Choosing the Right Technology for SAP BTP Solutions

(CAP, RAP, and Open-source Frameworks with JavaScript/Java)

White Paper: Choosing the Right Technology for SAP BTP Solutions

(CAP, RAP, and Open-source Frameworks with JavaScript/Java)

Contents

1. Abstract

This white paper provides SAP professionals with a detailed guide on how to choose the most suitable technology for building SAP BTP solutions. It examines SAP's native development frameworks—CAP (Cloud Application Programming Model) and RAP (Restful ABAP Programming Model)—alongside open-source frameworks in JavaScript and Java. The analysis focuses on architecture, scalability, maintainability, integration, developer experience, and long-term support, enabling informed decisions that align with business and technical requirements.

2. Executive Summary

SAP Business Technology Platform (BTP) offers a robust environment for creating modern enterprise applications. As organizations increasingly adopt cloud-first strategies, selecting the appropriate technology becomes critical. This white paper discusses:

- **CAP:** A comprehensive model tailored for rapid development and seamless integration with SAP services.
- **RAP:** The Restful ABAP Programming Model, designed specifically for ABAP developers to build cloud-ready, RESTful applications that integrate deeply with SAP Fiori and SAP S/4HANA.
- **Open-Source Frameworks (JS/Java):** Tools that offer extensive flexibility and community support, suitable for scenarios demanding customization and integration with diverse ecosystems.

Key decision criteria include technical compatibility, developer proficiency, application requirements, and long-term maintainability. This document concludes with actionable recommendations to help SAP professionals align technology choices with strategic business outcomes.

3. Introduction

In the rapidly evolving landscape of enterprise software, SAP BTP stands out as a unified platform enabling innovation through integration, extension, and the acceleration of digital transformation. For SAP professionals responsible for application development, selecting the right technology stack is not just about coding—it's about ensuring agility, scalability, and seamless integration with existing SAP environments.

This white paper explores three primary technology paths:

- **SAP Native Models:** CAP and RAP, both designed to work cohesively within the SAP ecosystem.
- **Open-source Frameworks:** Leveraging widely adopted languages and tools like JavaScript and Java to foster flexibility and innovation.

By examining each option's strengths and trade-offs, we aim to provide clarity on how to choose the best approach tailored to specific project needs and organizational goals.

---

4. SAP Business Technology Platform (BTP) Overview

SAP BTP is a cloud platform that unifies database and data management, analytics, application development, and intelligent technologies into a single environment. Key features include:

- **Integration with SAP S/4HANA:** Seamless connectivity with core enterprise data.
- **Scalability and Flexibility:** Support for multi-cloud strategies and diverse application development models.
- **Ecosystem and Extensions:** A robust marketplace and pre-built services that accelerate time-to-value.

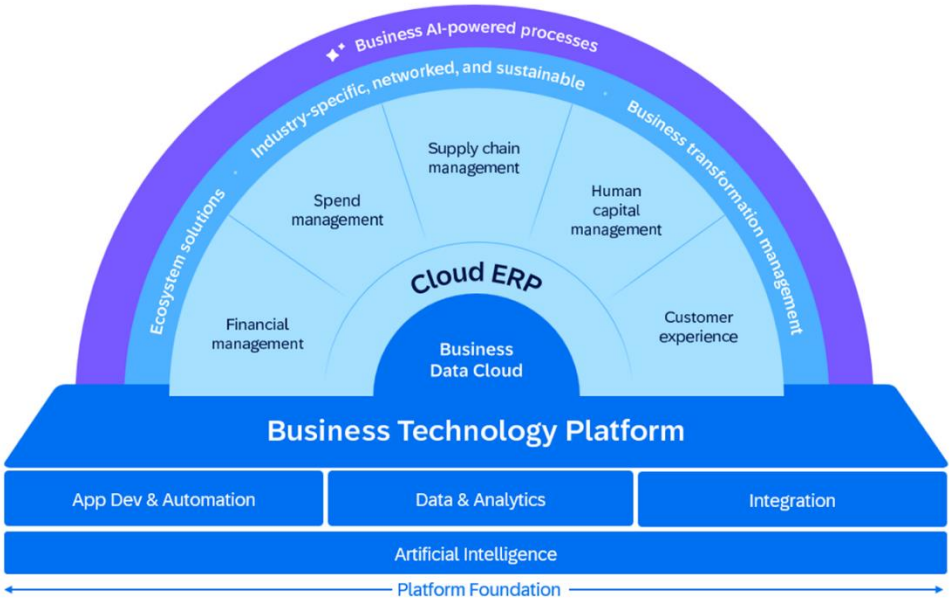


Figure 1. SAP BTP Overview

Understanding BTP's capabilities is essential when evaluating which technology model (CAP, RAP, or open-source) best aligns with the organizational architecture and strategic vision.

5. Technology Options for SAP BTP

Cloud Application Programming Model (CAP)

Overview:
CAP is SAP's opinionated development framework that simplifies the creation of enterprise-grade applications. It abstracts much of the underlying complexity, offering conventions over configuration to streamline the development process.

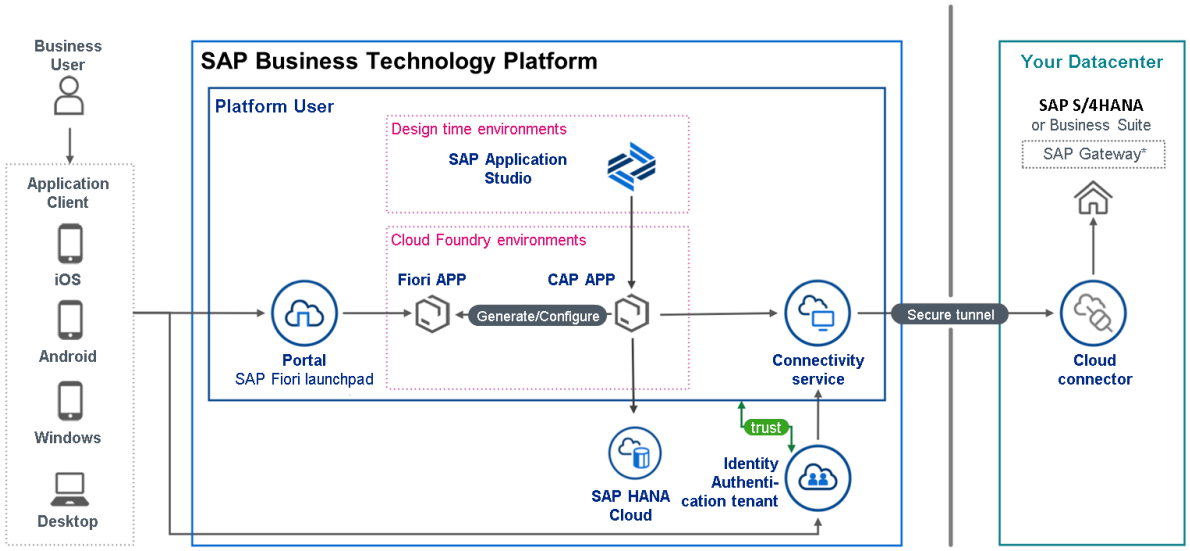


Figure 2. Example Architecture of a Full Stack Side-By-Side Extension with CAP

Key Features:

- **Rapid Development:** Emphasizes speed by using predefined templates and best practices.
- **Integration:** Deep integration with SAP services (e.g., SAP HANA, Fiori) and BTP environments.
- **Multi-Language Support:** Primarily uses Node.js and Java, allowing teams to choose based on their expertise.
- **Service Orientation:** Simplifies the creation of services and APIs tailored for enterprise needs.

Pros:

- Tight integration with the SAP ecosystem.

- Robust support and documentation from SAP.
- Pre-configured environments reduce setup time.

Cons:

- May feel restrictive for highly customized or non-standard requirements.
- Less flexibility compared to building with open-source frameworks from scratch.

Restful ABAP Programming Model (RAP)

Overview:
RAP stands for the **Restful ABAP Programming Model**. It is a framework designed specifically for ABAP developers to build cloud-ready applications that adhere to RESTful principles. RAP leverages the strengths of ABAP, enabling the creation of scalable and maintainable services that integrate seamlessly with SAP Fiori and SAP S/4HANA.

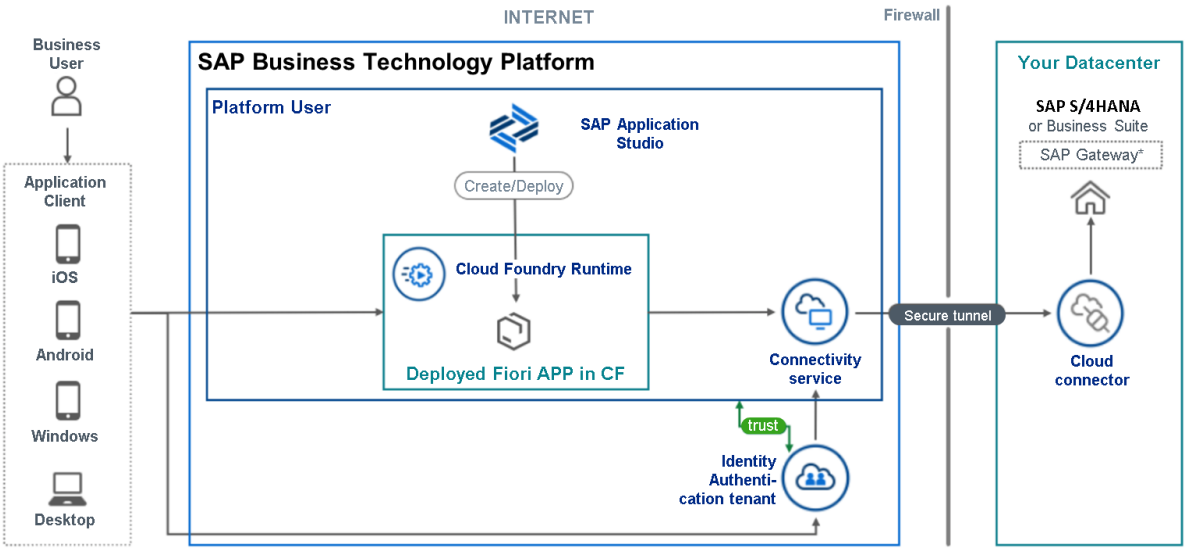


Figure 3. Example Architect of a RAP application (BTP + S/4HANA)

Key Features:

- **ABAP-Centric Development:** Empowers ABAP developers to use their existing skills to create modern, cloud-based applications.
- **RESTful Design:** Adopts RESTful principles to ensure interoperability and modern service-oriented architecture.
- **Seamless Integration:** Provides out-of-the-box connectivity with SAP Fiori, SAP S/4HANA, and other SAP systems.

- **Standardization and Consistency:** Promotes standardized development practices to enhance maintainability and scalability.
- **Optimized for Cloud:** Tailored for the SAP BTP environment, ensuring performance and efficiency in cloud deployments.

Pros:

- Deep integration with the SAP ecosystem tailored for ABAP developers.
- Leverages existing ABAP skills and infrastructure.
- Ensures consistent development practices aligned with SAP standards.

Cons:

- Requires proficiency in ABAP and a shift towards RESTful paradigms.
- May have a steeper learning curve for teams transitioning from traditional on-premises ABAP development to cloud-based RESTful approaches.

Open-Source Frameworks (JavaScript/Java)

Overview:
Using open-source frameworks (such as Express.js, Angular, Spring Boot, etc.) allows organizations to leverage the vast ecosystem of non-SAP tools while integrating with SAP BTP. These frameworks are especially appealing when customization and a non-standard approach are required.

Key Features:

- **Flexibility:** No strict guidelines, allowing full customization of the application stack.
- **Community Support:** Extensive libraries, plugins, and active communities for JavaScript and Java.
- **Interoperability:** Can integrate with a variety of SAP services via APIs and connectors.
- **Diverse Ecosystem:** Choose from a wide array of tools and libraries to match specific needs.

Pros:

- High degree of control over architecture and design.
- Ability to adopt cutting-edge development practices.
- Broad talent pool, as many developers are proficient in these languages.

Cons:

- Requires more initial configuration and setup.

- Less out-of-the-box integration with SAP BTP compared to CAP or RAP.
- Potentially higher maintenance if not following a consistent architectural standard.

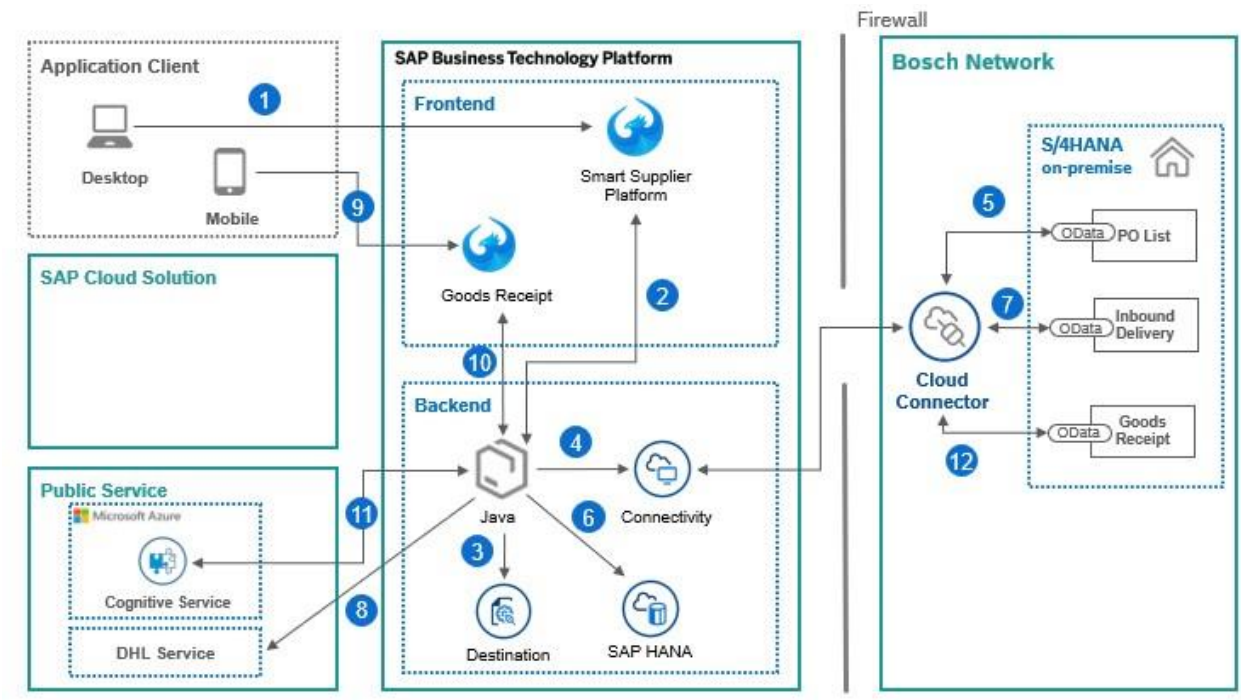


Figure 4. Example Architect of an solution usining open-source framework (Fiori + Java)

6. Decision Criteria for Technology Selection

When selecting the right technology for your SAP BTP solution, consider the following factors:

1. **Business Requirements:**
    - **Time-to-Market:** How quickly must the solution be developed and deployed?
    - **Complexity:** Does the project require extensive customization or adhere to standard processes?
2. **Technical Fit:**
    - **Integration Needs:** How deeply does the solution need to interact with SAP services (e.g., S/4HANA, Fiori)?
    - **Scalability and Performance:** Are there significant scalability or performance requirements?
3. **Development and Maintenance:**
    - **Developer Expertise:** Is your team more familiar with SAP-centric models (CAP/RAP) or open-source languages (JS/Java)?
    - **Long-Term Support:** What is the anticipated lifecycle of the application and how will updates be managed?
4. **Cost and Resource Availability:**
    - **Licensing and Support Costs:** Consider the total cost of ownership including SAP support versus open-source community-driven support.
    - **Development Resources:** Evaluate availability of skilled developers and community resources.
5. **Future-Proofing:**
    - **Innovation and Flexibility:** Does the technology allow for rapid adoption of new features or pivoting based on changing business needs?
    - **Community and Ecosystem Growth:** Consider the roadmap and community support around the chosen technology.

---

7. Comparative Analysis

Criteria	CAP	RAP (Restful ABAP Programming Model)	Open-source (JS/Java)
Integration with SAP	Deep, native integration with SAP BTP services.	Optimized for ABAP-based development with seamless integration to SAP Fiori and S/4HANA.	Requires connectors; integration is manual.
Development Speed	Rapid with predefined templates and conventions.	Streamlined for ABAP developers adopting RESTful patterns; moderate learning curve.	Highly variable; custom setup required.
Flexibility/Customization	Moderate -designed for SAP best practices.	Focused on standardized, ABAP-driven design which may be less	High—full control over design and architecture.

Criteria	CAP	RAP (Restful ABAP Programming Model)	Open-source (JS/Java)
		flexible for non-ABAP-centric needs.	
Developer Ecosystem	Strong within the SAP community.	Ideal for teams with strong ABAP expertise; growing focus on modern ABAP development.	Broad global community and resources.
Learning Curve	Lower for SAP professionals.	Moderate—requires understanding both ABAP and RESTful design in a cloud context.	Variable; can be steep if building from scratch.
Maintenance & Support	Backed by SAP's support and roadmap.	Supported by SAP with clear guidelines for ABAP development in the cloud.	Community-driven; may require additional governance.

8. Recommendations

Based on the comparative analysis and decision criteria, here are actionable recommendations for SAP professionals:

1. **Leverage CAP When:**
    - Rapid development and deep integration with SAP services are paramount.
    - The project fits well within standard SAP processes.
    - The development team has strong SAP-centric skills and is comfortable with Node.js or Java.
2. **Adopt RAP If:**
    - Your team consists primarily of ABAP developers seeking to transition to cloud-based, RESTful applications.
    - The project involves creating scalable, modern applications that integrate seamlessly with SAP Fiori and S/4HANA.
    - Standardized, maintainable, and SAP-guided development practices are desired.
3. **Choose Open-Source Frameworks (JS/Java) When:**
    - Flexibility and extensive customization are required.
    - Integration with non-SAP systems is a priority.
    - Your team possesses strong expertise in open-source technologies and can manage additional integration overhead.

---

9. Conclusion

Selecting the appropriate technology for SAP BTP solutions requires a careful balance between rapid development, seamless integration, and long-term flexibility. CAP and RAP provide powerful, SAP-optimized approaches that ensure tight integration and support, while open-source frameworks offer the flexibility needed for highly customized solutions and integration with diverse ecosystems.

By evaluating project-specific requirements against the decision criteria outlined in this white paper, SAP professionals can make informed choices that not only align with current business needs but also pave the way for future innovation and scalability.

10. References

- SAP Cloud Application Programming Model (CAP) Documentation
- SAP Restful ABAP Programming Model (RAP) Guidelines
- SAP Business Technology Platform (BTP) Overview
- Industry case studies and best practices in enterprise application development

---

This white paper is intended for SAP professionals engaged in developing, integrating, or architecting SAP BTP solutions. For further inquiries or detailed technical assistance, please refer to official SAP documentation or contact your SAP representative.

Appendix

RAP

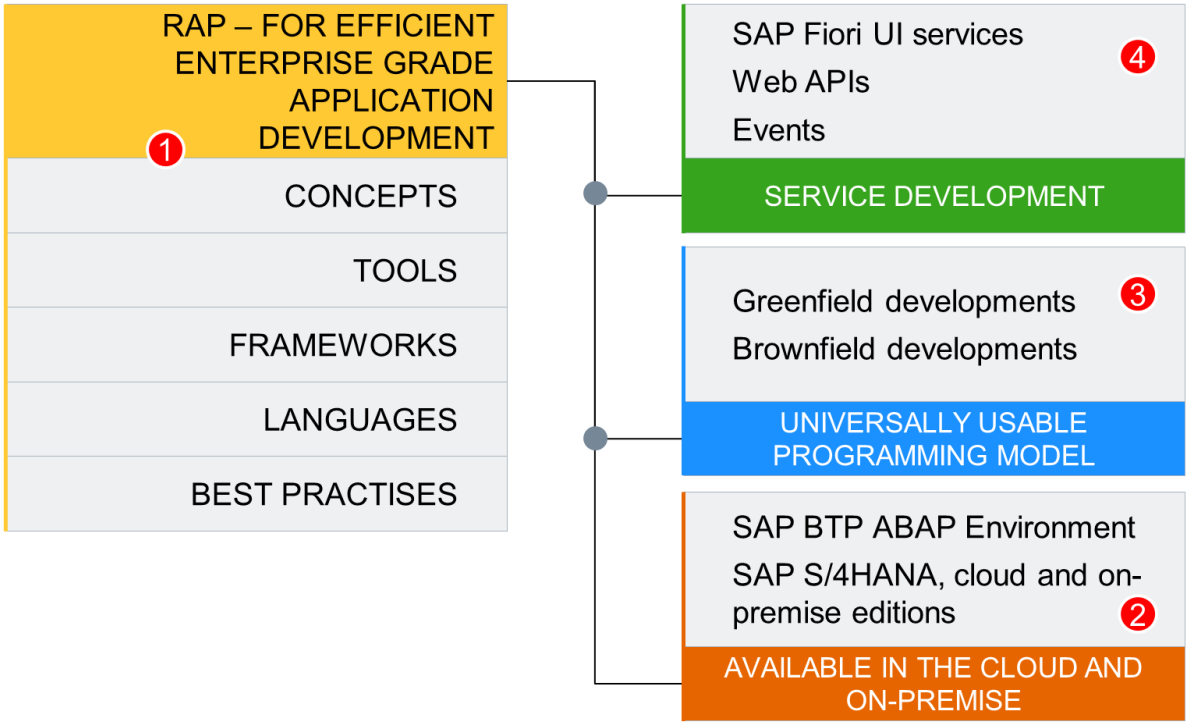
Below you see a matrix outlining some of the different ABAP programming frameworks (including historical ones) that can be used with different products / versions and the specific features available. As an essential part of ABAP Cloud, ABAP RESTful application programming enables development using all modern features that cloud native design requires. These include supporting both OData V2 & V4 as well as annotations both in back-end CDS views as well as in SAP Fiori tools as part of SAP Business Application Studio.

RAP Programming Models Matrix (By Product | By Version | By Feature)

SAP Gateway OData V2 framework	SAP Gateway OData V4 framework	ABAP Programming Model for SAP Fiori (BOPF)	ABAP RESTful Application Programming Model (RAP)	ABAP RESTful Application Programming Model (RAP)	
Release	SAP ECC (NW ≥ 7.0)	SAP S/4HANA ≥ 1709 (≥ NW 7.52)	SAP S/4HANA ≥ 1610 (≥ NW 7.51)	SAP BTP ABAP Environment	SAP S/4HANA ≥ 1909
Greenfield implementation	+	+	+	+	≥ 2020 FSP1
Brownfield implementation	+	+	–	+	+
OData V2	+	–	+	+	+
OData V4	–	+	–	+	≥ 2020 FSP1
Draft handling	–	–	+	+	≥ 2020 FSP1
Multi-Inline-Edit	–	–	+	+	–
Asynchronous OData V4 Requests	–	+	–	+	≥ 2020 FSP1
Backend annotations in CDS views	(+)	–	+	+	+
Backend annotations using code based implementation	+	+	(+)	–	–
Local annotations in SAP Fiori Tools	+	+	+	+	+

Figure 5. RAP Programming Models Matrix (By Product | By Version | By Feature)

ABAP RESTful Application Programming Model - Birds Eye View



No. 1

A collection of concepts, tools, best practices, frameworks, generators and much more.

A strategic development approach for SAP and an essential part of ABAP Cloud

No. 2

Can be used in the following products.

- SAP S/4HANA Cloud public edition
- SAP S/4HANA Cloud private edition
- SAP S/4HANA (on-premises deployment)
- SAP BTP ABAP Environment

No. 3

Can be used for all project implementation scenarios:

- Greenfield
- Brownfield

- System Landscape Transformation

No. 4

The following types of APIs are possible:

- OData: v2 and V4, optimized for use in SAP Fiori apps.
- Web API: OData V2/V4, SQL, for use with any clients.
- Events

ABAP RESTful Application Programming Model from a Technical Perspective

ABAP RESTful application programming model has several layers where artifacts at each layer implement specific functions.

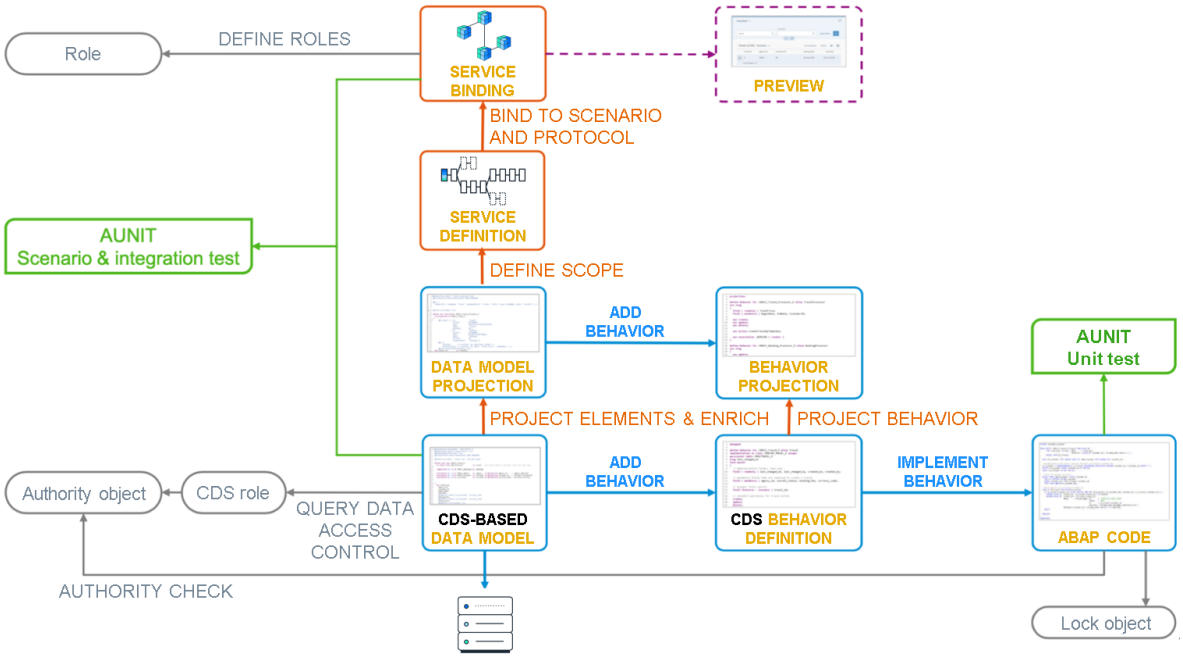


Figure 6. ABAP RESTful Application Programming Model Layers

Building Blocks

Business Objects:

Business objects (BO) represent the data model and define the data-related logic, called behavior, independent of the specific consumption. ABAP RESTful application programming model business objects are defined by CDS data modeling views, CDS behavior definitions, and behavior implementations in ABAP classes.

Business Object Projections:

The business object projection is an approach to project a subset of the business object for a specific business service. The projection enables flexible service consumption and role-based service designs. In ABAP RESTful application programming model, a BO projection consists of CDS projection views, CDS behavior projections and, if necessary, additional, or consumption-specific implementations.

Service Definition:

A service definition defines the scope of a business service, in particular the business object projection that is to be represented through this service.

Service Binding:

A service binding defines the communication protocol such as OData V2 or OData V4 and the type of service to be offered to a consumer, such as UI services or a Web service.

SAP Fiori UI:

SAP Fiori UI provides a special user interface for frequently used application patterns. In this case, for example, UI annotations are transferred to the SAP Fiori app development. The SAP Fiori Ui is created on SAP BTP in the SAP Business Application Studio. It can be hosted either in the SAP S/4HANA back-end or on SAP BTP.

Web API:

A Web API provides a public interface for accessing the OData service by any OData client independent of a particular UI.

Creation of SAP Fiori UIs

The following is an example of using SAP Business Application Studio to create an SAP Fiori app. After creating the SAP Fiori app, you can decide where the app should be deployed.

Either on:

- SAP BTP, Cloud Foundry runtime or
- SAP S/4HANA (all possible deployments)

Source: Exploring Application Frameworks