SAP BTP Programming Models Development Enablement (RAP)

Introduction to ABAP RESTful Application Programming Model

Leonardo Britz, Pedro Sangaletti, SAP

CONFIDENTIAL

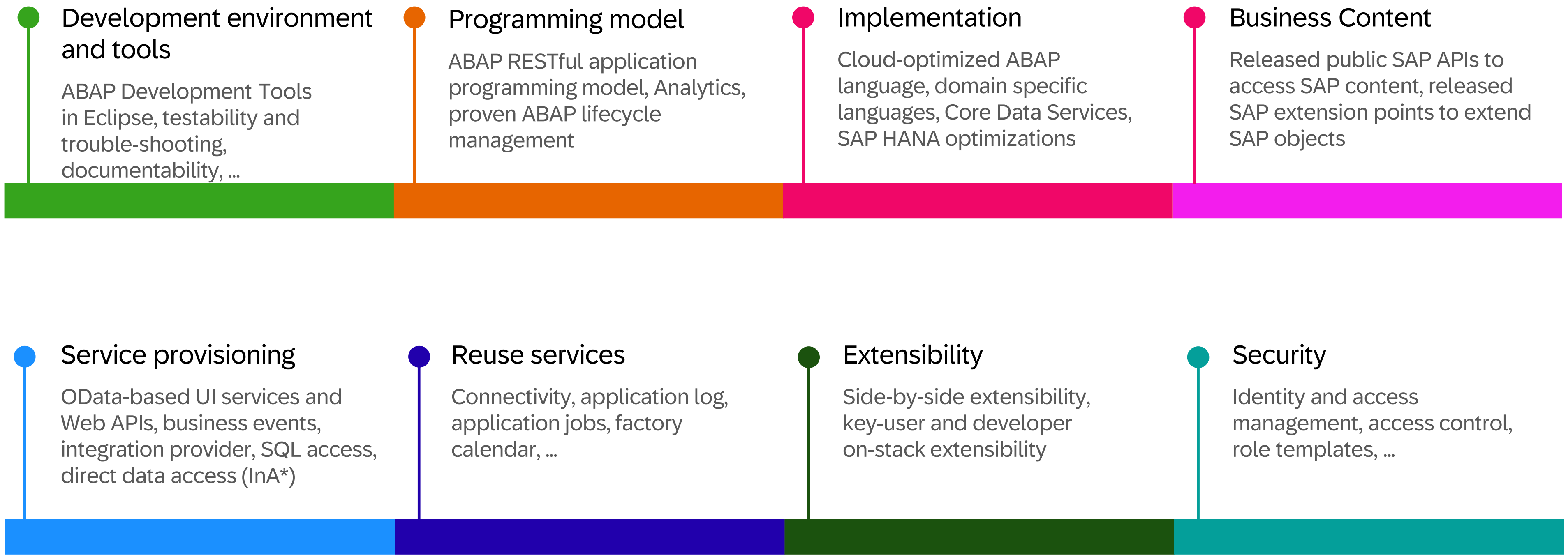
Evolution of the ABAP Programming Model

Clean Core with the ABAP Cloud development model

ABAP Cloud

… is the ABAP development model to build cloud-ready business apps, services and extensions.
… comes with SAP Business Technology Platform (BTP) and SAP S/4HANA.
… works with public or private cloud, and even on-premise.

ABAP Cloud at a glance



Development environment and tools

ABAP Development Tools in Eclipse, testability and trouble-shooting, documentability, …

Programming model

ABAP RESTful application programming model, Analytics, proven ABAP lifecycle management

Implementation

Cloud-optimized ABAP language, domain specific languages, Core Data Services, SAP HANA optimizations

Business Content

Released public SAP APIs to access SAP content, released SAP extension points to extend SAP objects

Service provisioning

OData-based UI services and Web APIs, business events, integration provider, SQL access, direct data access (InA*)

Reuse services

Connectivity, application log, application jobs, factory calendar, …

Extensibility

Side-by-side extensibility, key-user and developer on-stack extensibility

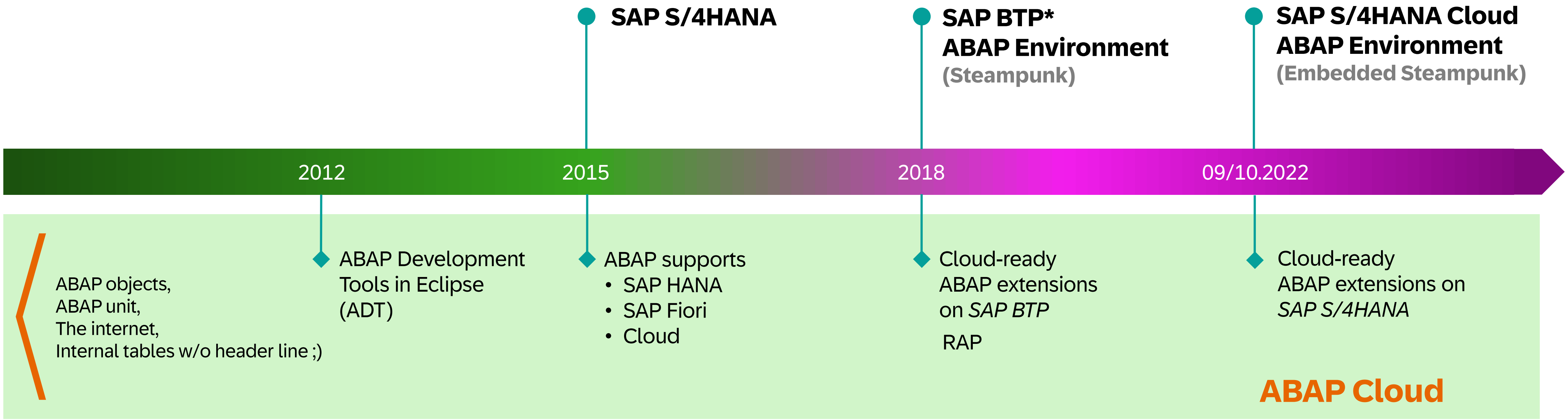
Security

Identity and access management, access control, role templates, …

CONFIDENTIAL

* SAP Information Access protocol

The evolution to ABAP Cloud



SAP S/4HANA

SAP BTP*
ABAP Environment
(Steampunk)

SAP S/4HANA Cloud
ABAP Environment
(Embedded Steampunk)

2012 2015 2018 09/10.2022

ABAP objects,
ABAP unit,
The internet,
Internal tables w/o header line ;)

ABAP Development Tools in Eclipse (ADT)

ABAP supports
- SAP HANA
- SAP Fiori
- Cloud

Cloud-ready ABAP extensions on *SAP BTP*

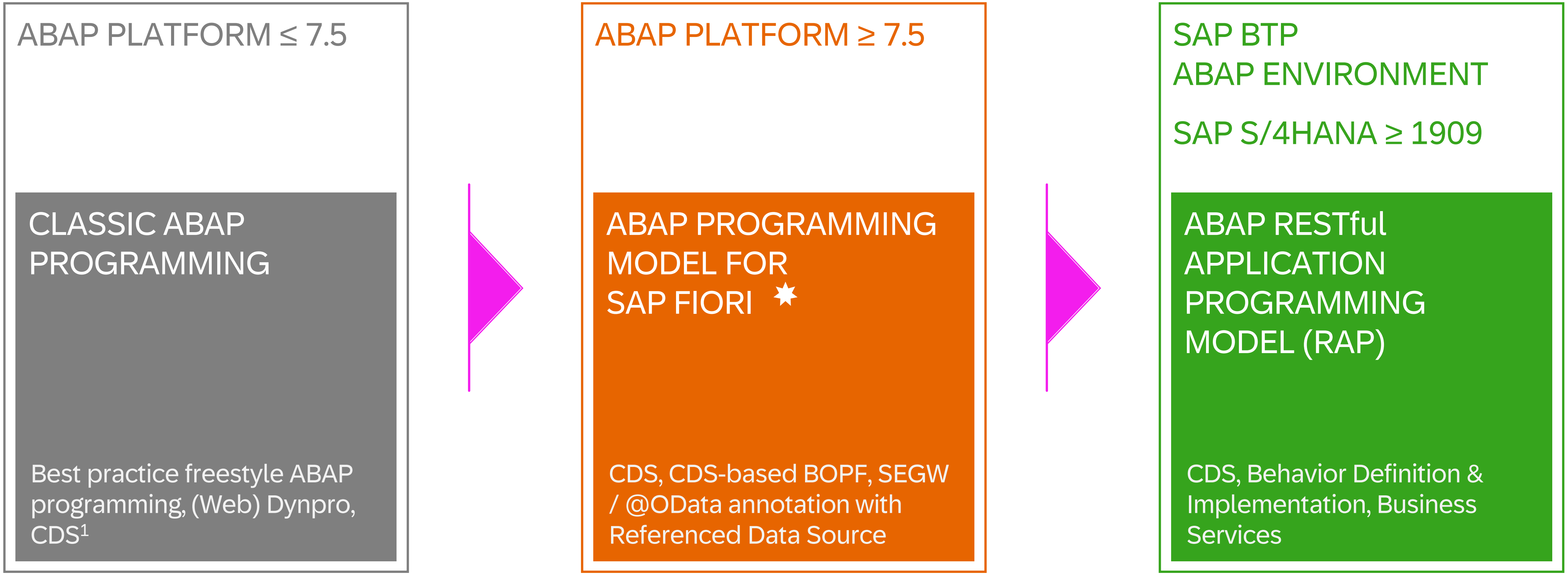
RAP

Cloud-ready ABAP extensions on *SAP S/4HANA*

ABAP Cloud

* SAP Business Technology Platform

Evolution of the ABAP programming model



ABAP PLATFORM ≤ 7.5

CLASSIC ABAP PROGRAMMING

Best practice freestyle ABAP programming, (Web) Dynpro, CDS[1]

ABAP PLATFORM ≥ 7.5

ABAP PROGRAMMING MODEL FOR SAP FIORI ✸

CDS, CDS-based BOPF, SEGW / @OData annotation with Referenced Data Source

SAP BTP
ABAP ENVIRONMENT

SAP S/4HANA ≥ 1909

ABAP RESTful APPLICATION PROGRAMMING MODEL (RAP)

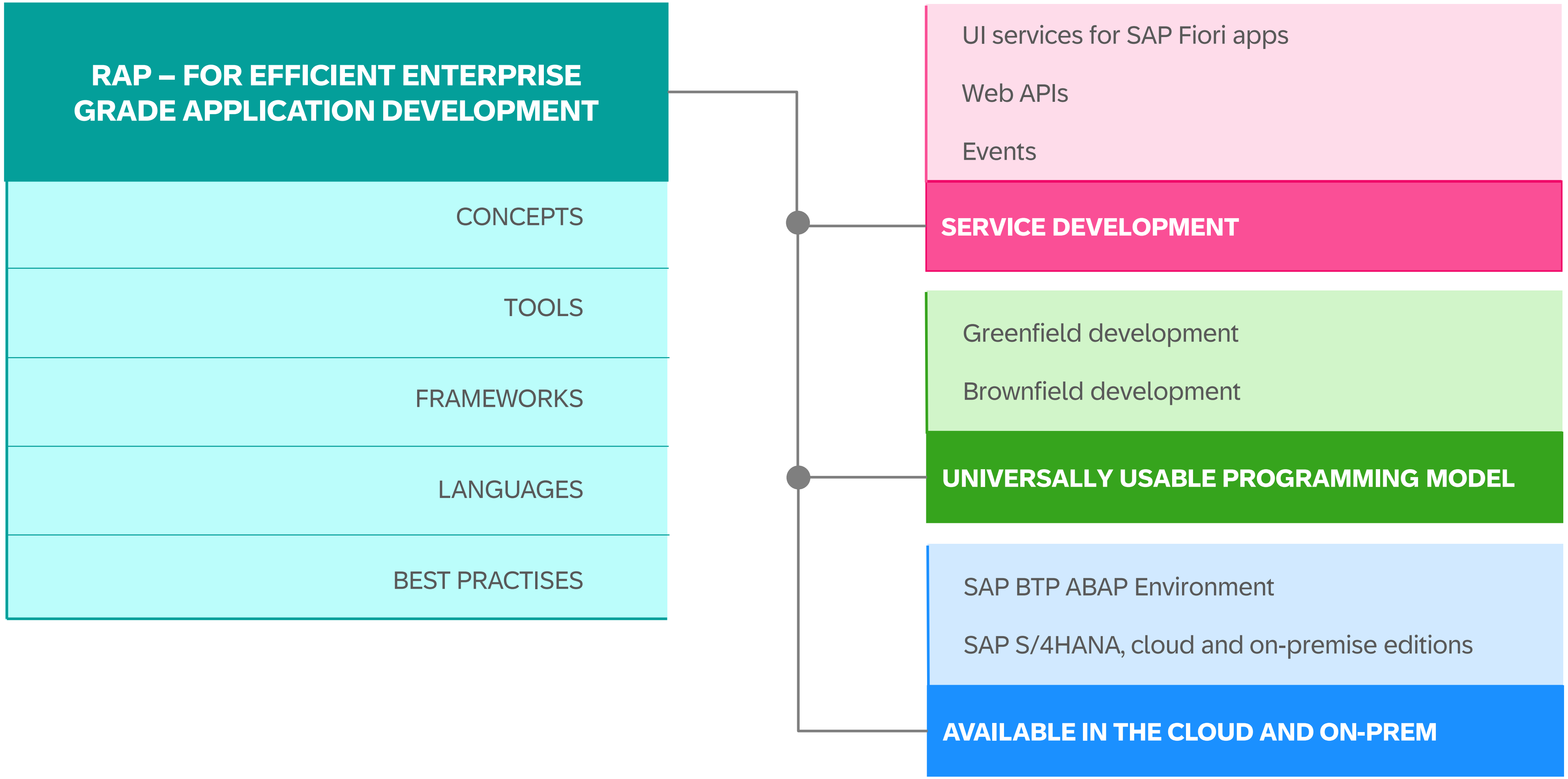
CDS, Behavior Definition & Implementation, Business Services

[1] starting with release 7.4 SPS05

✸ Safe investments!

RAP the Big Picture

ABAP RESTful Application Programming Model (RAP) at a glance



RAP – FOR EFFICIENT ENTERPRISE GRADE APPLICATION DEVELOPMENT

CONCEPTS

TOOLS

FRAMEWORKS

LANGUAGES

BEST PRACTISES

UI services for SAP Fiori apps

Web APIs

Events

SERVICE DEVELOPMENT

Greenfield development

Brownfield development

UNIVERSALLY USABLE PROGRAMMING MODEL

SAP BTP ABAP Environment

SAP S/4HANA, cloud and on-premise editions

AVAILABLE IN THE CLOUD AND ON-PREM

The Key Players

ABAP Development Tools in Eclipse for all development tasks

Easy developer onboarding
End-to-end development flow

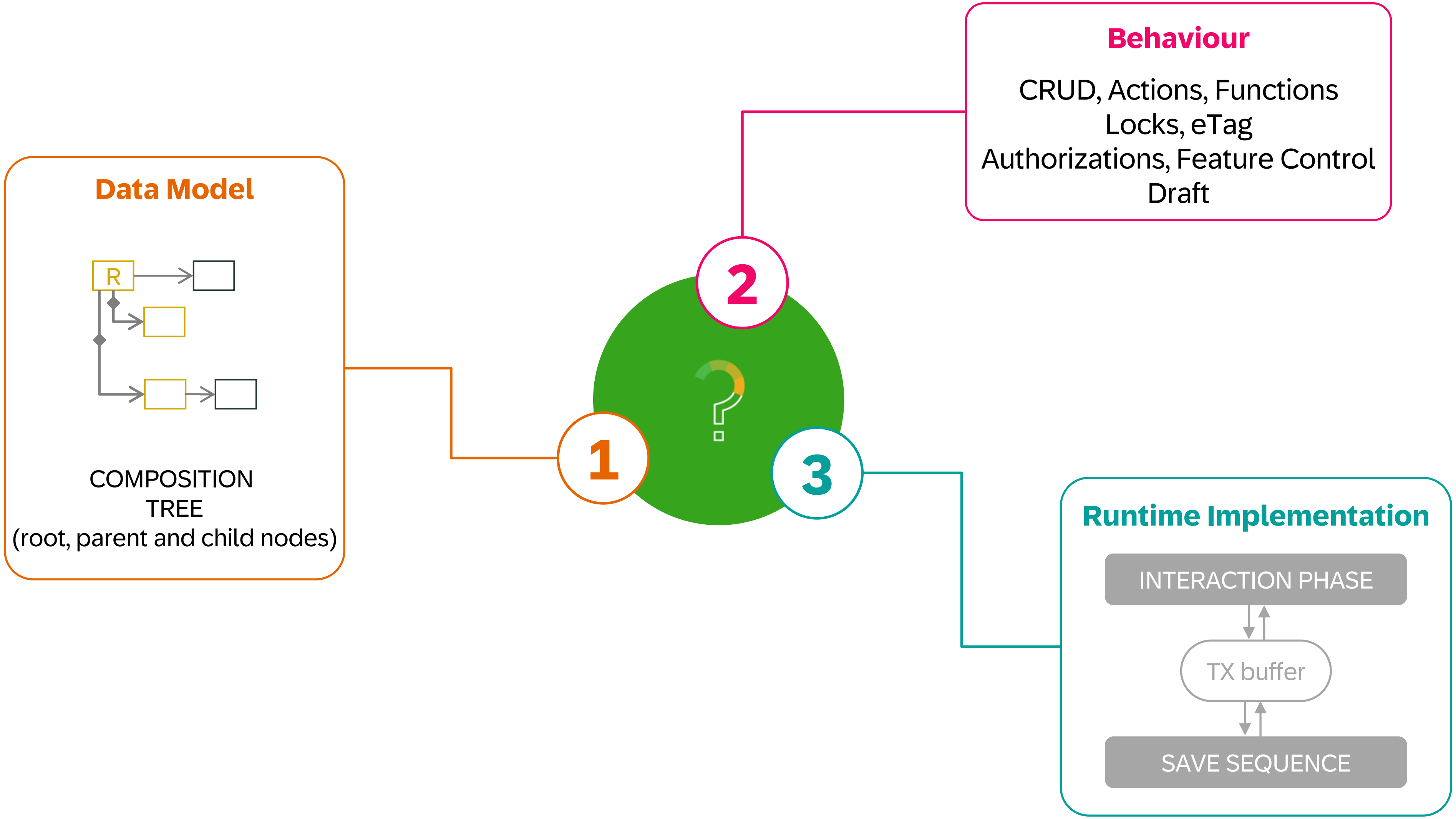
Languages: ABAP and Core Data Services (CDS)

Standard implementation tasks via typed APIs supporting static code checks, auto-completion, element info

Powerful frameworks

Take over technical implementation tasks
Business logic added in code exits on protocol-agnostic layers

Business Object (BO)



Data Model

COMPOSITION
TREE
(root, parent and child nodes)

Behaviour

CRUD, Actions, Functions
Locks, eTag
Authorizations, Feature Control
Draft

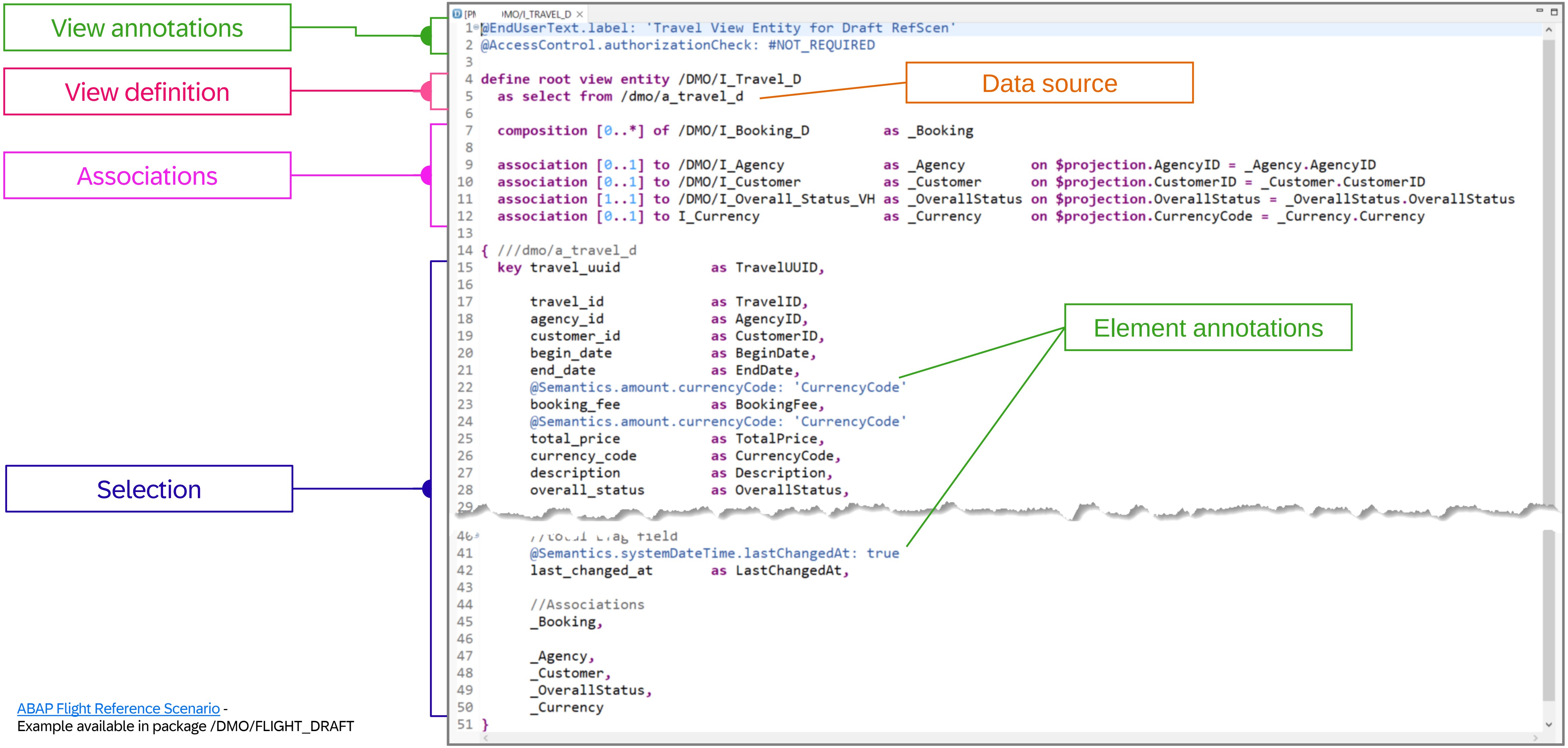
Runtime Implementation

INTERACTION PHASE

TX buffer

SAVE SEQUENCE

Next generation data modeling and access with ABAP CDS



View annotations

View definition

Associations

Selection

Data source

Element annotations

```
@EndUserText.label: 'Travel View Entity for Draft RefScen'
@AccessControl.authorizationCheck: #NOT_REQUIRED

define root view entity /DMO/I_Travel_D
  as select from /dmo/a_travel_d

  composition [0..*] of /DMO/I_Booking_D         as _Booking

  association [0..1] to /DMO/I_Agency            as _Agency        on $projection.AgencyID = _Agency.AgencyID
  association [0..1] to /DMO/I_Customer          as _Customer      on $projection.CustomerID = _Customer.CustomerID
  association [1..1] to /DMO/I_Overall_Status_VH as _OverallStatus on $projection.OverallStatus = _OverallStatus.OverallStatus
  association [0..1] to I_Currency               as _Currency      on $projection.CurrencyCode = _Currency.Currency

{ ///dmo/a_travel_d
  key travel_uuid            as TravelUUID,

      travel_id              as TravelID,
      agency_id              as AgencyID,
      customer_id            as CustomerID,
      begin_date             as BeginDate,
      end_date               as EndDate,
      @Semantics.amount.currencyCode: 'CurrencyCode'
      booking_fee            as BookingFee,
      @Semantics.amount.currencyCode: 'CurrencyCode'
      total_price            as TotalPrice,
      currency_code          as CurrencyCode,
      description            as Description,
      overall_status         as OverallStatus,

      //total etag field
      @Semantics.systemDateTime.lastChangedAt: true
      last_changed_at        as LastChangedAt,

      //Associations
      _Booking,

      _Agency,
      _Customer,
      _OverallStatus,
      _Currency
}
```

ABAP Flight Reference Scenario -
Example available in package /DMO/FLIGHT_DRAFT

Entity Manipulation Language (EML)

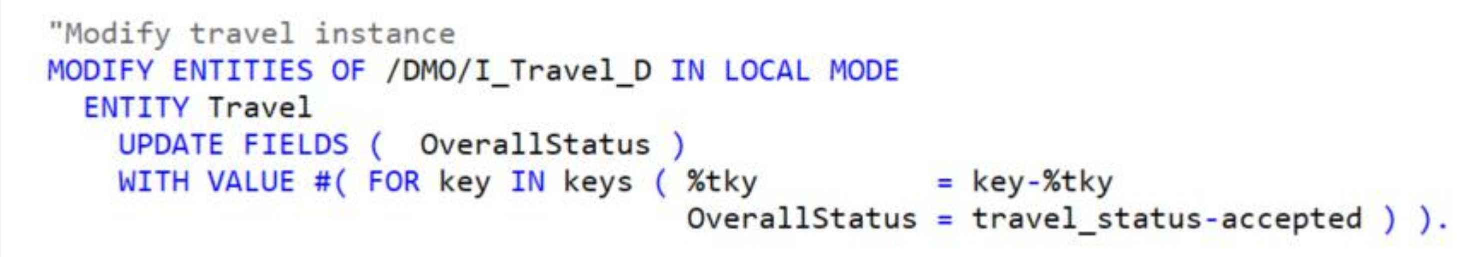
Extension of the ABAP Language with SQL-like syntax

Used to control the transactional RAP business object (BO) behavior

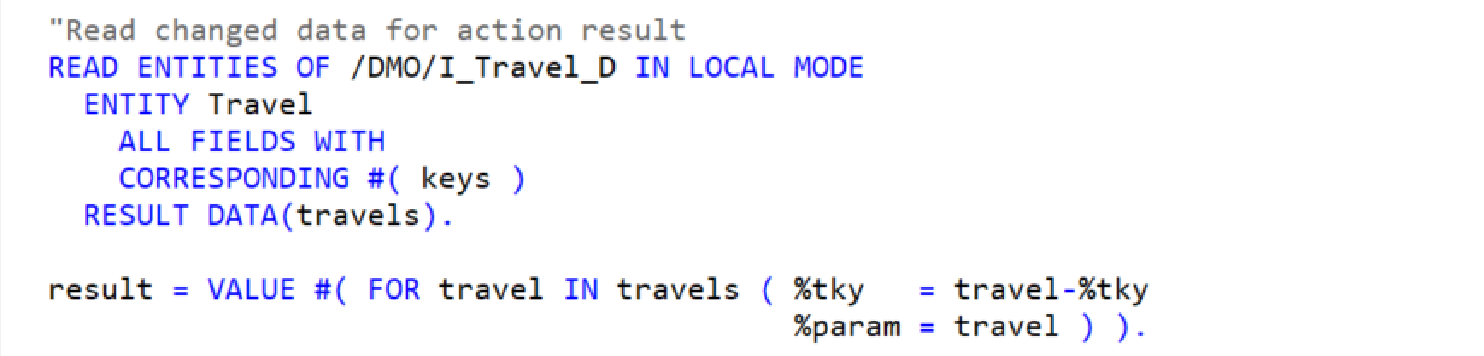
Direct API-based access to RAP BOs

API reference available in the ABAP keyword documentation (link)

EXAMPLES



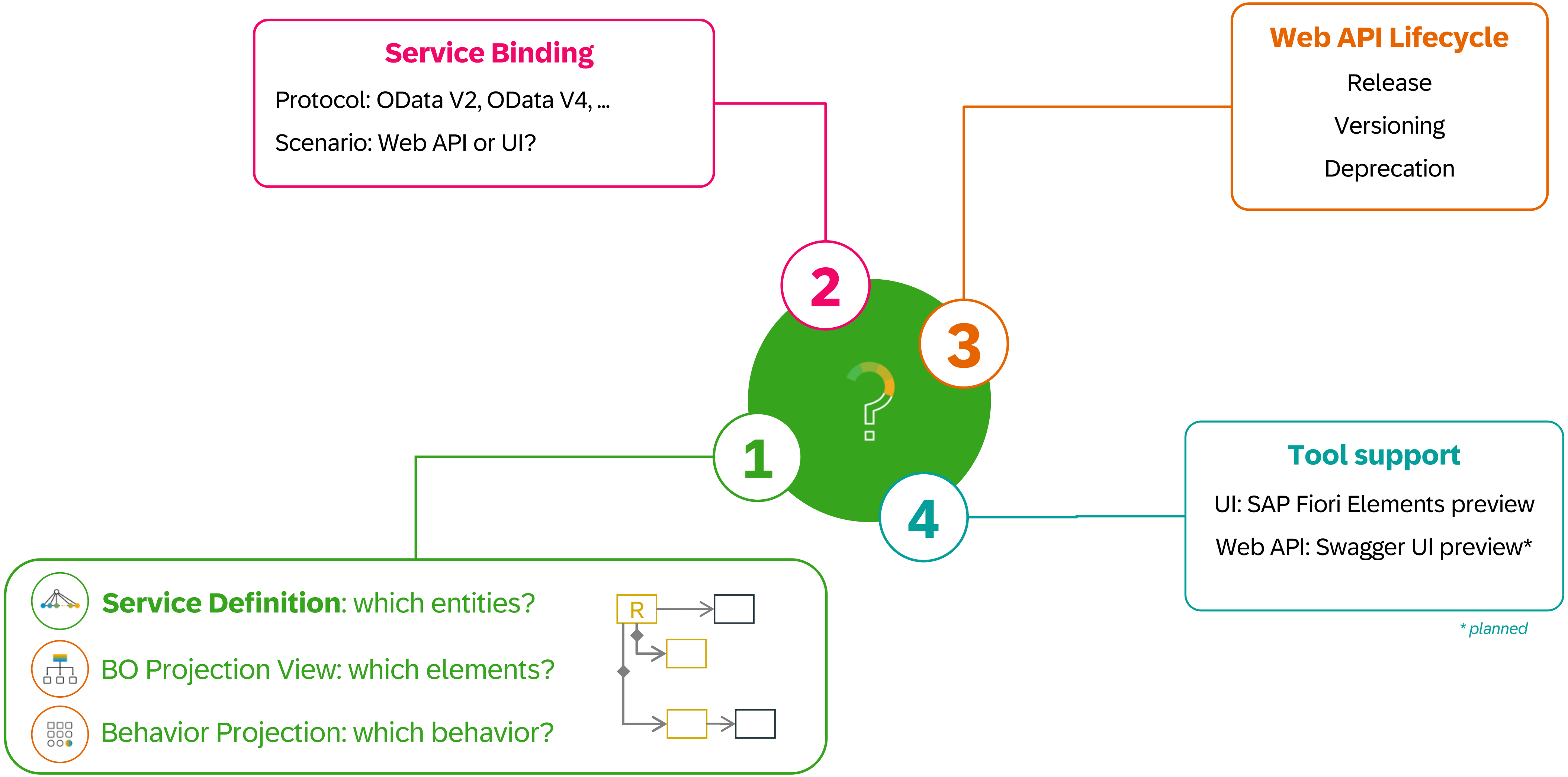
```
"Modify travel instance
MODIFY ENTITIES OF /DMO/I_Travel_D IN LOCAL MODE
  ENTITY Travel
    UPDATE FIELDS (  OverallStatus )
    WITH VALUE #( FOR key IN keys ( %tky          = key-%tky
                                    OverallStatus = travel_status-accepted ) ).
```



```
"Read changed data for action result
READ ENTITIES OF /DMO/I_Travel_D IN LOCAL MODE
  ENTITY Travel
    ALL FIELDS WITH
    CORRESPONDING #( keys )
  RESULT DATA(travels).

result = VALUE #( FOR travel IN travels ( %tky   = travel-%tky
                                          %param = travel ) ).
```

Business Service



Service Binding

Protocol: OData V2, OData V4, …

Scenario: Web API or UI?

Web API Lifecycle

Release

Versioning

Deprecation

Tool support

UI: SAP Fiori Elements preview

Web API: Swagger UI preview*

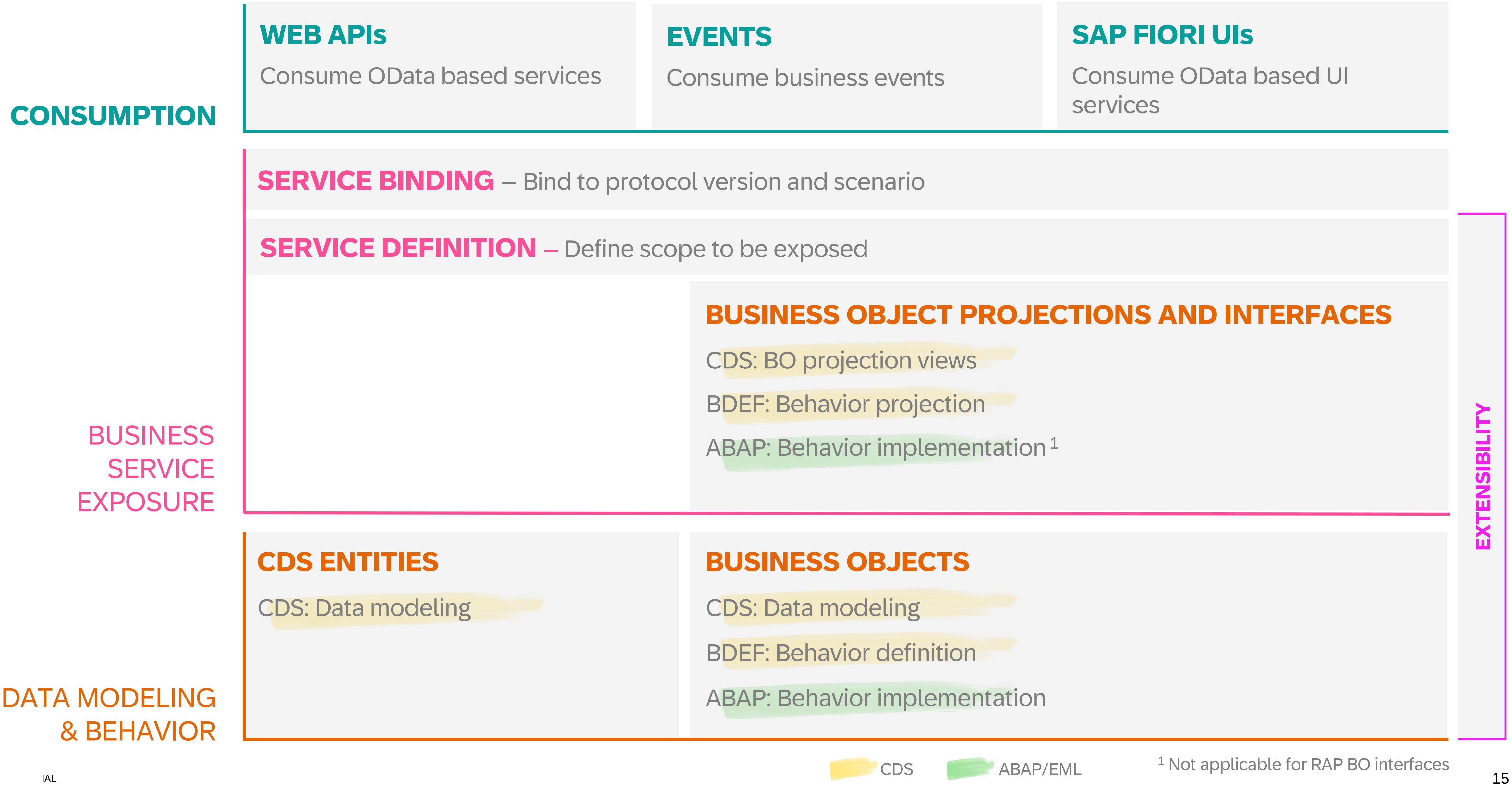
Service Definition: which entities?

BO Projection View: which elements?

Behavior Projection: which behavior?

** planned*

ABAP RESTful Application Programming Model (RAP) the big picture



CONSUMPTION

WEB APIs

Consume OData based services

EVENTS

Consume business events

SAP FIORI UIs

Consume OData based UI services

BUSINESS SERVICE EXPOSURE

SERVICE BINDING – Bind to protocol version and scenario

SERVICE DEFINITION – Define scope to be exposed

BUSINESS OBJECT PROJECTIONS AND INTERFACES

- CDS: BO projection views
- BDEF: Behavior projection
- ABAP: Behavior implementation [1]

DATA MODELING & BEHAVIOR

CDS ENTITIES

- CDS: Data modeling

BUSINESS OBJECTS

- CDS: Data modeling
- BDEF: Behavior definition
- ABAP: Behavior implementation

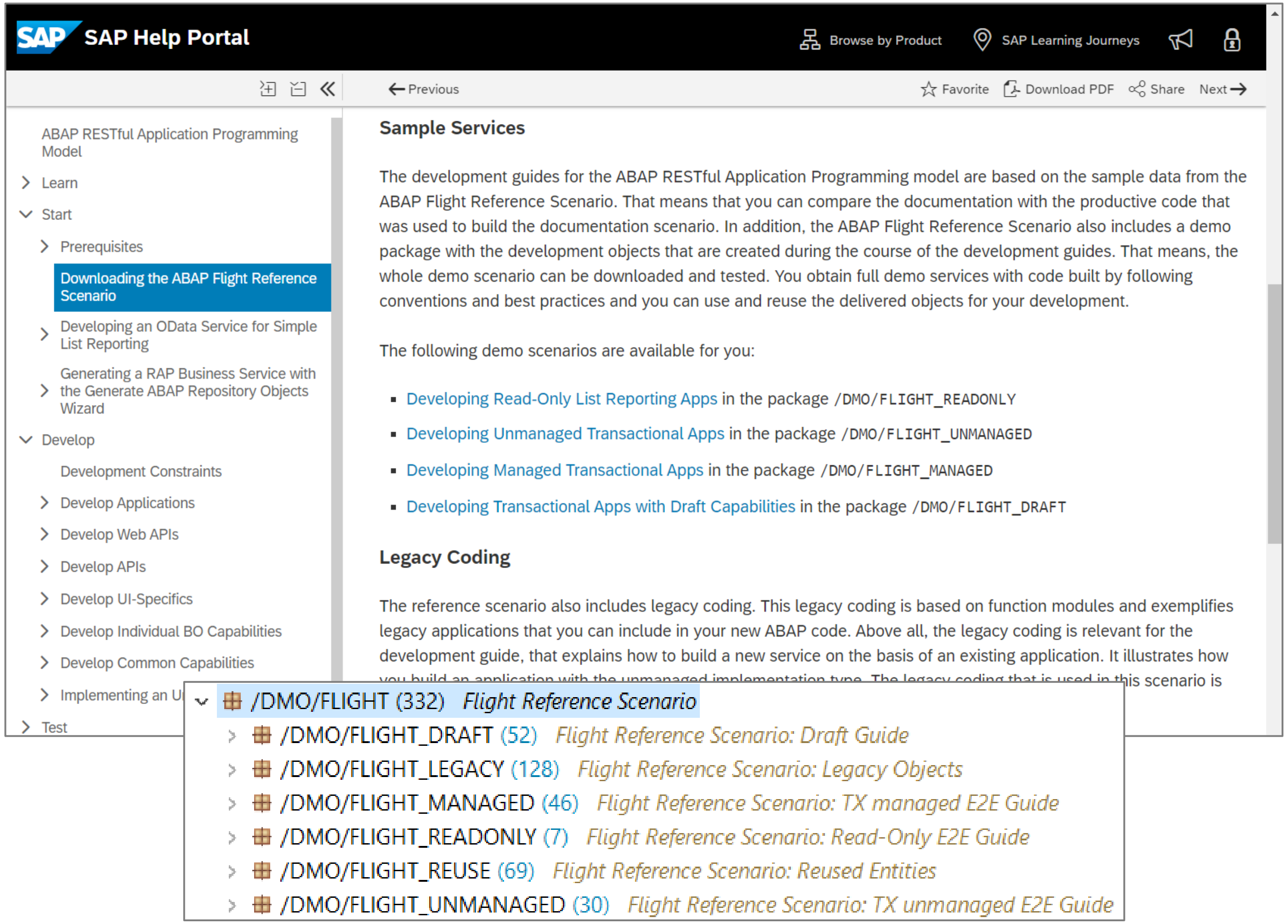
EXTENSIBILITY

🟨 CDS 🟩 ABAP/EML

[1] Not applicable for RAP BO interfaces

Key Takeaways

Sample implementations with the ABAP Flight Reference Scenario


SAP Help Portal
Browse by Product
SAP Learning Journeys
Previous
Favorite
Download PDF
Share
Next
ABAP RESTful Application Programming Model
Learn
Start
Prerequisites
Downloading the ABAP Flight Reference Scenario
Developing an OData Service for Simple List Reporting
Generating a RAP Business Service with the Generate ABAP Repository Objects Wizard
Develop
Development Constraints
Develop Applications
Develop Web APIs
Develop APIs
Develop UI-Specifics
Develop Individual BO Capabilities
Develop Common Capabilities
Implementing an U
Test

Sample Services

The development guides for the ABAP RESTful Application Programming model are based on the sample data from the ABAP Flight Reference Scenario. That means that you can compare the documentation with the productive code that was used to build the documentation scenario. In addition, the ABAP Flight Reference Scenario also includes a demo package with the development objects that are created during the course of the development guides. That means, the whole demo scenario can be downloaded and tested. You obtain full demo services with code built by following conventions and best practices and you can use and reuse the delivered objects for your development.

The following demo scenarios are available for you:

- Developing Read-Only List Reporting Apps in the package `/DMO/FLIGHT_READONLY`
- Developing Unmanaged Transactional Apps in the package `/DMO/FLIGHT_UNMANAGED`
- Developing Managed Transactional Apps in the package `/DMO/FLIGHT_MANAGED`
- Developing Transactional Apps with Draft Capabilities in the package `/DMO/FLIGHT_DRAFT`

Legacy Coding

The reference scenario also includes legacy coding. This legacy coding is based on function modules and exemplifies legacy applications that you can include in your new ABAP code. Above all, the legacy coding is relevant for the development guide, that explains how to build a new service on the basis of an existing application. It illustrates how you build an application with the unmanaged implementation type. The legacy coding that is used in this scenario is

```
/DMO/FLIGHT (332)  Flight Reference Scenario
  /DMO/FLIGHT_DRAFT (52)  Flight Reference Scenario: Draft Guide
  /DMO/FLIGHT_LEGACY (128)  Flight Reference Scenario: Legacy Objects
  /DMO/FLIGHT_MANAGED (46)  Flight Reference Scenario: TX managed E2E Guide
  /DMO/FLIGHT_READONLY (7)  Flight Reference Scenario: Read-Only E2E Guide
  /DMO/FLIGHT_REUSE (69)  Flight Reference Scenario: Reused Entities
  /DMO/FLIGHT_UNMANAGED (30)  Flight Reference Scenario: TX unmanaged E2E Guide
```

Demonstrate how to use different RAP capabilities concretely

Based on a simple to use data model: SFLIGHT reloaded

Feature scope regularly enhanced

Downloadable from GitHub

Read more in the RAP documentation: Cloud | SAP S/4HANA

Key Takeaways

The **ABAP Cloud** development model enables you to build cloud-ready, enterprise-grade business apps, services or extensions in the cloud and on-premise.

The **ABAP RESTful Application Programming Model (RAP)** is the center piece for building efficiently transactional services.

RAP best supports **SAP HANA** and **SAP Fiori elements**.

What's New: [SAP BTP ABAP Environment]() | [SAP S/4HANA]() | [SAP S/4HANA Cloud]()

What's Next: [interactive SAP Road Map Explorer]()

Thank you.

Contact information:

Pedro Sangaletti

pedro.sangaletti@sap.com