscape-top-8-architectural-styles-2023-patel/

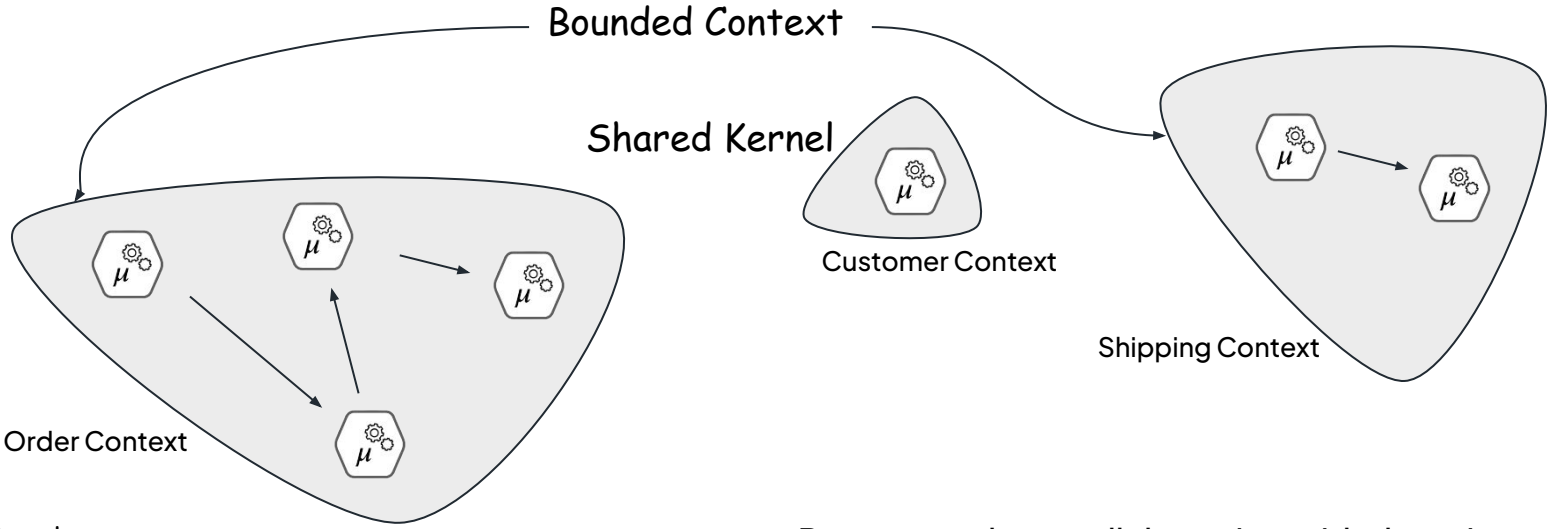
Gateway Architecture Pattern

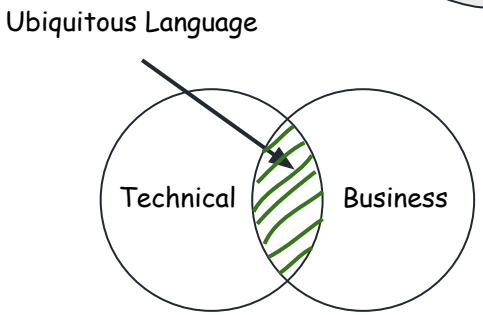


- API Authorization
- Protocol Translation
- Message Transformation
- Rate limiting/Throttling
- Caching



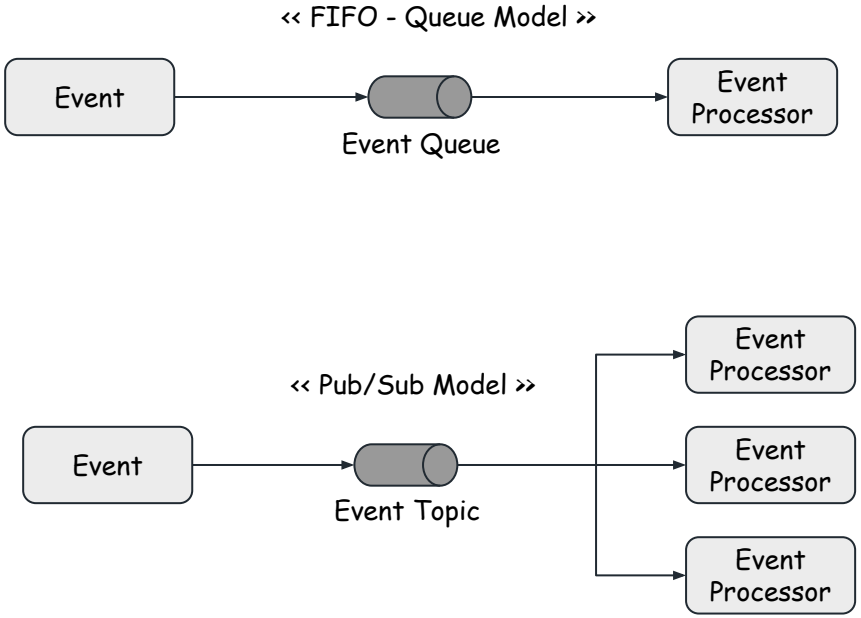
Domain Driven Design (DDD)





- Promotes close collaboration with domain experts
- Teams can prioritize business critical integrations
- Modularity
- Flexibility and scalability

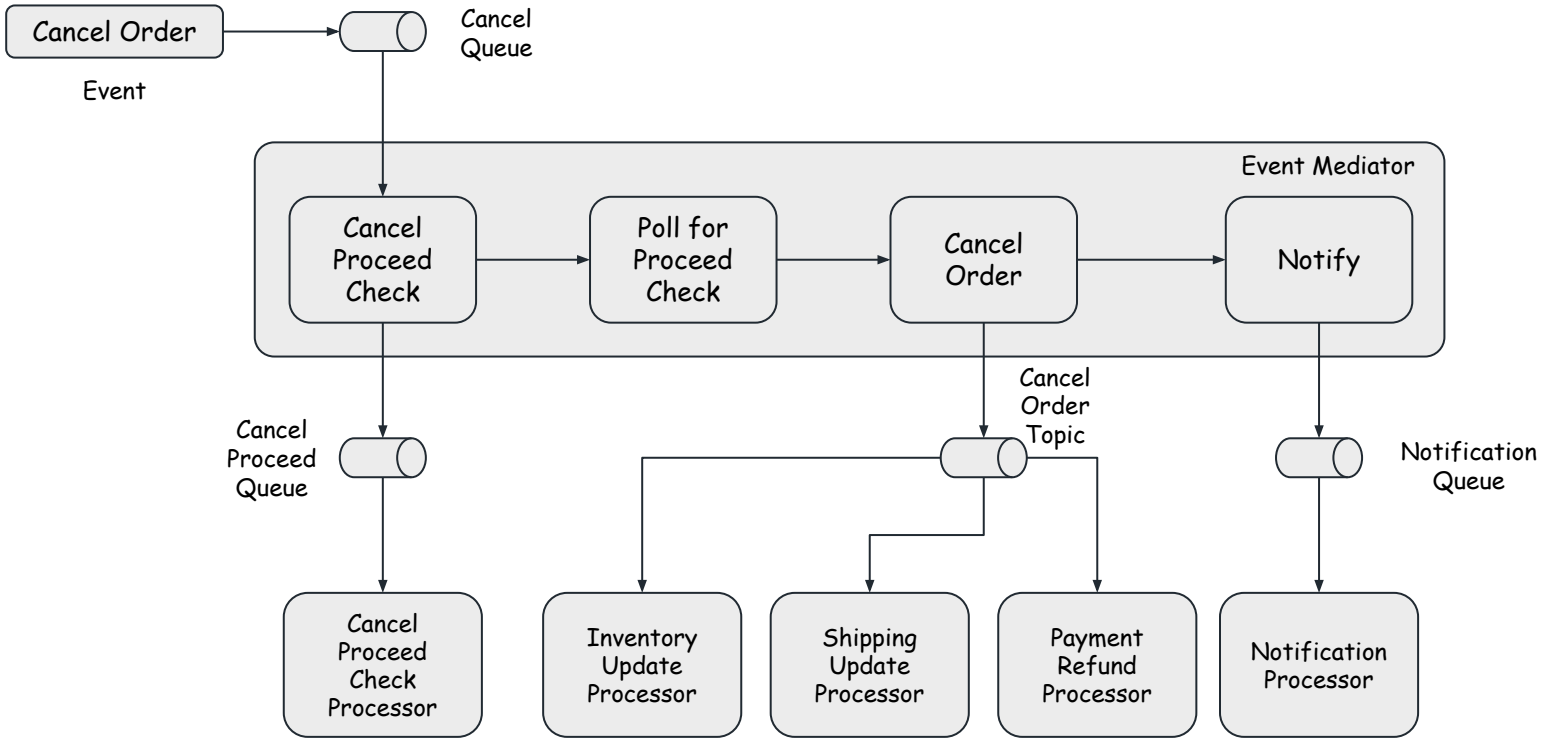
Event Driven Architecture (EDA)



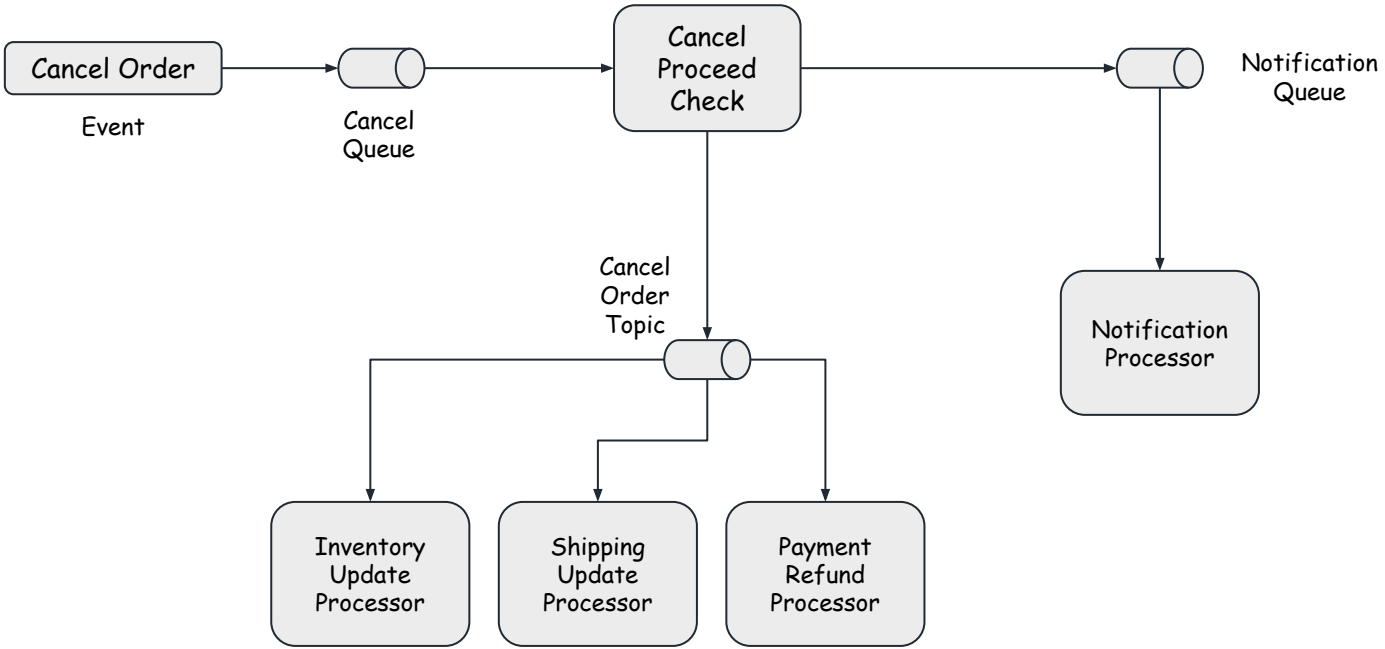
Message Brokers play a key part in this architecture

Pros
Enables asynchronous communication
Suitable for high-load use-cases
Dynamic (in case of pub/sub model)
Highly available and scalable
Cons
Difficult to handle sequencing
Storage and backlog
Challenging to handle transactions

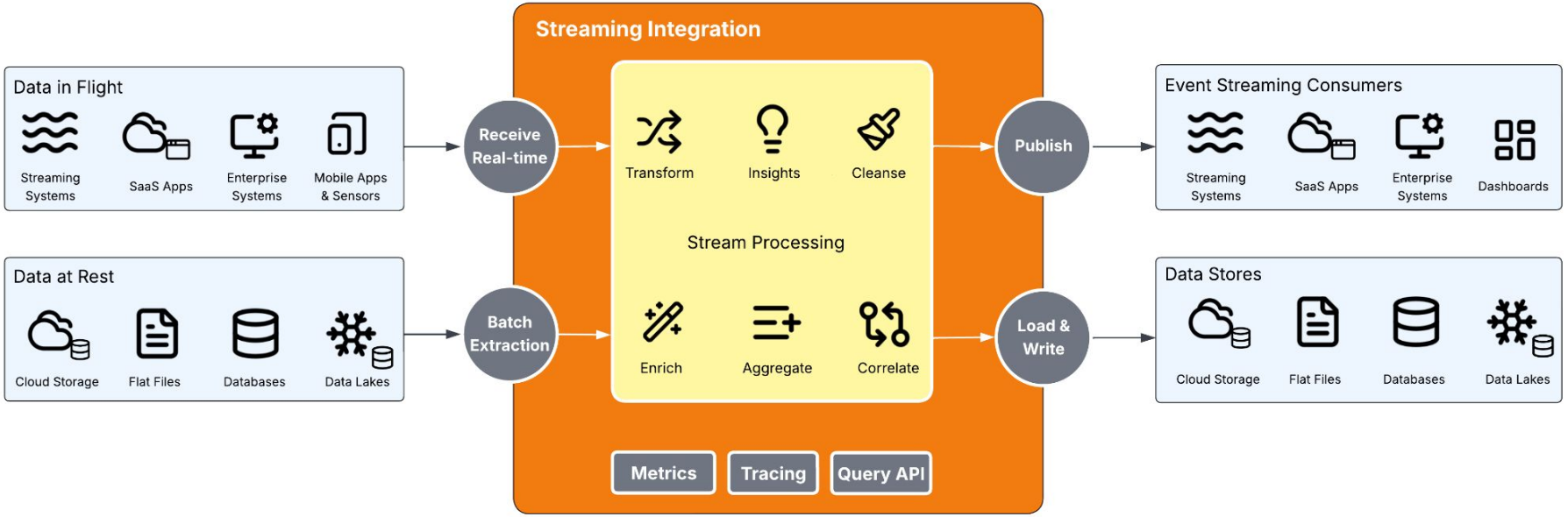
EDA – Mediator Topology



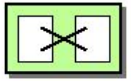
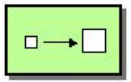
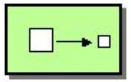
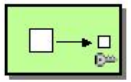
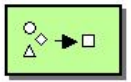
EDA – Broker Topology



Streaming Architecture

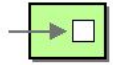
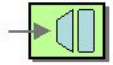
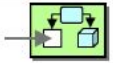
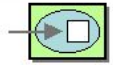
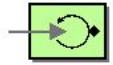
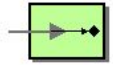
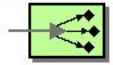
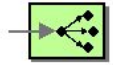
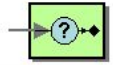
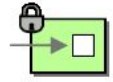


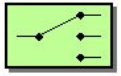
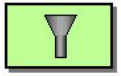
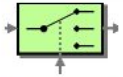
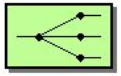
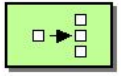
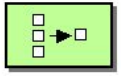
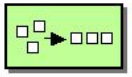
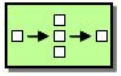
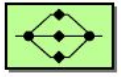
Integration Patterns

Message Transformation	
	Introduction to Message Transformation
	Envelope Wrapper
	Content Enricher
	Content Filter
	Claim Check
	Normalizer

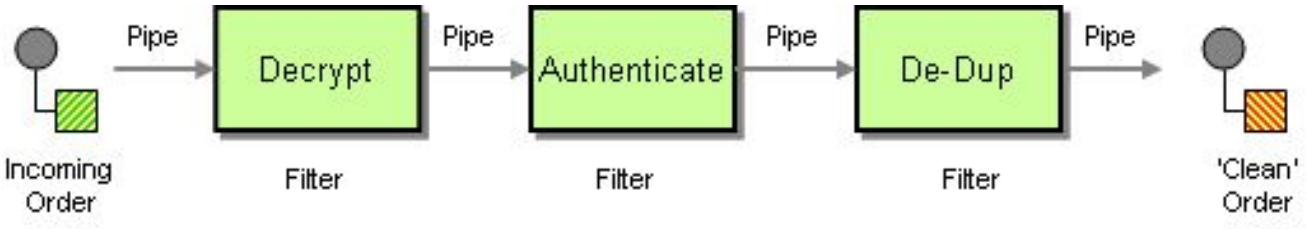
Commonly accepted design solutions for recurring integration problems

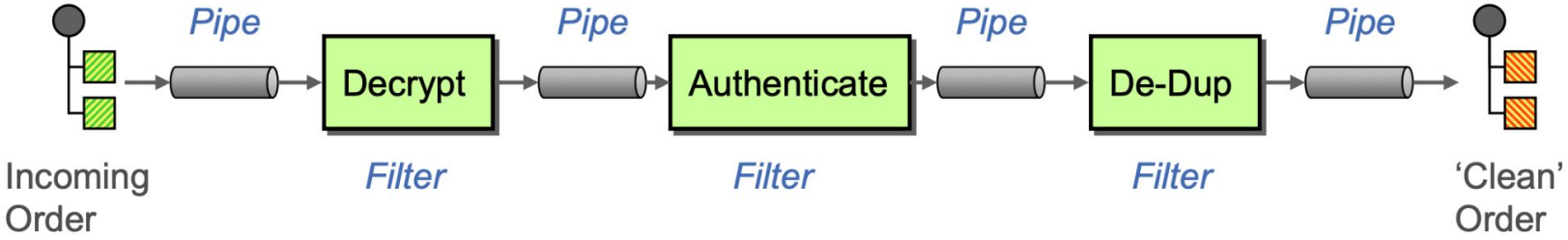
Reference - https://www.enterpriseintegrationpatterns.com/patterns/messaging/toc.html

Messaging Endpoints	
	Introduction to Messaging Endpoints
	Messaging Gateway
	Messaging Mapper
	Transactional Client
	Polling Consumer
	Event-Driven Consumer
	Competing Consumers
	Message Dispatcher
	Selective Consumer
	Durable Subscriber

Message Routing	
	Introduction to Message Routing
	Content-Based Router
	Message Filter
	Dynamic Router
	Recipient List
	Splitter
	Aggregator
	Resequencer
	Composed Message Processor
	Scatter-Gather

Pipes and Filters





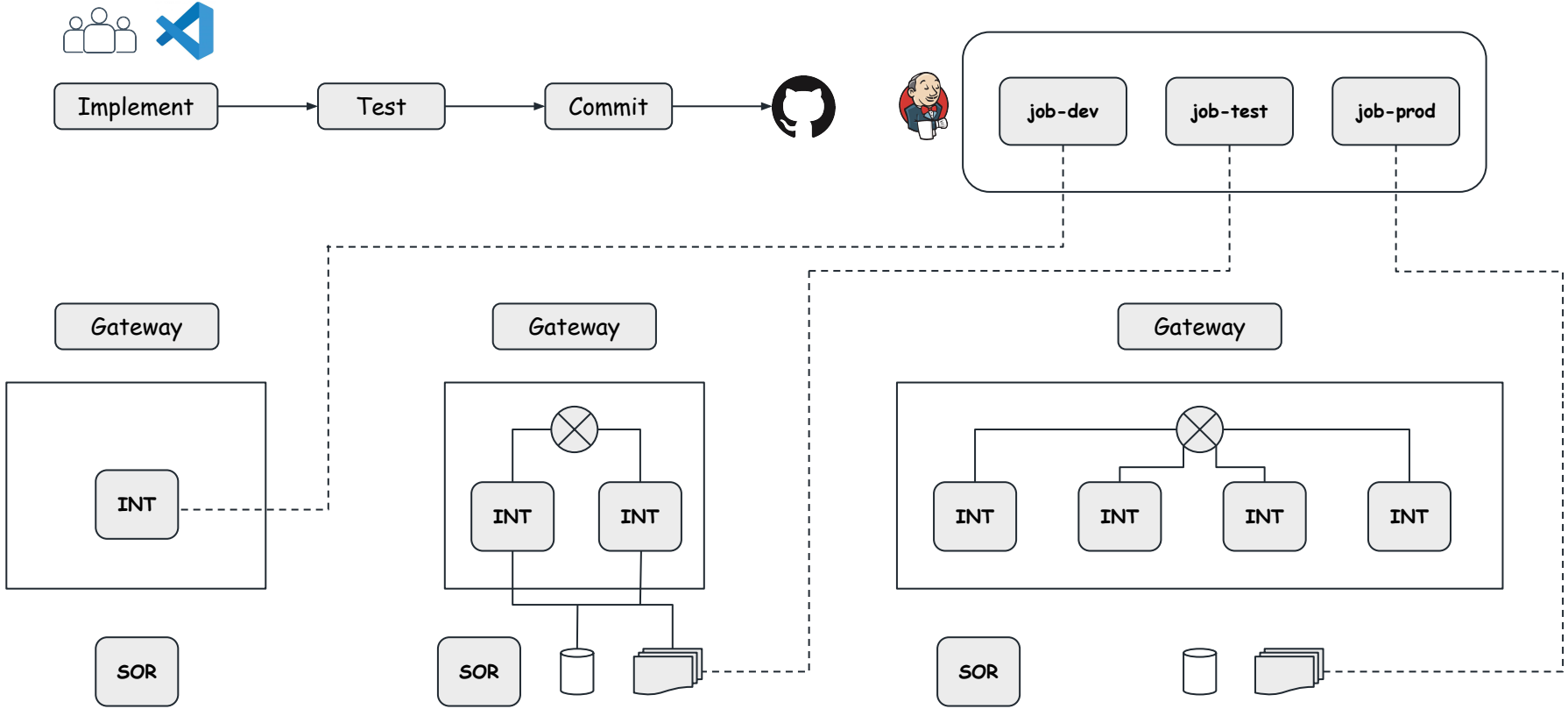
Reference - https://www.enterpriseintegrationpatterns.com/patterns/messaging/toc.html

Twelve-factor app development

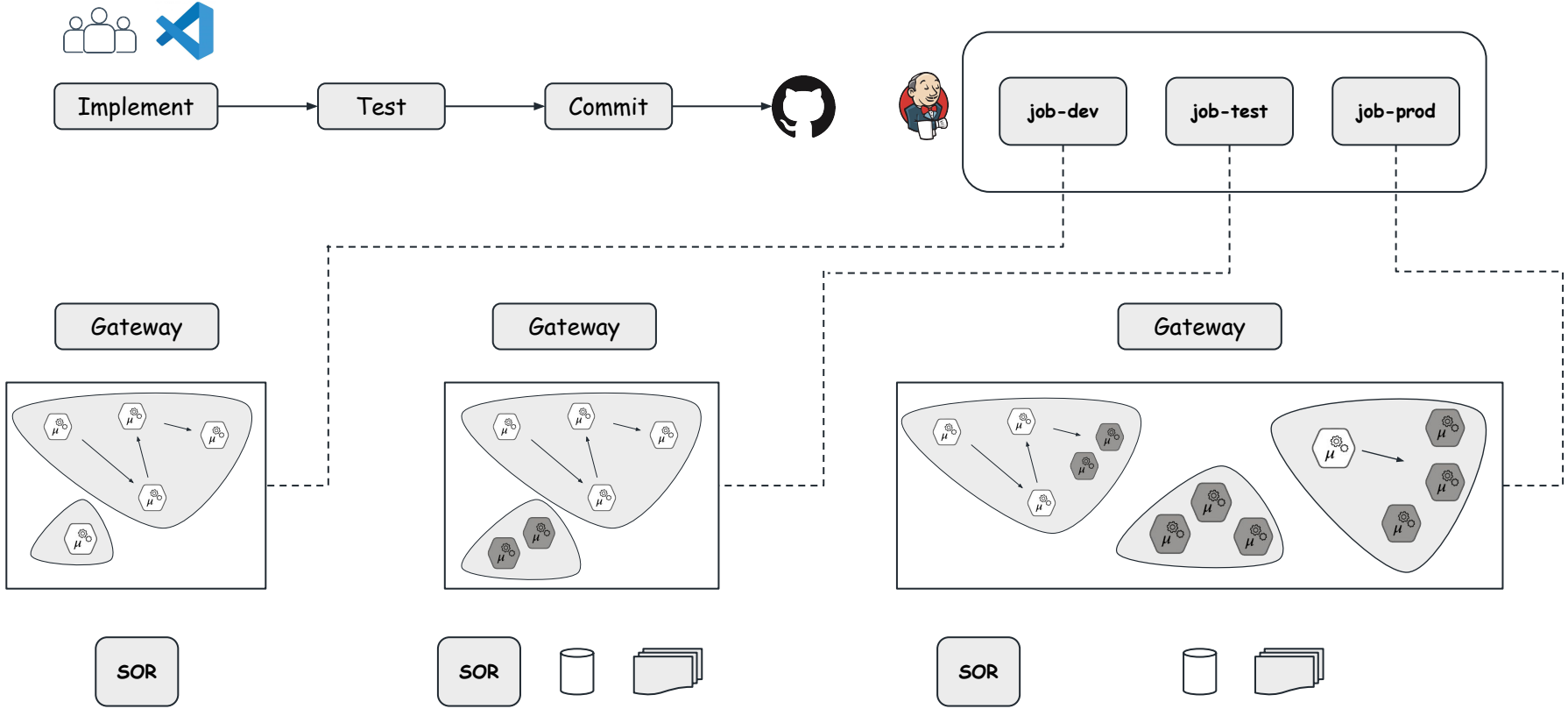


Deployment Architecture

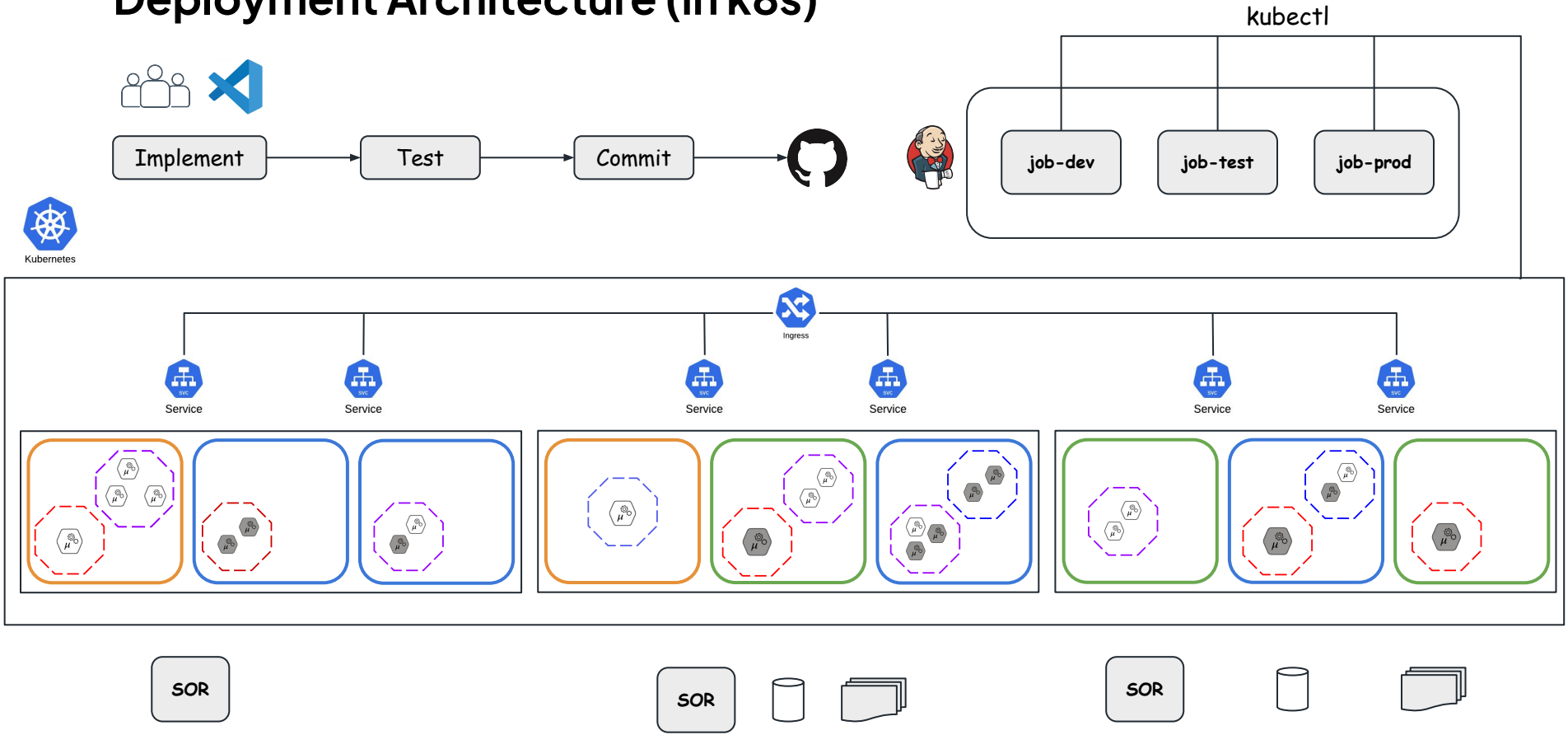
Deployment Architecture (Centralized ESB style)



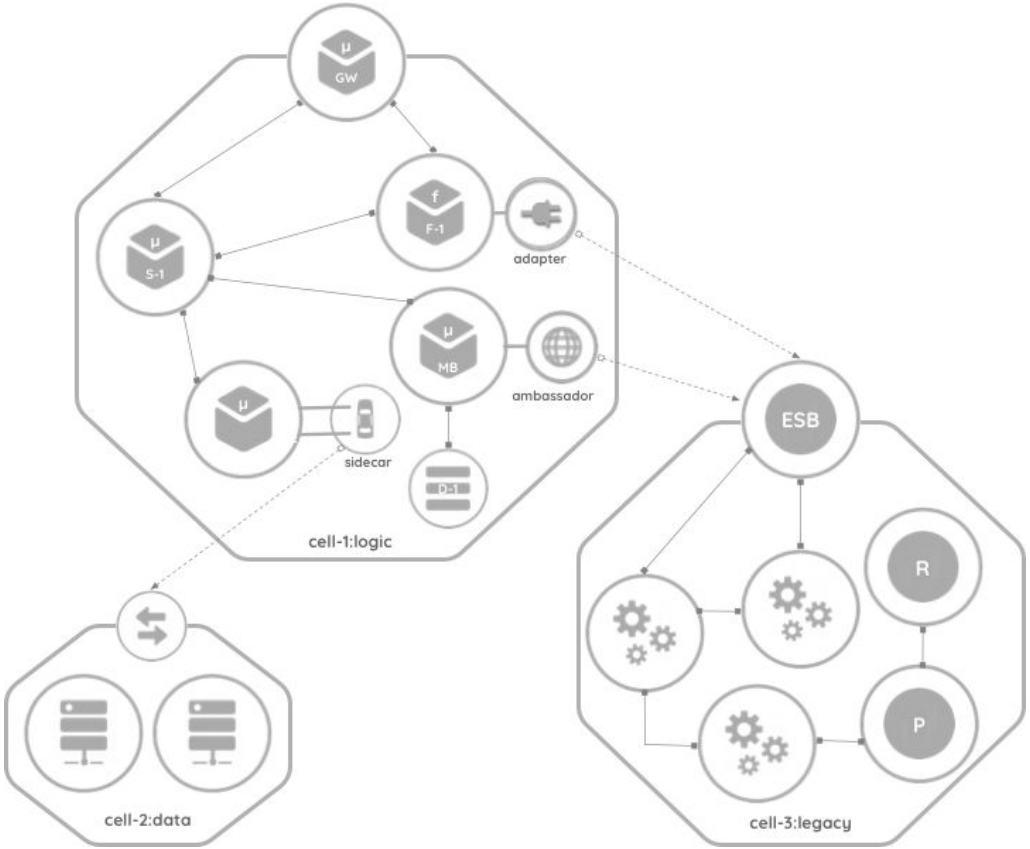
Deployment Architecture (Microservices style)



Deployment Architecture (in k8s)

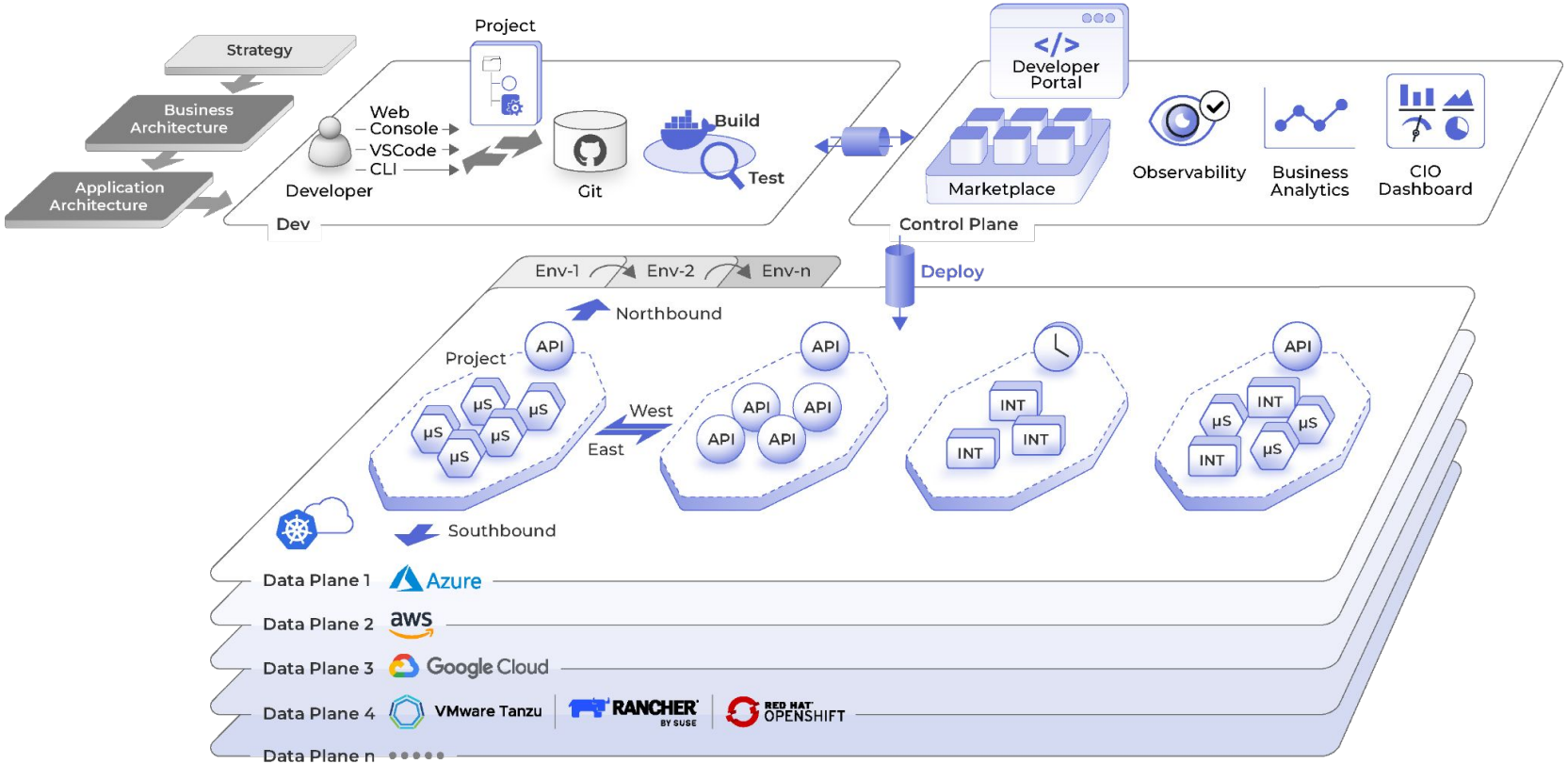


Cell Based Architecture

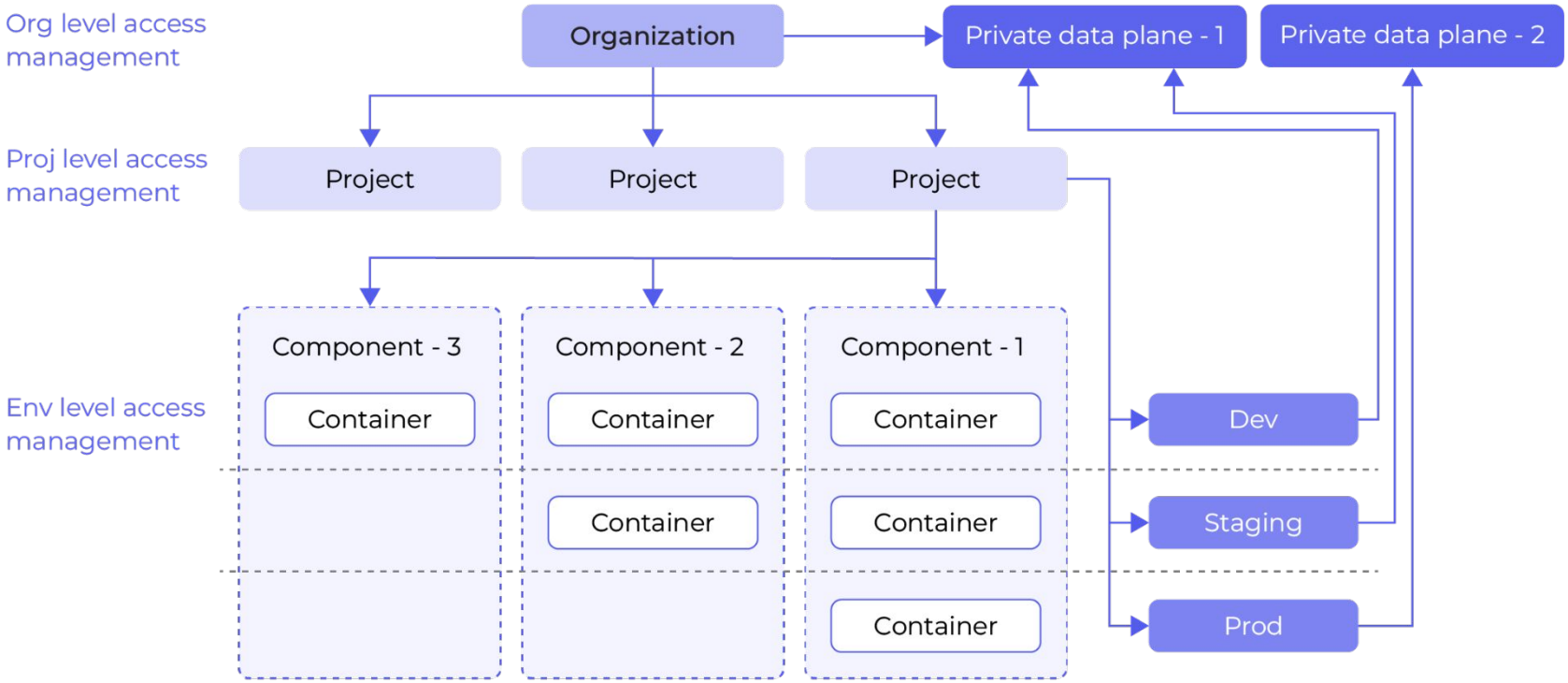


https://github.com/wso2/reference-architecture/blob/master/reference-architecture-cell-based.md

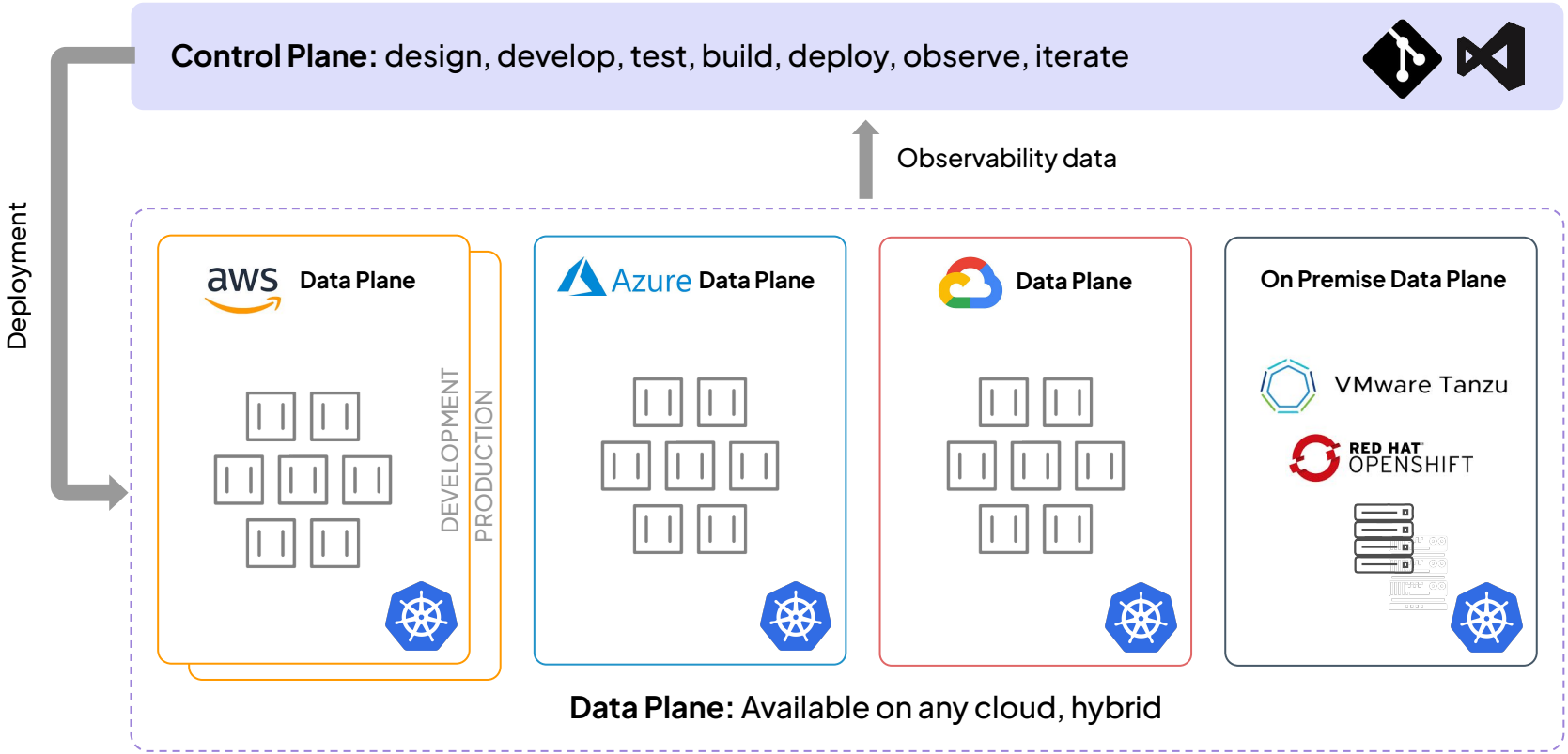
Integration Platform as a Service (iPaaS)



Hierarchies, teams, and permissions



Multi-cloud


Control Plane: design, develop, test, build, deploy, observe, iterate
Observability data
Deployment
aws Data Plane
DEVELOPMENT
PRODUCTION
Azure Data Plane
Data Plane
On Premise Data Plane
VMware Tanzu
RED HAT OPENSHIFT
Data Plane: Available on any cloud, hybrid

Key Takeaways

- Modern enterprises are complex and distributed
- Integration is paramount for enterprises to modernize
- Enterprise Integration Architecture is important for innovation, modernization and digital transformation
- Consider it as a key enabler for business transformation
- It’s an iterative process
- Choose right integration topologies that suits your integration needs
- Architect your integration for cloud
- Choose a flexible integration solution that can address multitude of integration design challenges

Software Architecture isn’t about finding the best solution – it’s about finding the least worst solution

- Neil Ford -

Co-author of “Fundamentals of Software Architecture”



Question Time!




Thank you!
20TH ANNIVERSARY EDITION
WSO2CONASIA
PLATFORMLESS MODERNIZATION