

# ODATA SERVICE DEVELOPMENT OPTIONS

**Be prepared for the  
ABAP RESTful Application  
Programming Model**

Andre Fischer  
January, 2022



# Agenda



What is RAP?



ODATA V2 DEVELOPMENT OPTIONS in SAP Business Suite

- Managed implementations
- Unmanaged implementations
- Unmanaged implementations (without existing CDS views)



Summary / Outlook / Further information



**What is RAP ?**

# Evolution of the ABAP programming model

ABAP PLATFORM  $\leq 7.5$

CLASSIC ABAP  
PROGRAMMING

Best practice freestyle ABAP  
programming, (Web) Dynpro

ABAP PLATFORM  $\geq 7.5$

ABAP  
PROGRAMMING  
MODEL FOR  
SAP FIORI\*

CDS, CDS-based BOPF,  
SEGW / @OData annotation  
with Referenced Data Source,  
SAP Fiori, UI5

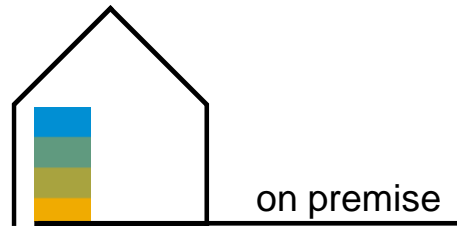
SAP BTP  
ABAP Environment  
SAP S/4HANA  $\geq 1909$

ABAP RESTful  
APPLICATION  
PROGRAMMING  
MODEL (RAP)

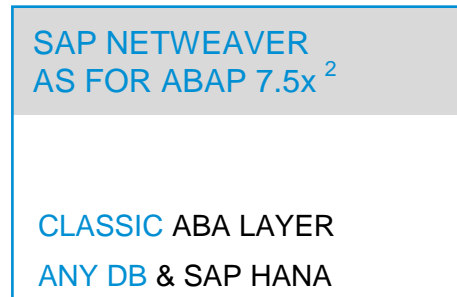
CDS, Behavior Definition &  
Implementation, Business  
Services, SAP Fiori, UI5

★ Safe investments!

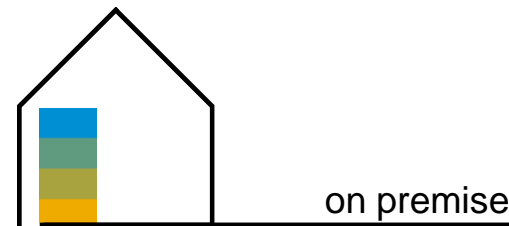
# ABAP Platform – Versions and consumers



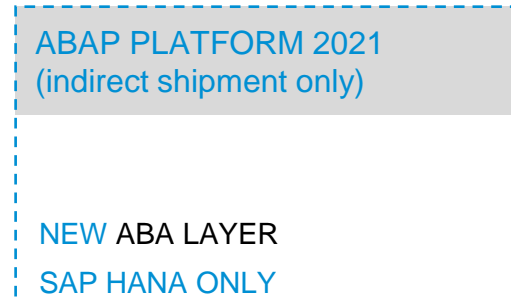
SAP Business Suite<sup>1</sup>  
 SAP NetWeaver hubs  
 SAP NetWeaver add-ons  
 Custom applications



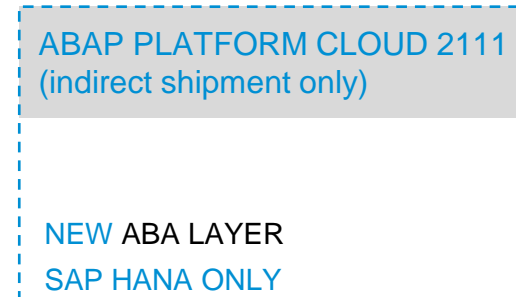
<sup>1</sup> (7.50) <sup>2</sup> (7.50, 7.51 and 7.52)



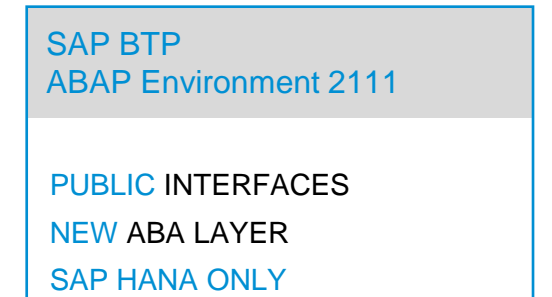
SAP S/4HANA On-Premise



SAP S/4HANA Cloud



ABAP-based SaaS solutions  
 Extensions for SAP S/4HANA  
 Custom cloud apps



# RAP – The big picture for service consumption scenarios

## CONSUMPTION

### DATA INTEGRATION\*

Consume SQL based services

### WEB APIs

Consume OData based services

### SAP FIORI UIs

Consume OData based UI services

### SAP ANALYTICS CLOUD\*

Consume InA based UI services for live data access



**SERVICE BINDING** – Bind to protocol version and scenario



**SERVICE DEFINITION** – Define scope to be exposed

## BUSINESS SERVICE PROVISIONING

### BUSINESS OBJECT PROJECTION



CDS: BO projection views



BDEF: Behavior projection



ABAP: Behavior implementation

### ANALYTICAL PROJECTION



CDS: Analytical projection views

## DATA MODELING & BEHAVIOR

### CDS ENTITIES



CDS: Data modeling

### BUSINESS OBJECTS



CDS: Data modeling



BDEF: Behavior definition



ABAP: Behavior implementation

### ANALYTICAL MODEL



CDS: Analytical cubes



CDS: Analytical dimensions

# CDS views are used to define business objects

```
define root view /DMO/I_Travel_U
  as select from /dmo/travel as Travel
  -- the travel table is the data source for this view

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

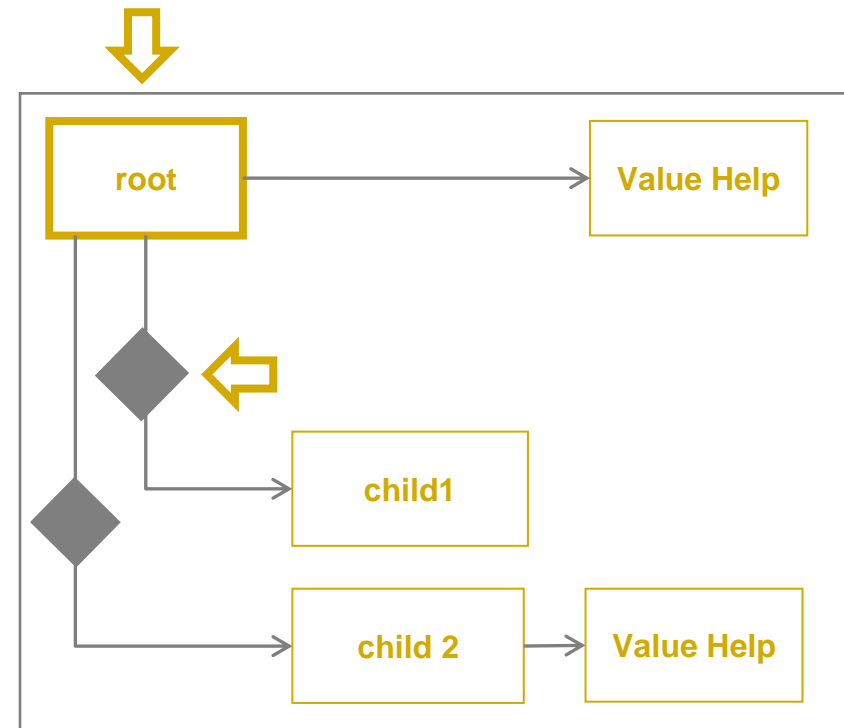
  association [0..1] to /DMO/I_Agency as _Agency
    on $projection.AgencyID = _Agency.AgencyID

  association [0..1] to /DMO/I_Customer as _Customer
    on $projection.CustomerID = _Customer.CustomerID

  association [0..1] to I_Currency as _Currency
    on $projection.CurrencyCode = _Currency.Currency
```

```
define view /DMO/I_Booking_U
  as select from /dmo/booking as Booking

  association to parent /DMO/I_Travel_U as _Travel
    on $projection.TravelID = _Travel.TravelID
```



# Behaviour definition language

```
implementation unmanaged;
```

```
// behavior definition for the TRAVEL root entity  
define behavior for /DMO/I_Travel_U alias travel  
implementation in class /DMO/BP_TRAVEL_U unique  
etag LastChangedAt
```

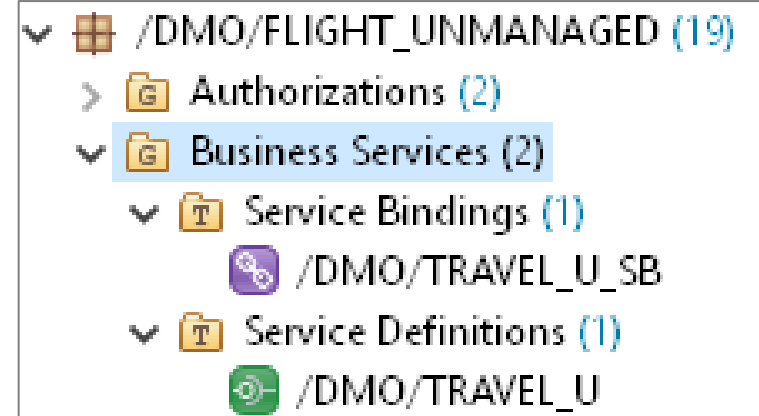
```
{  
  field (read only) TravelID;  
  field (mandatory) AgencyID, CustomerID,  
  BeginDate, EndDate;  
  
  create;  
  update;  
  delete;  
  
  action set_status_booked result [1] $self;  
  
  association _Booking { create; }  
}
```

The screenshot displays a SAP Fiori application interface. At the top, there is a search bar and filter fields for 'Travel ID', 'Agency ID', and 'Customer ID'. Below this is a table titled 'Travels (4,136)' with columns 'Travel ID' and 'Agency ID'. The table contains two rows: one with '1' and 'Maxitrip (7004)' and another with '2' and 'Hot Socks Tra'. To the right of the table are buttons for 'Set to Booked', 'Delete', and a plus sign. A modal window is open over the table, showing a form for a 'Travel' object. The form has a title '<Unnamed Object>' and a 'Travel' section. The 'Travel ID' field is highlighted with a yellow box. Other fields include 'Agency ID', 'Customer ID', 'Starting Date', 'End Date', 'Booking Fee', 'Total Price', and 'Comment'.



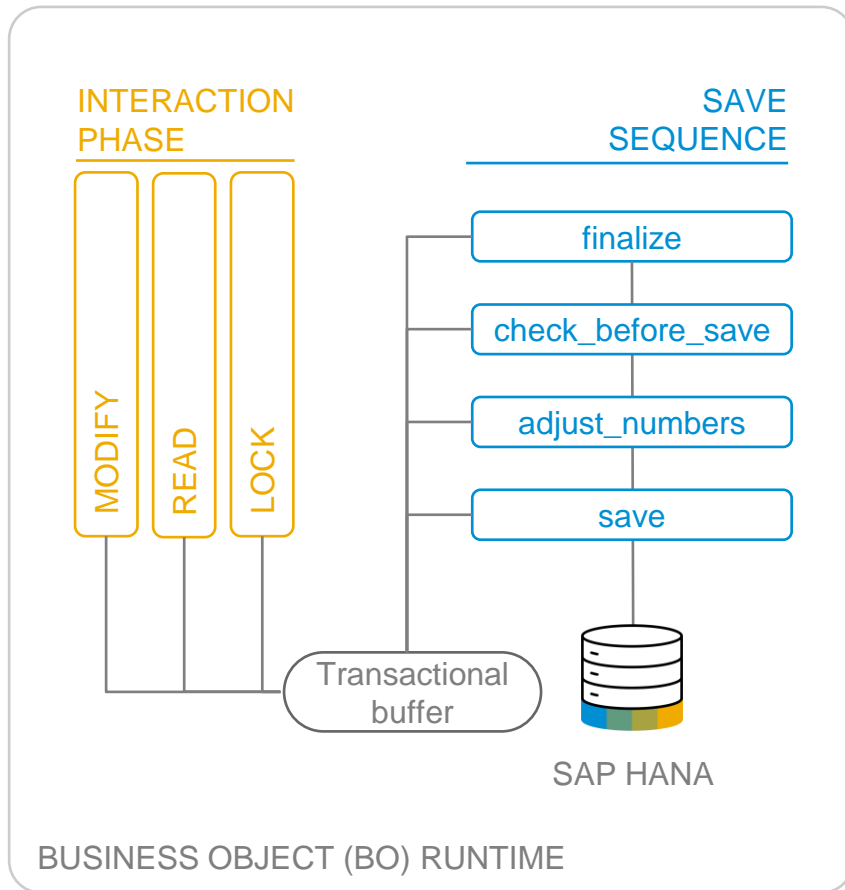
# Service Definition

```
@EndUserText.label: 'Service Definition for Managing Travels'  
define service /DMO/TRAVEL_U {  
  expose /DMO/I_Travel_U as Travel;  
  expose /DMO/I_Booking_U as Booking;  
  expose /DMO/I_BookingSupplement_U as BookingSupplement;  
  expose /DMO/I_Supplement as Supplement;  
  expose /DMO/I_SupplementText as SupplementText;  
  expose /DMO/I_Customer as Passenger;  
  expose /DMO/I_Agency as TravelAgency;  
  expose I_Currency as Currency;  
  expose I_Country as Country;  
  expose /DMO/I_Carrier as Airline;  
  expose /DMO/I_Connection as FlightConnection;  
  expose /DMO/I_Flight as Flight;  
}
```





# RAP BO runtime implementation types – Overview



## UNMANAGED

Brownfield development with application coding fully available:  
Interaction phase + Transactional buffer + Save sequence

## MANAGED

Greenfield development with standard implementation  
(opt. unmanaged appl. components: DB tables, lock/PFCG object, update task FM)

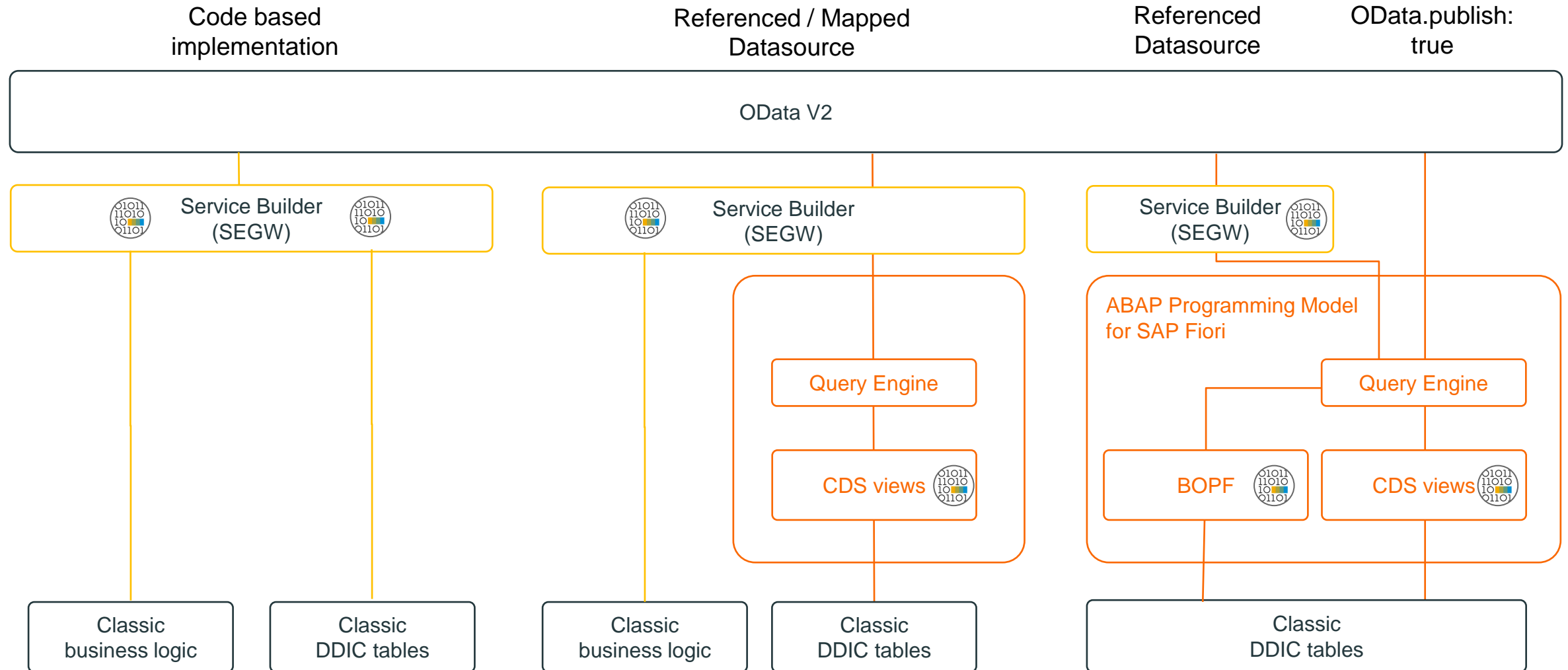


**OData**

**development options**

**SAP Business Suite**

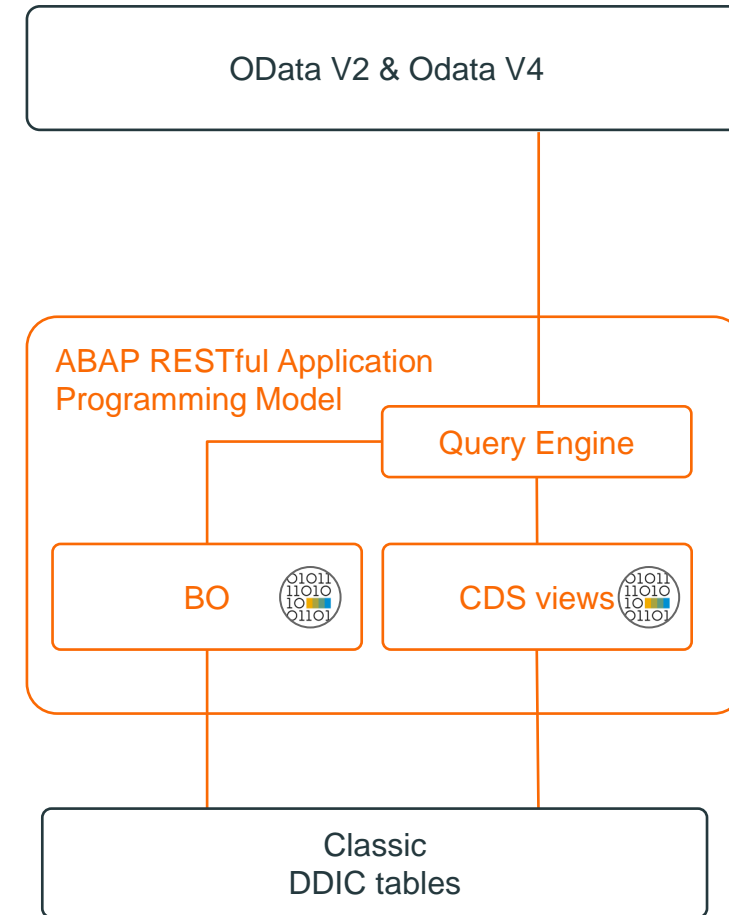
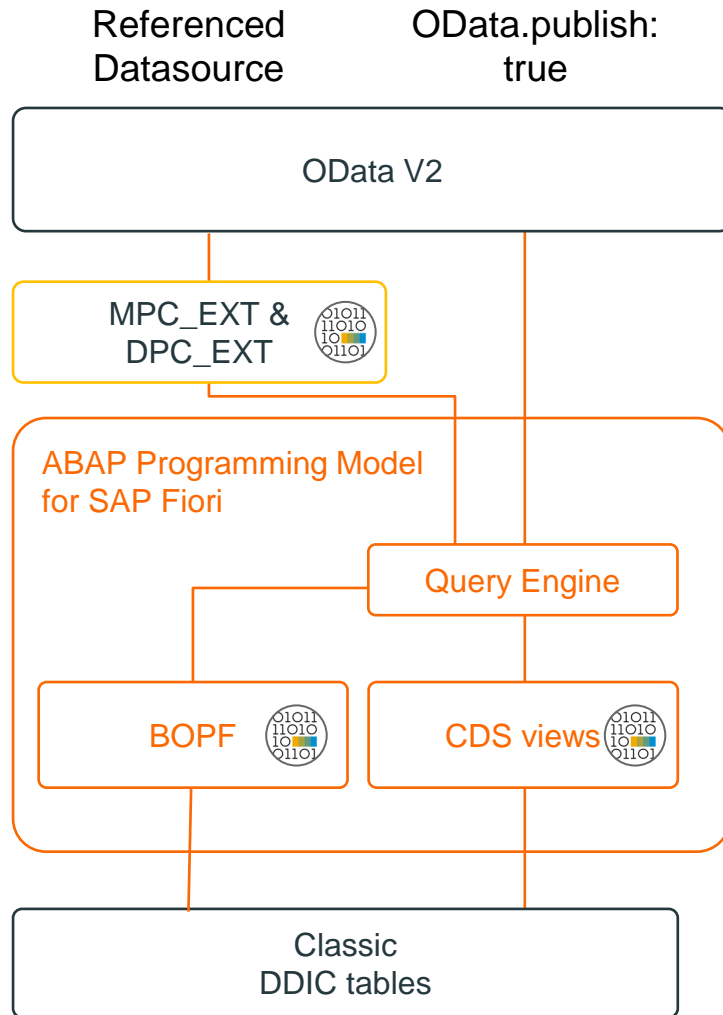
# OData V2 service development options in SAP Business Suite





# Managed implementations

# ABAP Programming Model for SAP Fiori vs. RAP



= code based implementation (ABAP, BOPF or CDS)

# ABAP Programming Model for SAP Fiori – CDS based BOPF Objects

Determination Overview

General Information of Determination

Name: CREATE\_SO\_ID

Description: Create Sales Order ID

Implementation Class: ZCL\_S4H\_D\_CREATE\_SO\_ID\_999

Category: React after modification

Triggers

Node	Association	Create	Update	Delete	Load	Determine
ZS4H_I_SLSORDER_TP_999		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

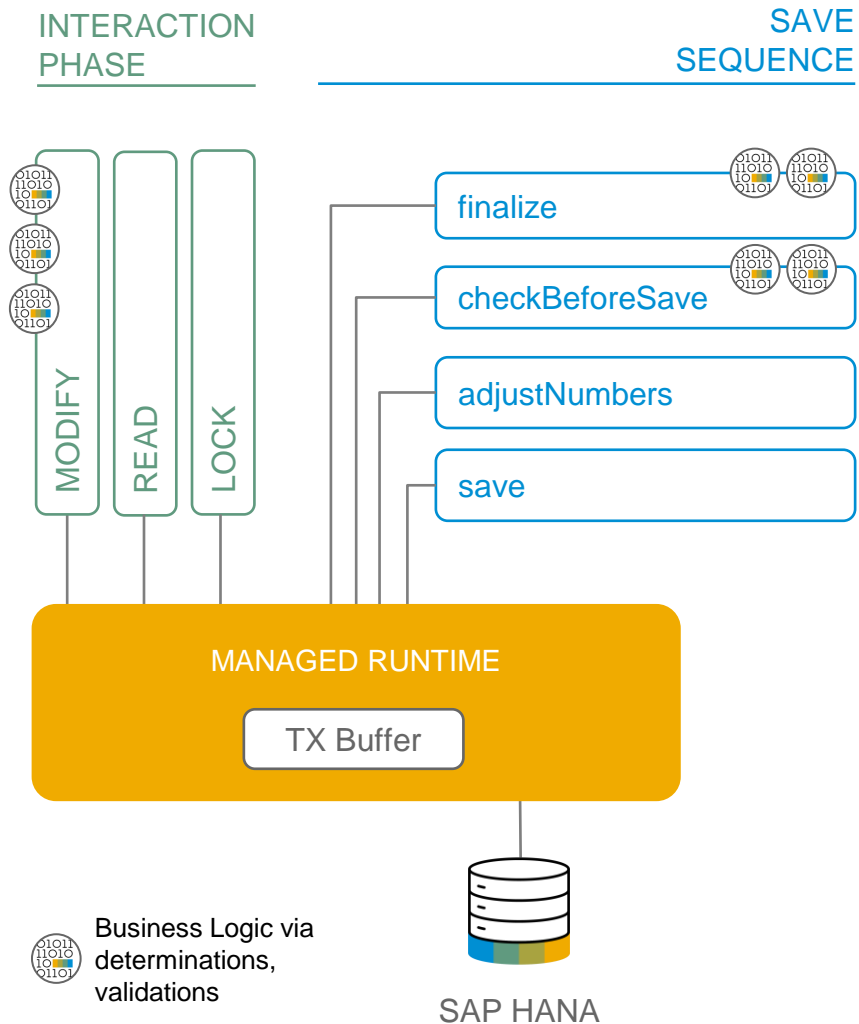


```
[A4H] ZS4H_I_SLSORDER_TP_999 [A4H] ZCL_S4H_D_CREATE_SO_ID_999
ZCL_S4H_D_CREATE_SO_ID_999 /BOBF/IF_FRW_DETERMINATION~EXECUTE
1 CLASS zcl_s4h_d_create_so_id_999 DEFINITION
2 PUBLIC
3 INHERITING FROM /bobf/cl_lib_d_supercl_simple
4 FINAL
5 CREATE PUBLIC .
6
7 PUBLIC SECTION.
8
9     METHODS /bobf/if_frw_determination~execute
10     REDEFINITION .
11 PROTECTED SECTION.
12 PRIVATE SECTION.
13 ENDClass.
14
15
16 CLASS zcl_s4h_d_create_so_id_999 IMPLEMENTATION.
17
18
19
20 METHOD /bobf/if_frw_determination~execute.
21 ENDMETHOD.
22 ENDClass.
```

- ▶ OData Model definition via CDS views
- ▶ Available as of 7.50 (Draft as of SAP S/4HANA 1610)
- ▶ Code based implementation of Determinations, Validations and Actions
- ▶ Managed scenario: Query engine (SADL) orchestrates all CRUD-Q calls to the SAP Gateway (OData) framework



# RAP - BO runtime implementation type - managed



## Application coding

- ▶ not yet available or fine granular reusable code available
- ▶ technical implementation tasks taken over by BO infrastructure
- ▶ developer focus on business logic, implemented via code exits: determinations, validation, actions,...

## Examples

- ▶ New applications in SAP Cloud Platform ABAP Environment

## Availability

- ▶ On premise as of SAP S/4HANA 2020

# Recommendations for managed scenarios

## Existing Implementations

- ▶ Will continue to run in SAP S/4HANA
- ▶ Planned: Reuse of CDS-based BOPF BOs in RAP managed BOs

## New Implementations

- ▶ SAP S/4HANA 2020 FSP1
  - ▶ Support for draft
  - ▶ Major limitations regarding key layout and numbering
  - ▶ If Limitations are crucial →  
Leverage RAP features that are not available in *ABAP Programming Model for SAP Fiori*

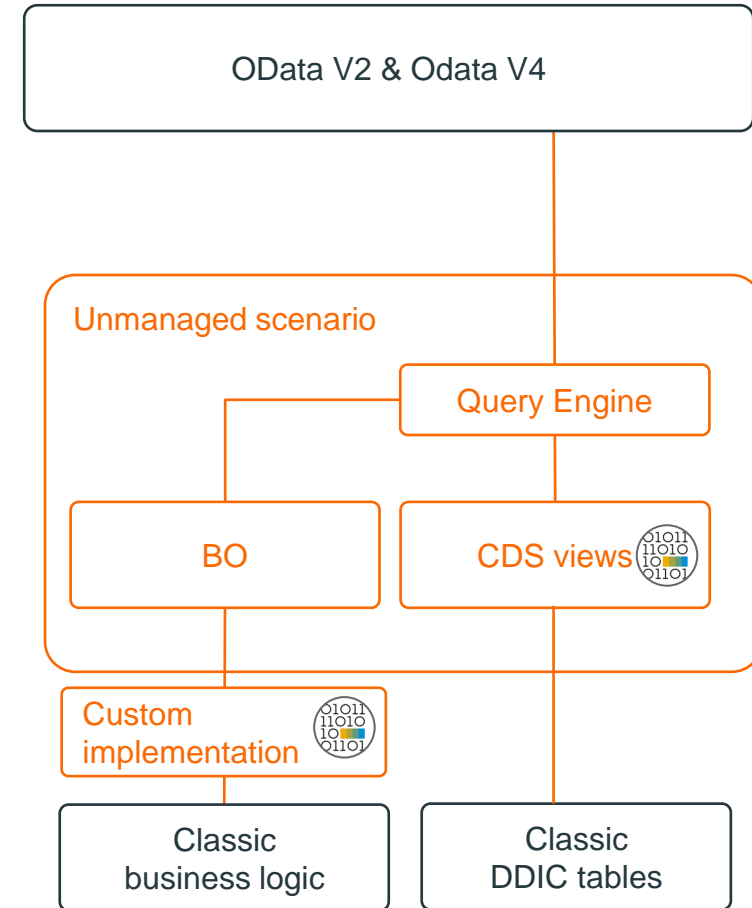
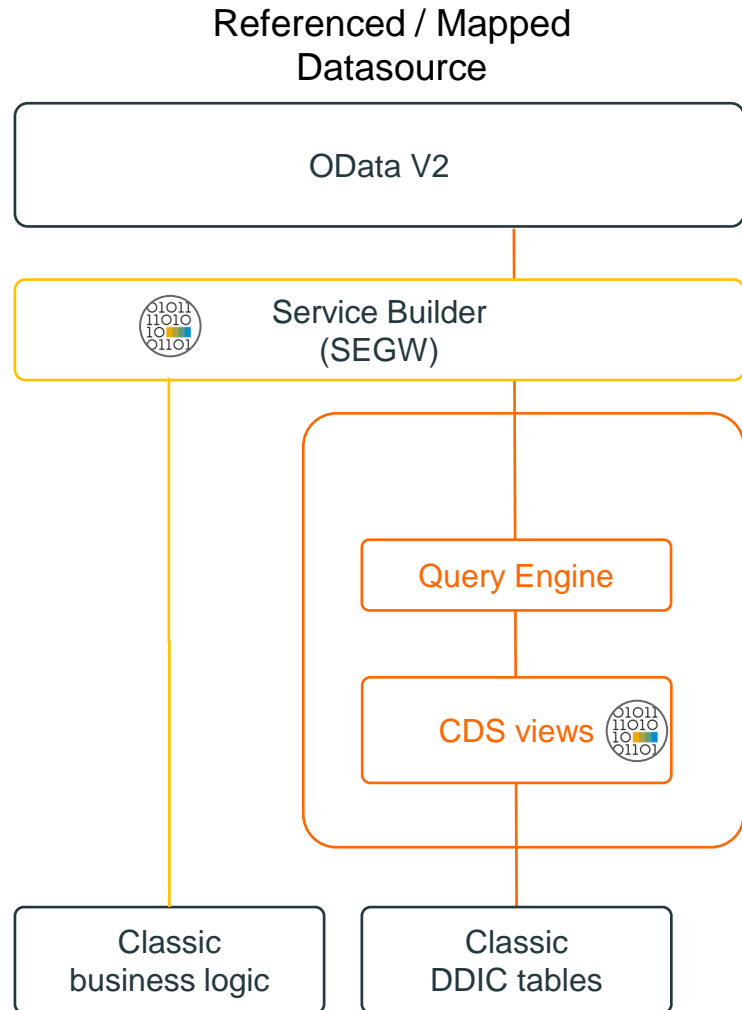
SAP S/4HANA 2021 FSP1 → Guidance: Use RAP

- ▶ Full support of all key types



# Unmanaged implementations

# Referenced / Mapped Datasource vs. RAP



= code based implementation (ABAP, BOPF or CDS)

# Referenced Data Source

The screenshot displays the SAP Gateway Service Builder interface. On the left, a tree view shows the project structure for 'Z\_BE2UI\_SOL', with 'Data Model' expanded to 'CDS-Entity Exposures'. The right pane shows a list of 'CDS-Entity Exposures' with checkboxes for selection and a 'Value Help' column.

CDS-Entity Exposures	Selected	Value Help
SEPMRA_I_ProductMainCategory	<input checked="" type="checkbox"/>	✓
_Category	<input type="checkbox"/>	
SEPMRA_I_ProductCategory		
ZC_Be2Ui_Product_Ex4	<input checked="" type="checkbox"/>	
_Currency	<input checked="" type="checkbox"/>	
I_Currency	<input checked="" type="checkbox"/>	
_ProductCategory	<input checked="" type="checkbox"/>	
SEPMRA_I_ProductCategory	<input checked="" type="checkbox"/>	
_ProductReview	<input checked="" type="checkbox"/>	
ZC_Be2Ui_ProductReview_Ex4	<input checked="" type="checkbox"/>	
_Supplier	<input checked="" type="checkbox"/>	
SEPMRA_I_Supplier	<input checked="" type="checkbox"/>	

- OData Model references selected CDS views and associations
- Available as of 7.50
- New fields are automatically included
- Code based implementation of CREATE, UPDATE, and DELETE methods

# Mapped Data Source

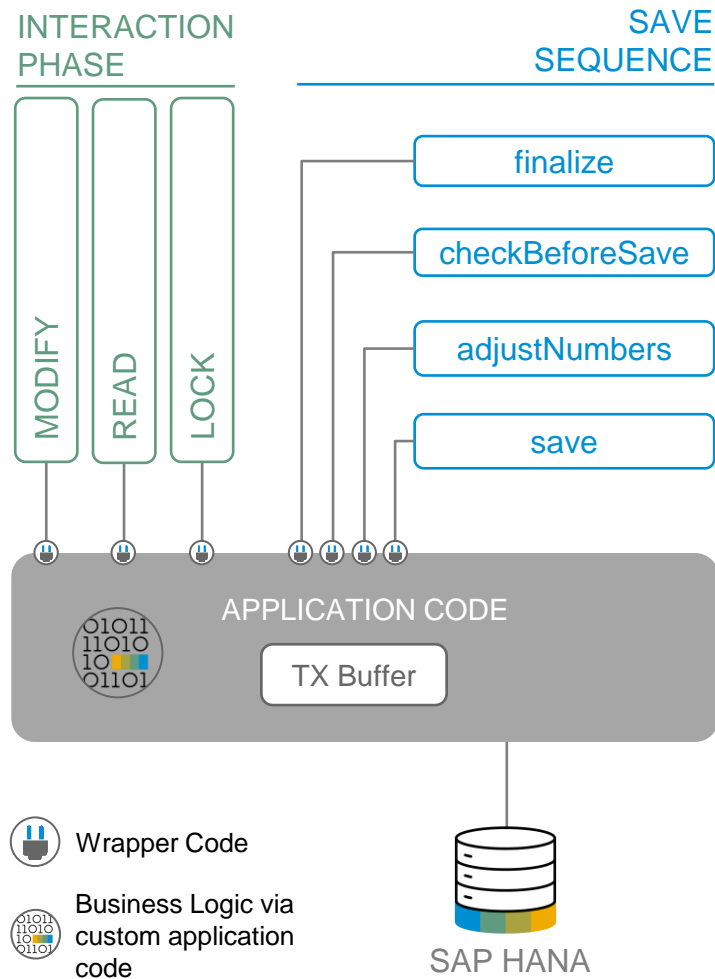
The screenshot displays the SAP Gateway Service Builder interface. On the left, a tree view shows the project structure for 'ZCDS\_MDS', including 'Data Model', 'Service Implementation', and 'Runtime Artifacts'. The 'Service Implementation' folder is expanded to show 'BusinessPartnerSet' with a 'Mapping' sub-item. The main workspace is titled 'Map Modeled Data Source Elements to Properties' and contains a table with the following data:

Property	ABAP Field Name	Key	Element
Businesspar...	BUSINESSPARTNERUUID	<input checked="" type="checkbox"/>	BUSINESSPARTNERUUID
Businesspar...	BUSINESSPARTNERID	<input type="checkbox"/>	BUSINESSPARTNERID
Companyname...	COMPANYNAME	<input type="checkbox"/>	COMPANYNAME

On the right side, a 'Modeled Data Source Name' table is visible, showing 'ZCDS\_BP\_EXAMPLE' with its properties: 'BUSINESSPARTNERUUID' (EPM: Ge), 'BUSINESSPARTNERID' (EPM: Bu), 'COMPANYNAME' (EPM: Co), and a 'Primary Key' indicator.

- OData Model has to be modelled in SEGW
- Manual mapping of READ and QUERY methods
- Available as of 7.40
- New fields are not automatically included
- Code based implementation of CREATE, UPDATE, and DELETE methods

# BO runtime implementation type - unmanaged



## Application coding

- ▶ already available
- ▶ for interaction phase, transactional buffer and save sequence
- ▶ decoupled from UI technology

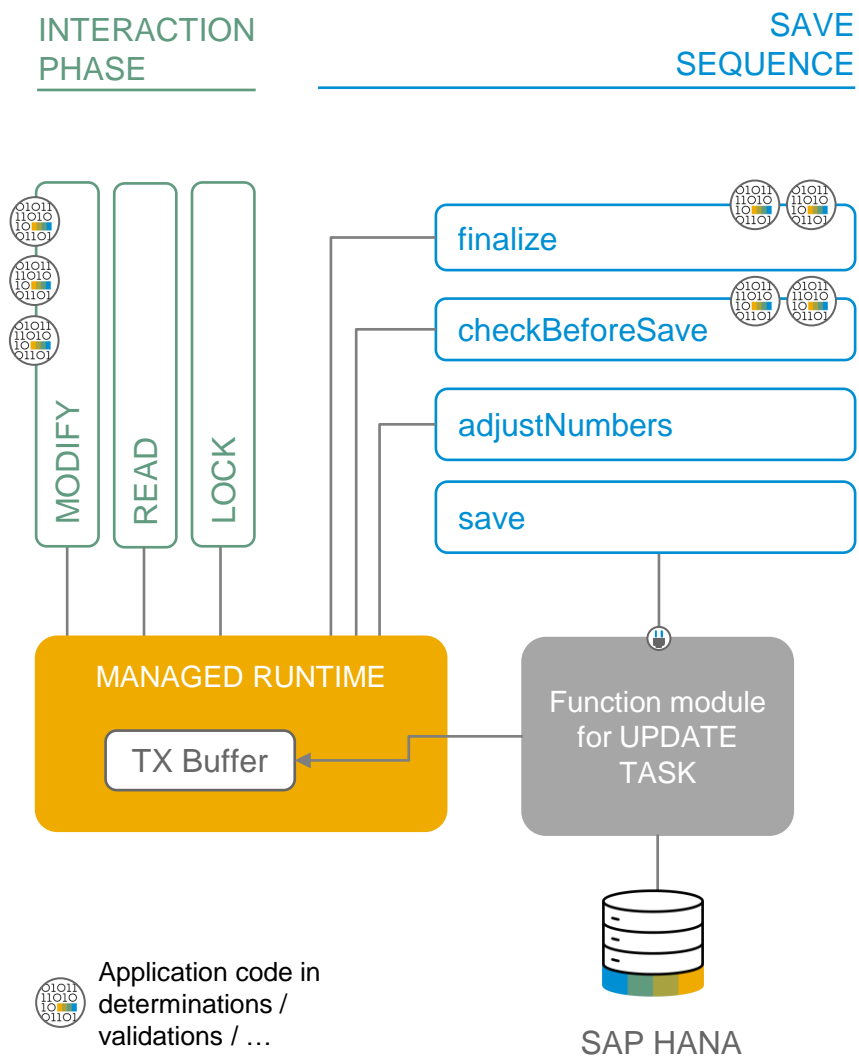
## Examples

- ▶ Sales Order, Purchase Order

## Availability

- ▶ On premise as of SAP S/4HANA 1909

# BO runtime implementation type – managed with unmanaged save



## Application coding

- ▶ “update-task function module” available
- ▶ coding for interaction phase not available (e.g. highly coupled in older UI technology: DYNP - PBO / PAI)
- ▶ technical implementation aspects to be taken over by BO infrastructure

## Examples

- ▶ Business Partner, Product

## Availability

- ▶ On premise as of SAP S/4HANA 2020



# Recommendations for unmanaged scenarios with CDS views

## Existing Implementations

- ▶ Will continue to run in SAP S/4HANA

## New Implementations

- ▶ SAP S/4HANA 1909
  - ▶ Several limitations (e.g. No filter on projection layer)
  - ▶ If limitations are crucial → Leverage Reference Data Source approach

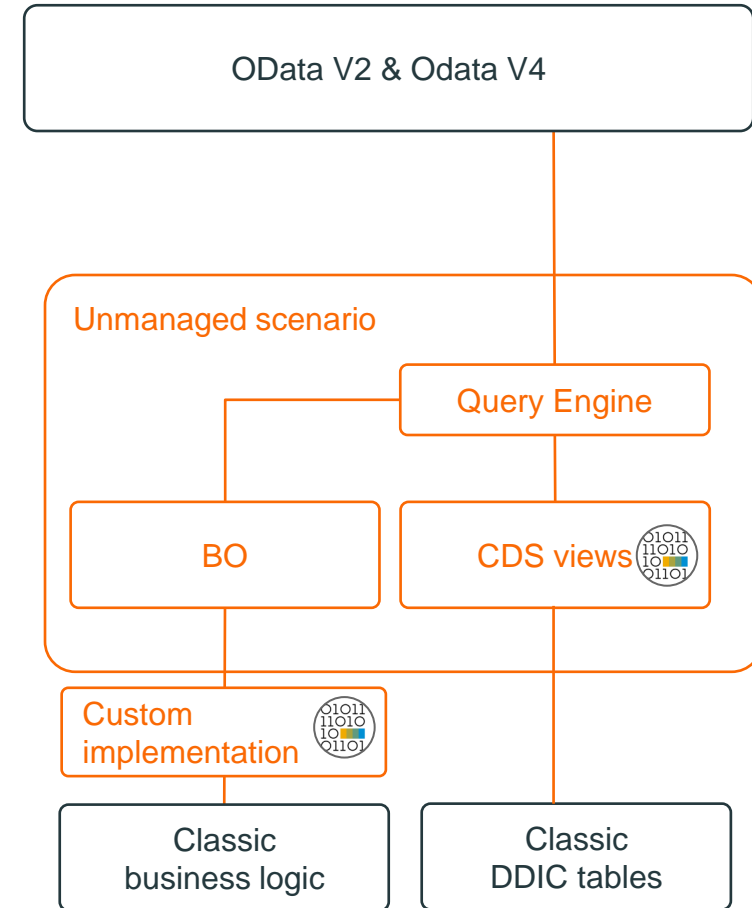
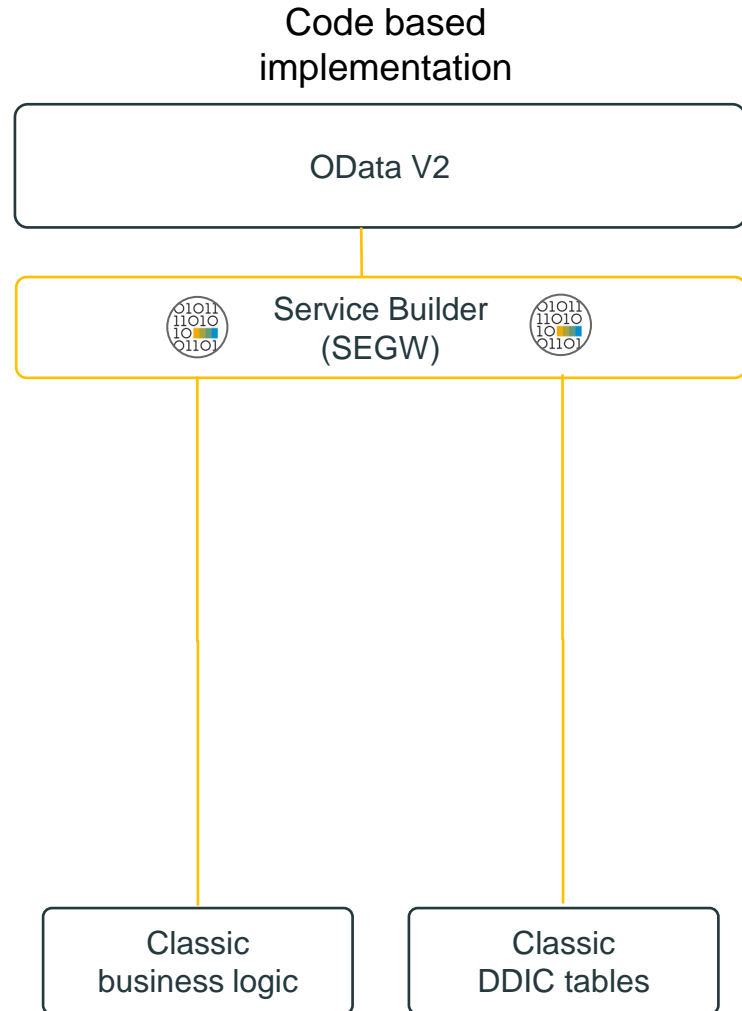
SAP S/4HANA 2020 FSP1 → Guidance: Use RAP

- ▶ Support of draft
- ▶ Support of OData V4



**Unmanaged  
implementations  
(without existing  
CDS Views )**

# SEGW code based implementation vs. RAP



# SAP Gateway

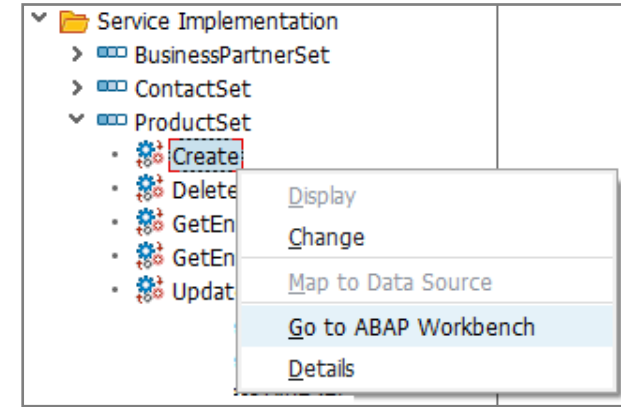
## Code-based service development

### CODE-BASED IMPLEMENTATION OF CRUD METHODS

- ▶ Create → <Entity\_Set>\_CREATE\_ENTITY
- ▶ Read → <Entity\_Set>\_GET\_ENTITY
- ▶ Query → <Entity\_Set>\_GET\_ENTITYSET
- ▶ Update → <Entity\_Set>\_UPDATE\_ENTITY
- ▶ Delete → <Entity\_Set>\_DELETE\_ENTITY

### CODE BASED IMPLEMENTATION OF ADVANCED ODATA FEATURES

- ▶ Offline scenarios (\$deltatoken, \$skiptoken)
- ▶ Complex transactional behavior  
(SoftState, DeepInsert, ChangeSets in \$batch)



```
METHOD productset_get_entityset.  
  
DATA: lv_source_entity_set_name TYPE /iwbp/mgw_tech_name.  
  
lv_source_entity_set_name = io_tech_request_context->get_source_entity_set_name( ).  
  
IF lv_source_entity_set_name IS INITIAL.  
  prod_get_entityset(  
    EXPORTING  
      io_tech_request_context = io_tech_request_context  
    IMPORTING  
      et_entityset           = et_entityset  
      es_response_context   = es_response_context ).  
ENDIF.
```

# Recommendations for unmanaged scenarios without existing CDS views

## Existing Implementations

- ▶ Will continue to run in SAP S/4HANA

## New Implementations

- ▶ Guidance: Build CDS views
- ▶ SAP S/4HANA 2020 FSP1
  - ▶ Support of draft
  - ▶ Support of OData V4



**Summary /  
Outlook /  
Further information**

# RAP vs. SEGW

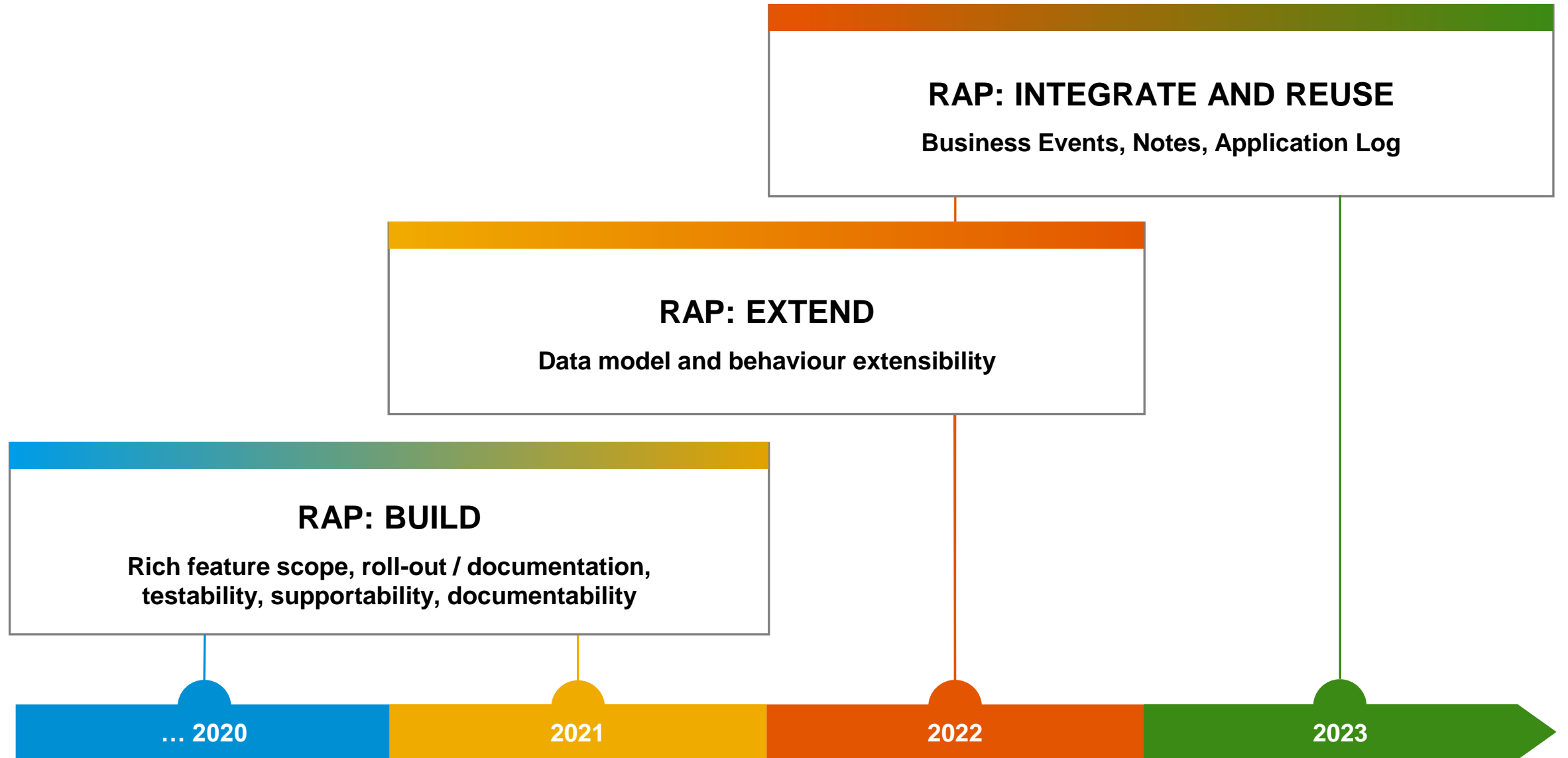
SCENARIO	USE-CASE	SAP Business Suite	ABAP RESTFUL APPLICATION PROGRAMMING MODEL
managed	Greenfield implementation	SEGW with BOPF (ABAP Programming Model for SAP Fiori)	✓ (SAP S/4HANA 2020)
managed with self-implemented save	Brownfield implementation	SEGW with RDS	✓ (SAP S/4HANA 2020)
un-managed	Brownfield implementation	SEGW code based	✓ (SAP S/4HANA 2019)

## MANAGED SCENARIO

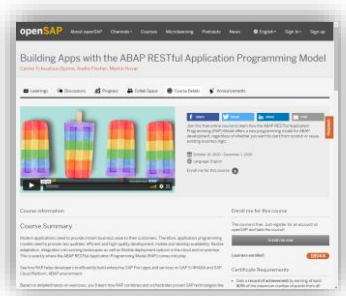
Query engine (SADL) orchestrates all CRUD-Q calls to the SAP Gateway (OData) framework

✓ = supported

# Outlook – Next steps







# FREE openSAP COURSE



## Building Apps with the ABAP RESTful Application Programming Model (RAP)



### Self-paced mode



- Week 1: Introduction
- Week 2: Developing a Read-Only List Report App
- Week 3: Enabling the Transactional Behavior of an App
- Week 4: Dealing with Existing Code
- Week 5: Service Consumption and Web APIs
- Week 6: Final Exam



**ENROLL NOW!**

<https://open.sap.com/courses/cp13>



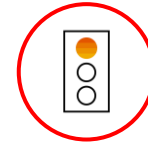
# Where to invest now to be prepared for the ABAP RESTful Application Programming Model?



## YOU SHOULD

Follow the programming model and [best practices](#) and use...

- Core Data Services (CDS) for database artefacts \*  
(≥ 7.40, [Documentation](#))
- CDS Metadata Extensions for UI Annotations \*  
(≥ 7.51 SP2, [Documentation](#))
- DCL for read/query instance-based authority checks \*  
(≥ 7.50, [Documentation](#))
- BOPF stand-alone  
(≤ 7.40, [Documentation](#), newer releases: CDS/BOPF integration)
- BOPF and CDS integration including draft \*  
(≥ 7.51 SP2, [Documentation](#))
- Gateway integration of CDS or BOPF  
(= 7.40, [Documentation](#), newer releases: OData Exposure)
- **OData Exposure of CDS / BOPF for SAP Fiori and future development \***  
(≥ 7.50 SP5, [Documentation](#))
- Floorplan-Manager integration of CDS and BOPF  
(≥ 7.40, [Documentation](#))



## DO NOT

Implement things already solved

- Manual implementation of **read-only calls** to DB
- Business logic mixed with **technical aspects**  
(e.g. locks, authority-check, LUW handling, persistency)
- Business logic mixed with **protocol specific APIs**  
(e.g. PBO/PAI, Gateway classes: DPC\_EXT)



## BENEFITS

Reuse / prepare your skillset and coding for the future

- Reuse CDS and DCL in SAP S/4HANA
- Lower TCD for the future: Minimal investment on technical protocol level



# FREE openSAP COURSE



## Building Apps with the ABAP RESTful Application Programming Model (RAP)



### Self-paced mode



- Week 1: Introduction
- Week 2: Developing a Read-Only List Report App
- Week 3: Enabling the Transactional Behavior of an App
- Week 4: Dealing with Existing Code
- Week 5: Service Consumption and Web APIs
- Week 6: Final Exam



**ENROLL NOW!**

<https://open.sap.com/courses/cp13>



# Thank you.

Contact information:

**Andre Fischer**

andre.fischer@sap.com

# Engage the community.sap.com

You are invited to network, collaborate and learn while enjoying these community offerings:



Questions and answers



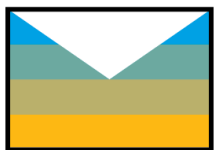
Blogging



Recognition Program



Personalization



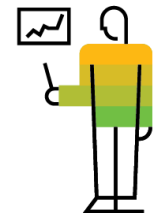
Direct Messaging



Coffee Corner



Events



Expert Pages