ODATA SERVICE DEVELOPMENT OPTIONS

Be prepared for the ABAP RESTful Application Programming Model

Andre Fischer
January, 2022

Agenda

What is RAP?

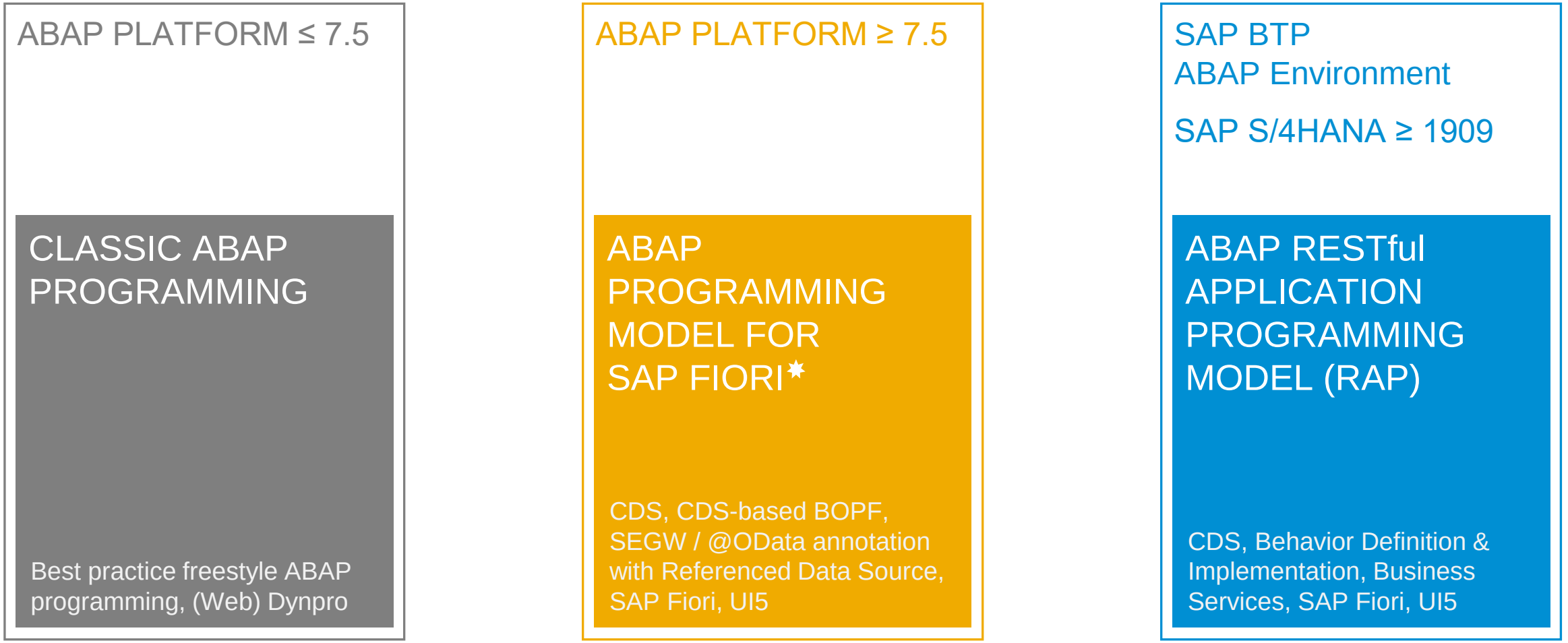
ODATA V2 DEVELOPMENT OPTIONS in SAP Business Suite

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

Summary / Outlook / Further information

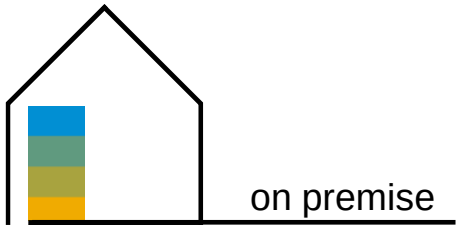
What is RAP ?

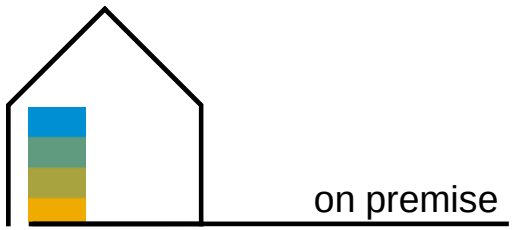
Evolution of the ABAP programming model



✸ Safe investments!

ABAP Platform – Versions and consumers





SAP Business Suite[1]
SAP NetWeaver hubs
SAP NetWeaver add-ons
Custom applications

SAP S/4HANA On-Premise

SAP S/4HANA Cloud

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

SAP NETWEAVER AS FOR ABAP 7.5x [2]

CLASSIC ABA LAYER
ANY DB & SAP HANA

ABAP PLATFORM 2021 (indirect shipment only)

NEW ABA LAYER
SAP HANA ONLY

ABAP PLATFORM CLOUD 2111 (indirect shipment only)

NEW ABA LAYER
SAP HANA ONLY

SAP BTP ABAP Environment 2111

PUBLIC INTERFACES
NEW ABA LAYER
SAP HANA ONLY

Improvements

One common ABAP Platform innovation codeline

[1] (7.50) [2] (7.50, 7.51 and 7.52)

This is the current state of planning and may be changed by SAP at any time.

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

BUSINESS SERVICE PROVISIONING

SERVICE BINDING – Bind to protocol version and scenario

SERVICE DEFINITION – Define scope to be exposed

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

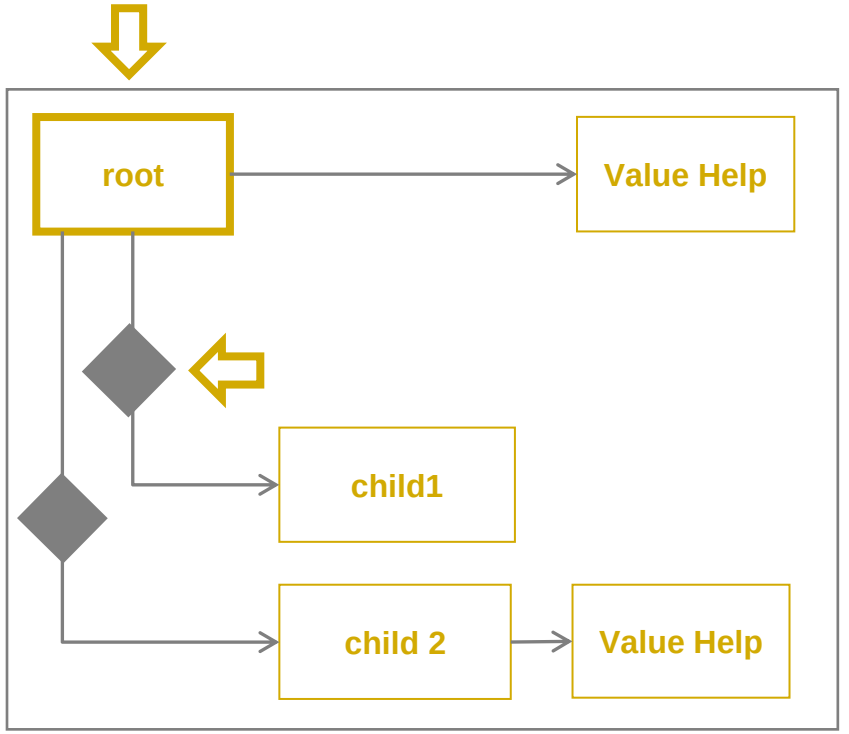
CDS: Analytical cubes


CDS: Analytical dimensions

* Scenarios currently only supported on SAP BTP ABAP Environment

CDS views are used to define business objects

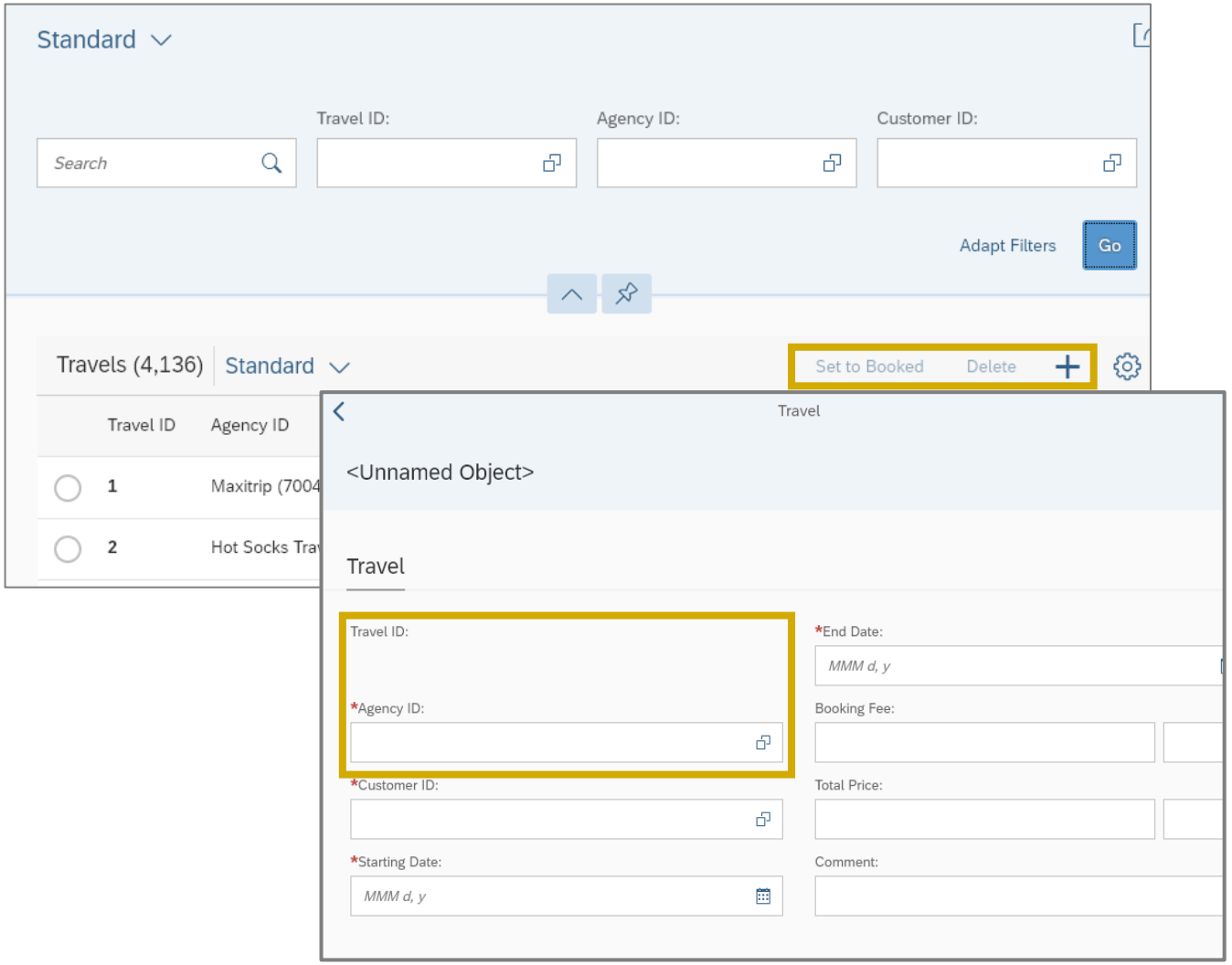
```
define root view /DMO/I_Travel_U
  as select from /dmo/travel as Travel
-- the travel table is the data source for this view

  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
```

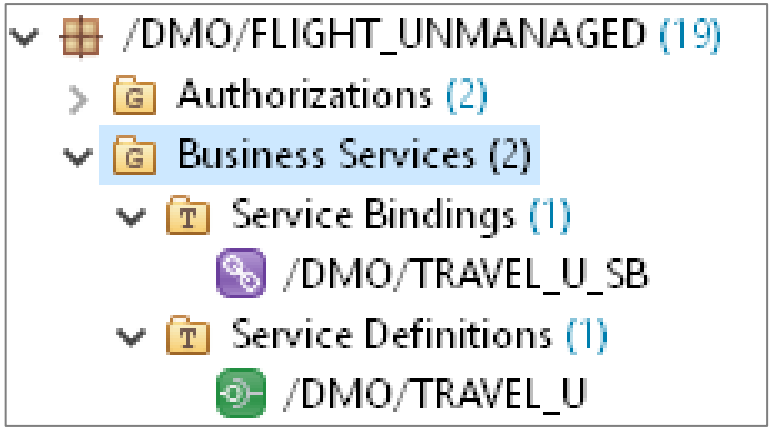
Behaviour definition language

```
implementation unmanaged;

// behavior defintion 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; }

}
```

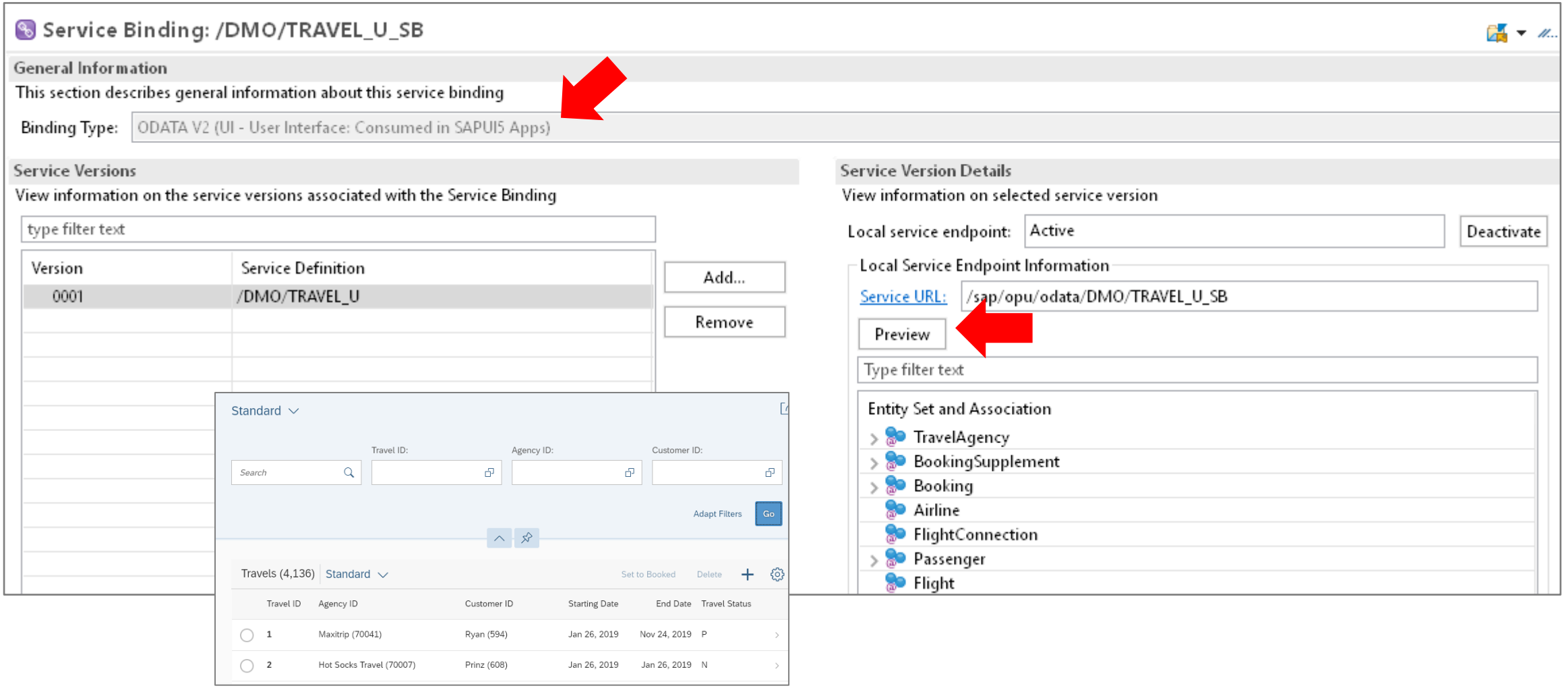


Service Definition

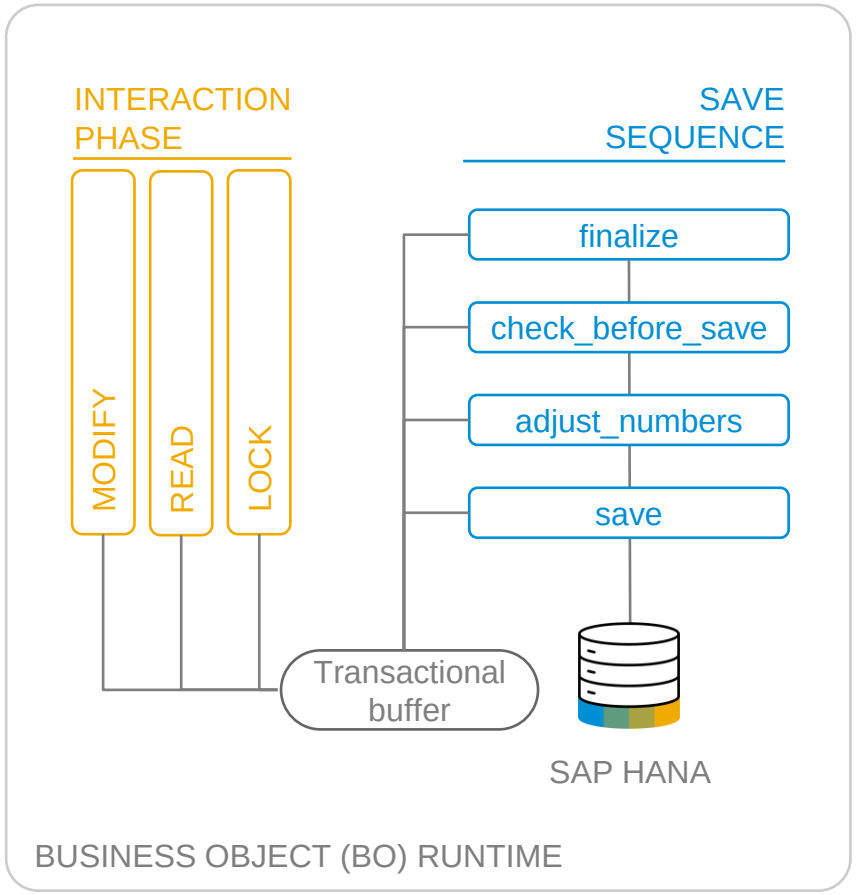
```
@EndUserText.label: 'Service Definition for Managing Travels'
define service /DMO/TRAVEL_U {
  expose /DMO/I_Travel_U as Travel;
  expose /DMO/I_Booking_U as Booking;
  expose /DMO/I_BookingSupplement_U as BookingSupplement;
  expose /DMO/I_Supplement as Supplement;
  expose /DMO/I_SupplementText 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;
}
```



Service Binding



RAP BO runtime implementation types – Overview



UNMANAGED

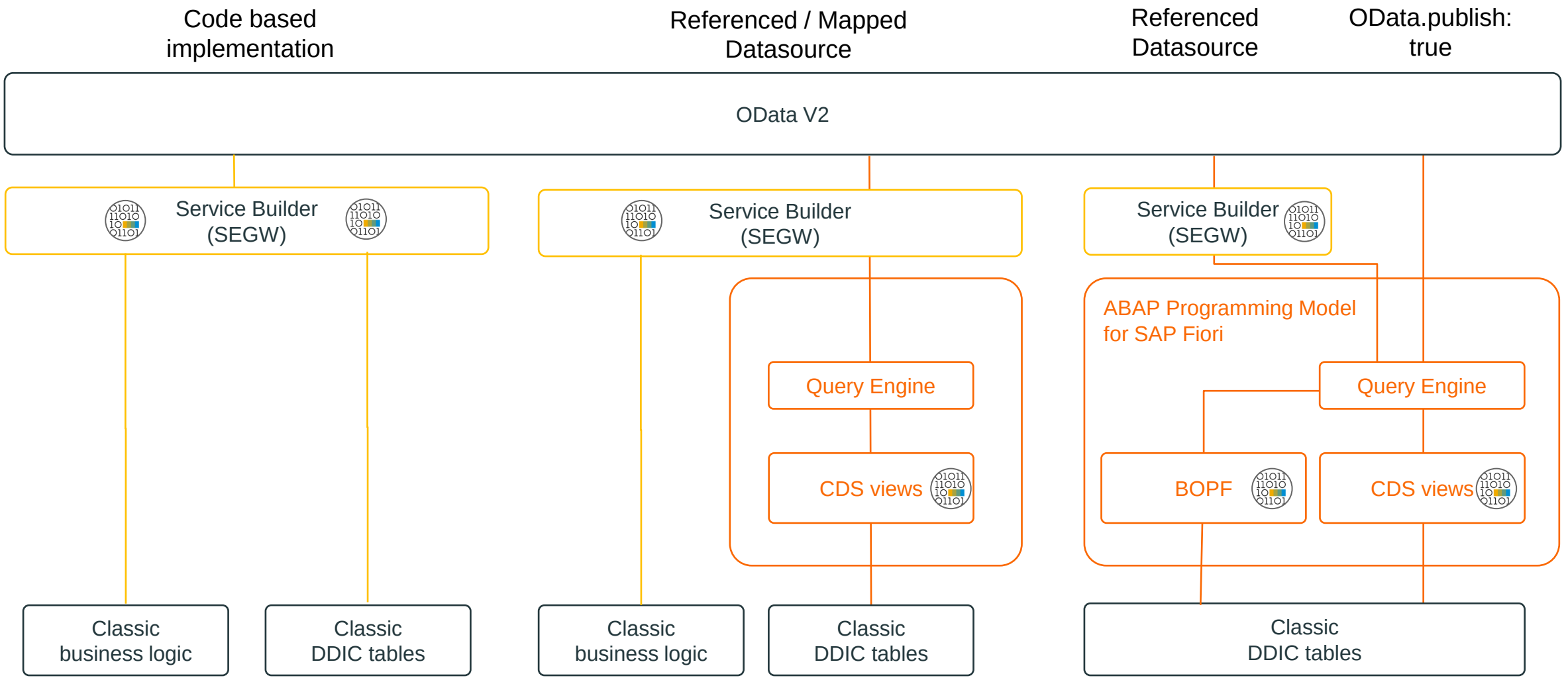
Brownfield development with application coding fully available:
Interaction phase + Transactional buffer + Save sequence

MANAGED

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

OData
development options
SAP Business Suite

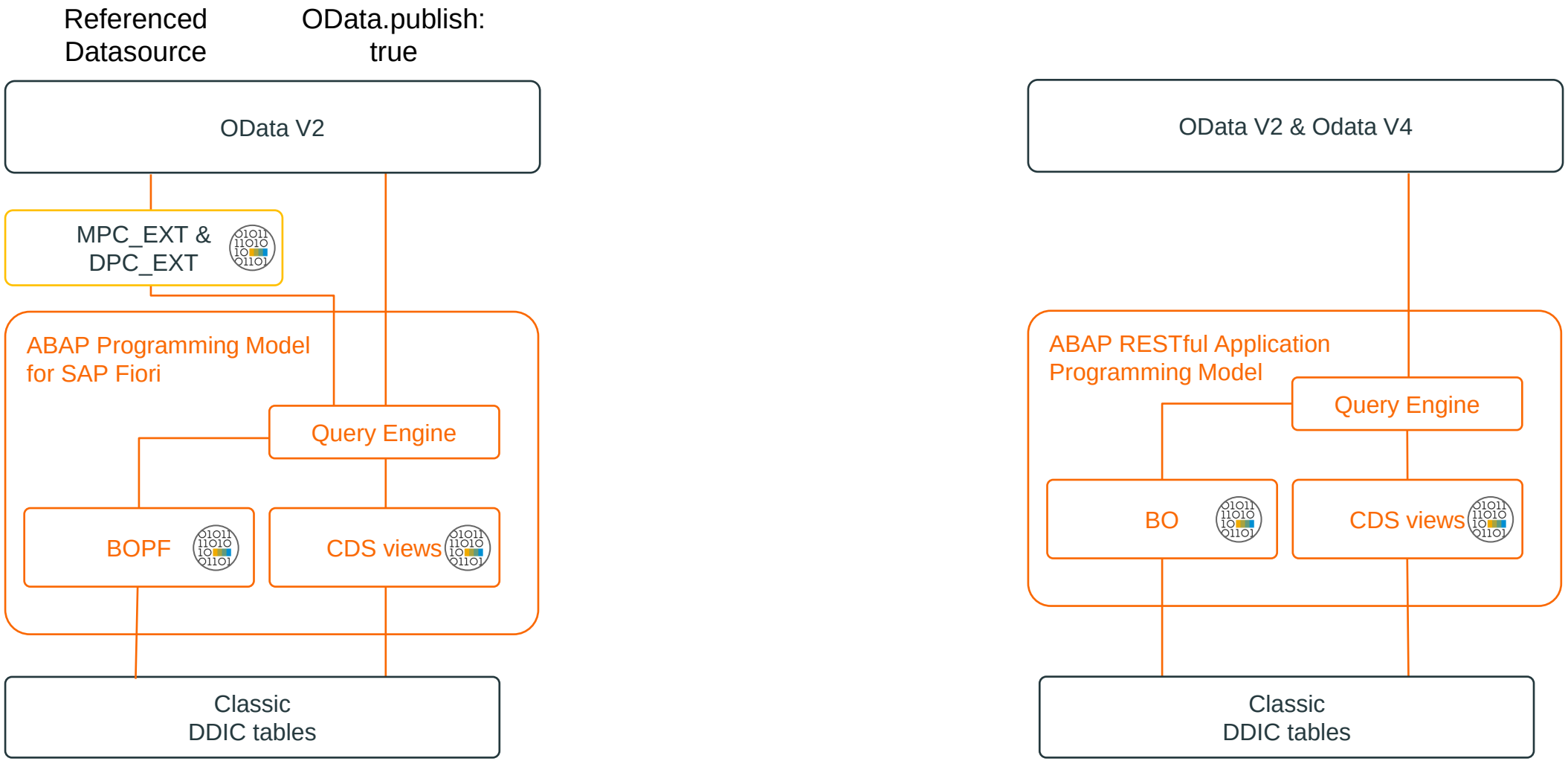
OData V2 service development options in SAP Business Suite



= code based implementation (ABAP, BOPF or CDS)

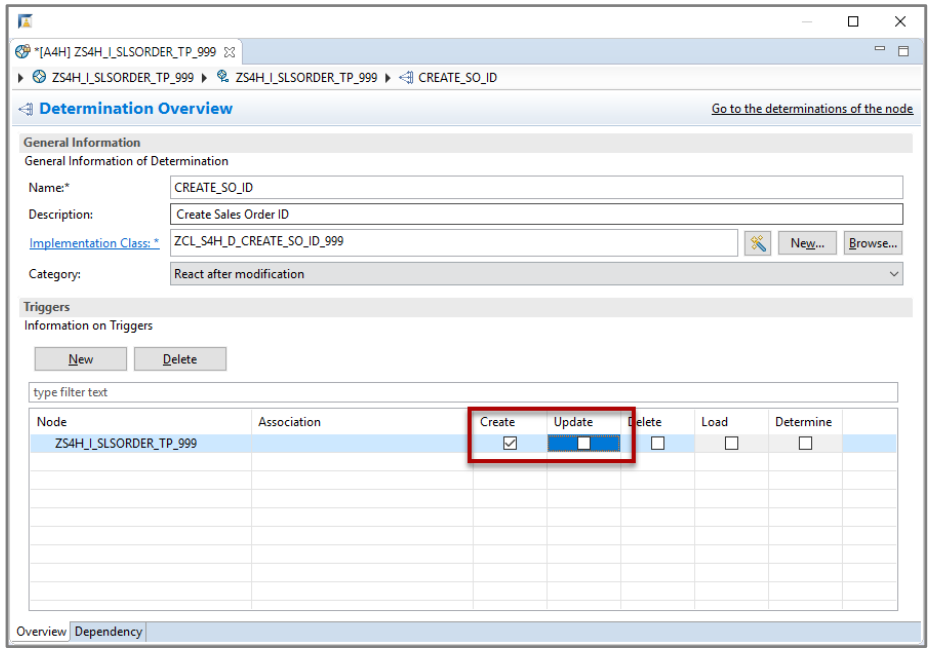
Managed implementations

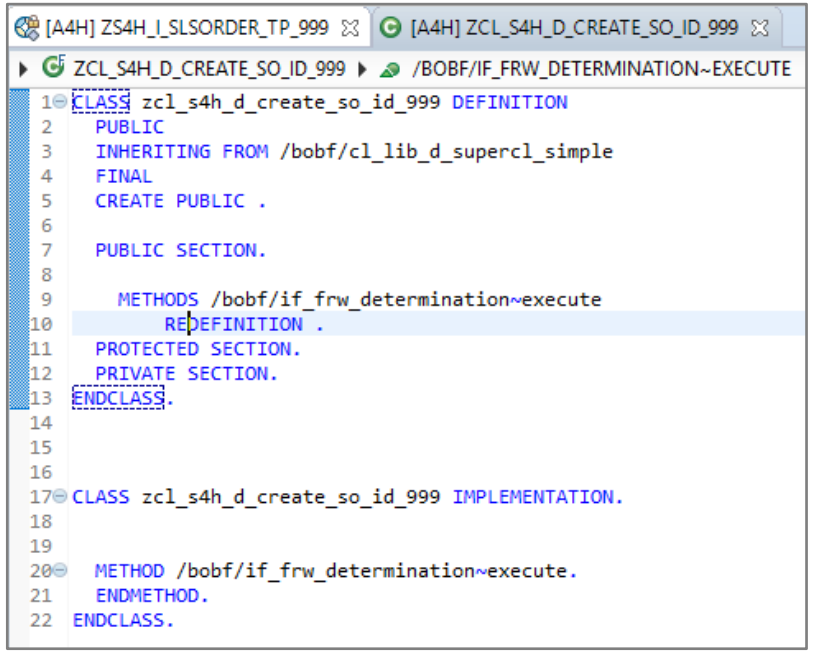
ABAP Programming Model for SAP Fiori vs. RAP



= code based implementation (ABAP, BOPF or CDS)

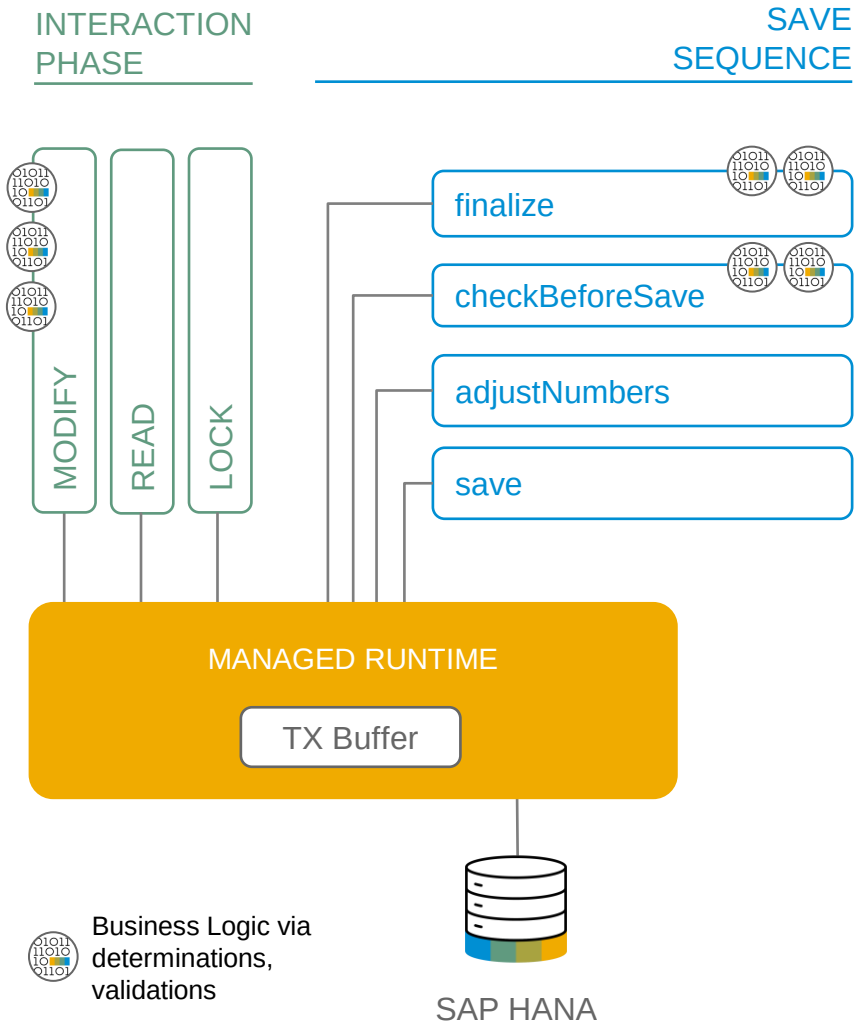
ABAP Programming Model for SAP Fiori – CDS based BOPF Objects





- 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

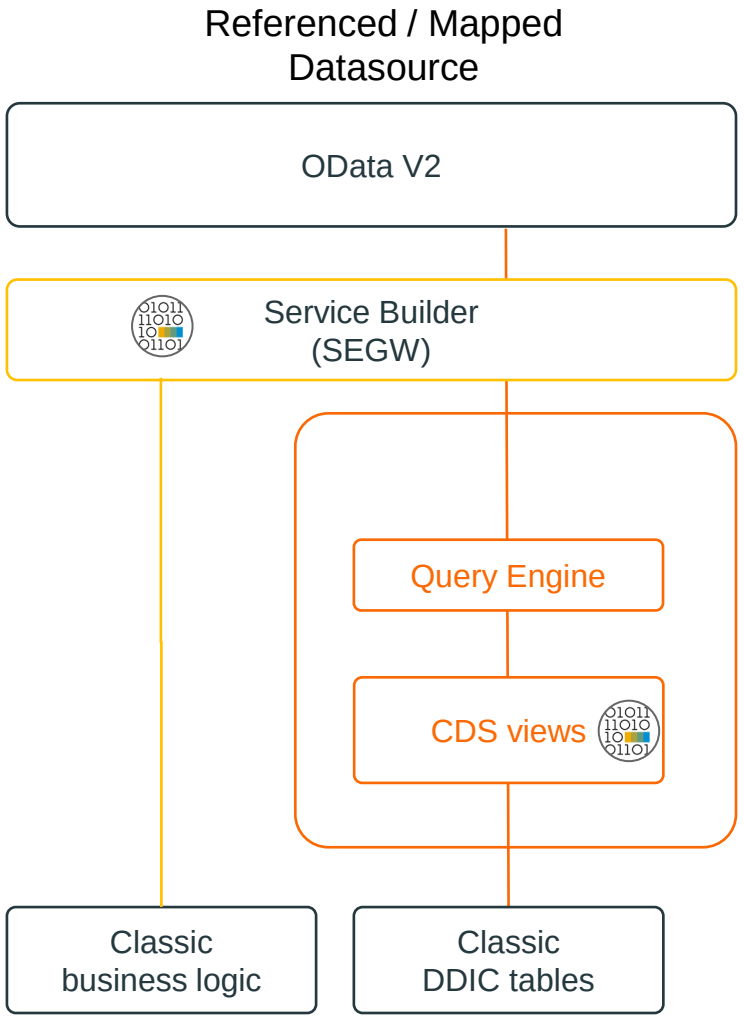
- SAP S/4HANA 2020 FSP1
  - Support for draft
  - Major limitations regarding key layout and numbering
  - If Limitations are crucial →
    Leverage RAP features that are not available in *ABAP Programming Model for SAP Fiori*

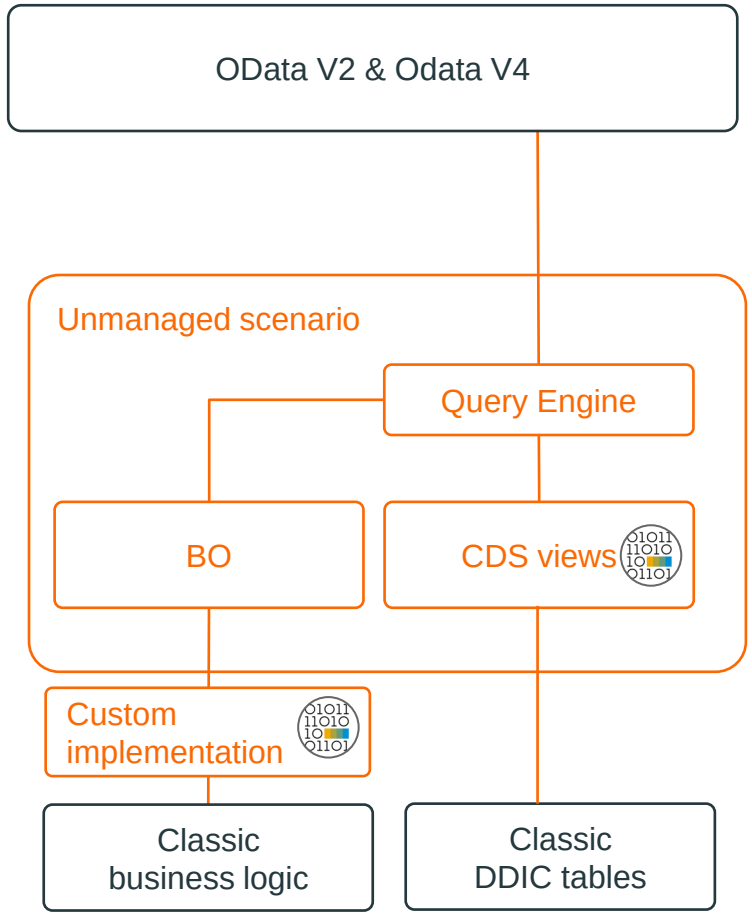
SAP S/4HANA 2021 FSP1 → Guidance: Use RAP

- Full support of all key types

Unmanaged implementations

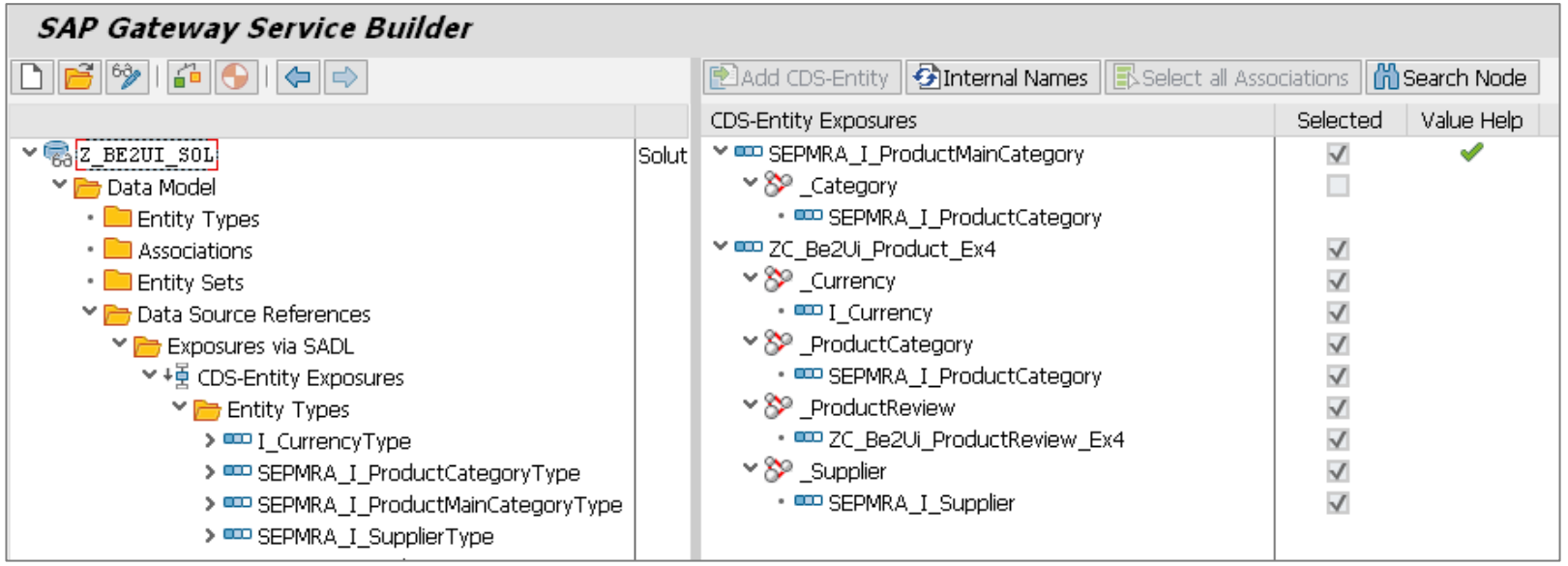
Referenced / Mapped Datasource vs. RAP





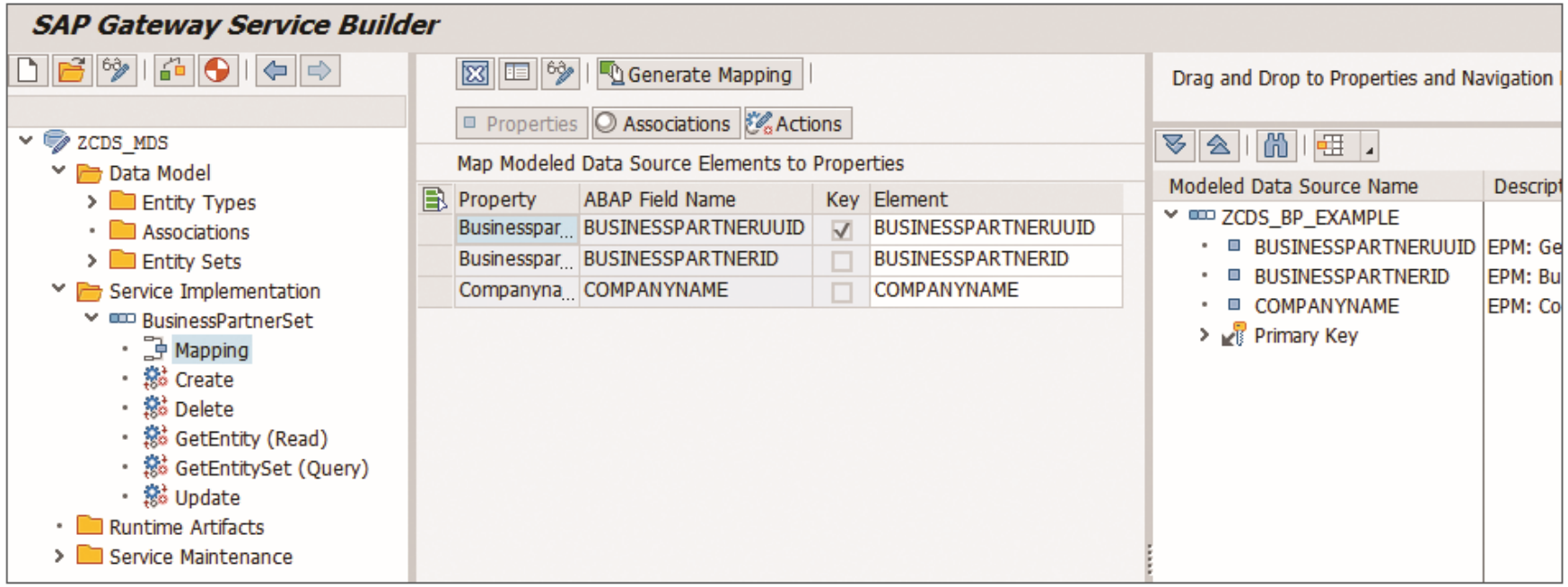
= code based implementation (ABAP, BOPF or CDS)

Referenced Data Source



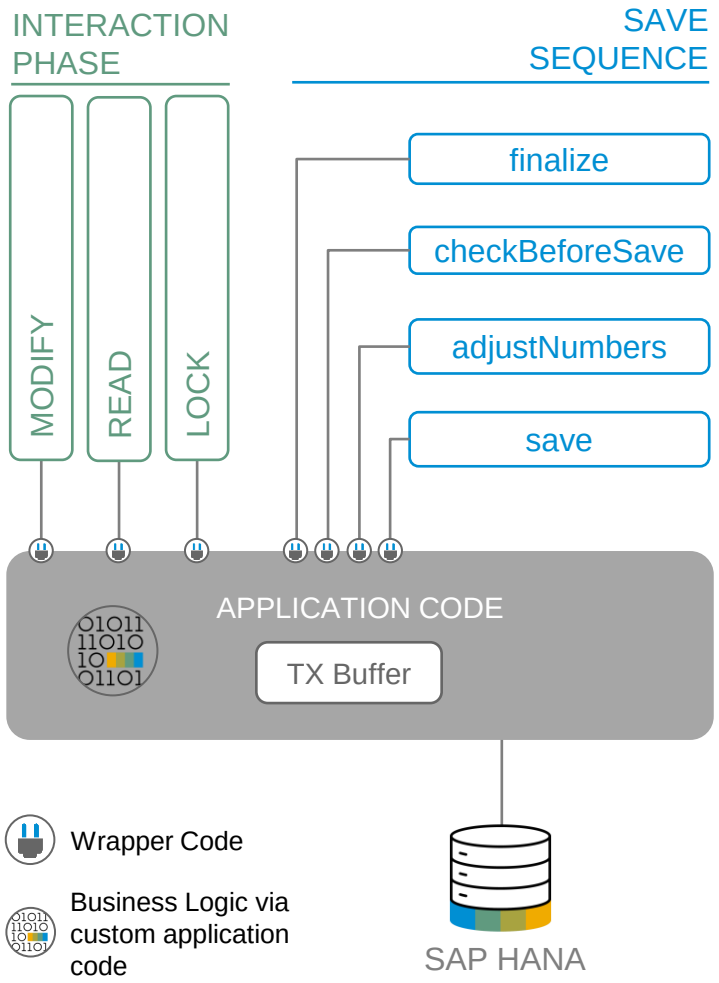
- OData Model references selected CDS views and associations
- Available as of 7.50
- New fields are automatically included
- Code based implementation of CREATE, UPDATE, and DELETE methods

Mapped Data Source



- 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

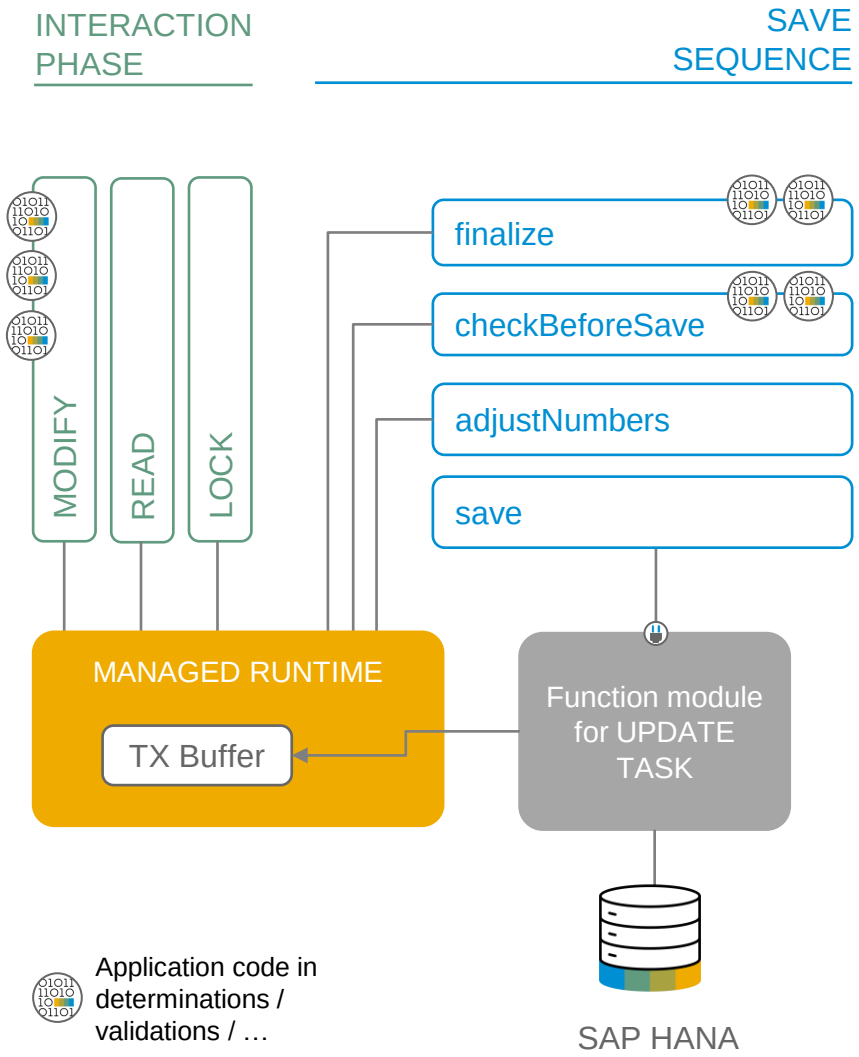
Examples

- 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

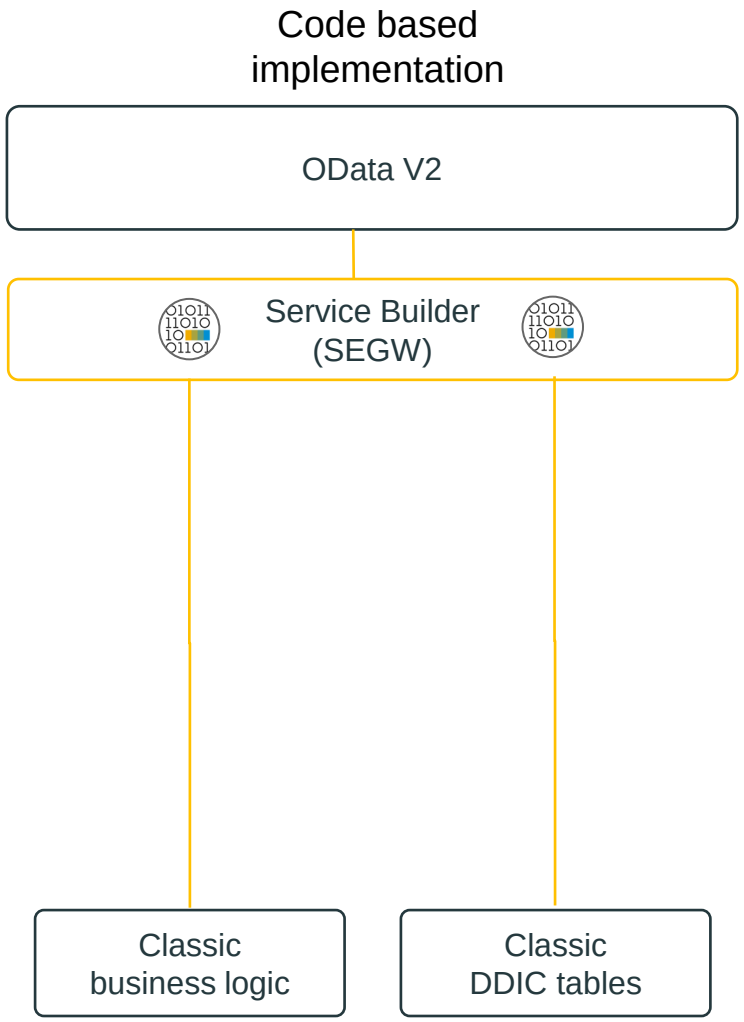
- SAP S/4HANA 1909
  - Several limitations (e.g. No filter on projection layer)
  - If limitations are crucial → Leverage Reference Data Source approach

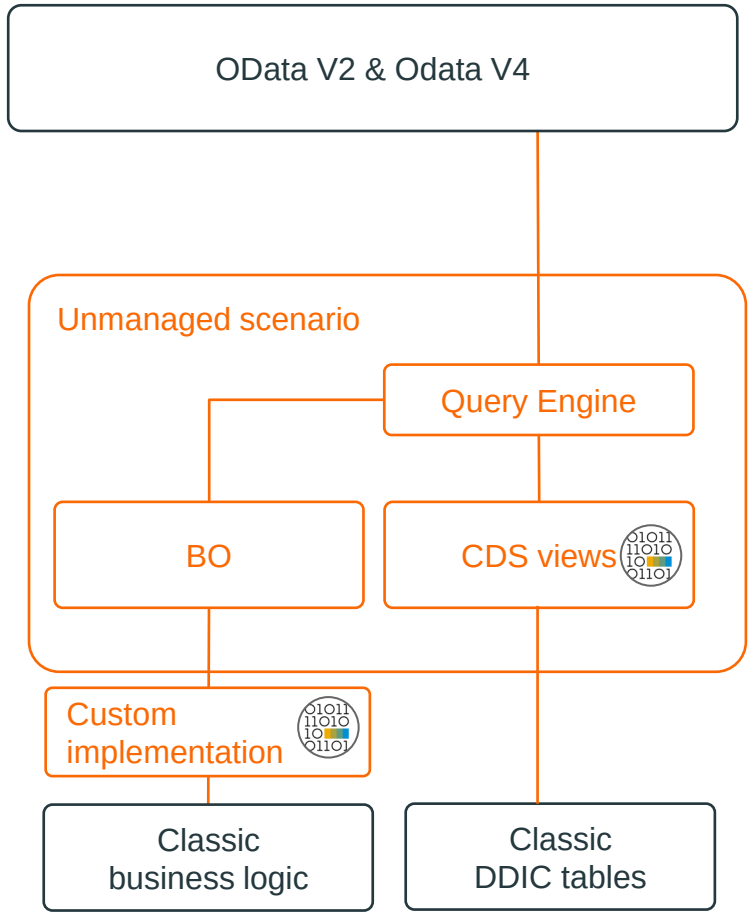
SAP S/4HANA 2020 FSP1 → Guidance: Use RAP

- Support of draft
- Support of OData V4

Unmanaged implementations (without existing CDS Views)

SEGW code based implementation vs. RAP





= code based implementation (ABAP, BOPF or CDS)

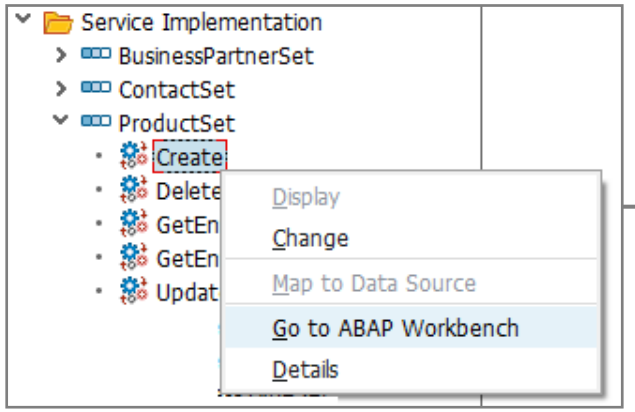
SAP Gateway
Code-based service development

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 /iwbep/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 ).
```

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

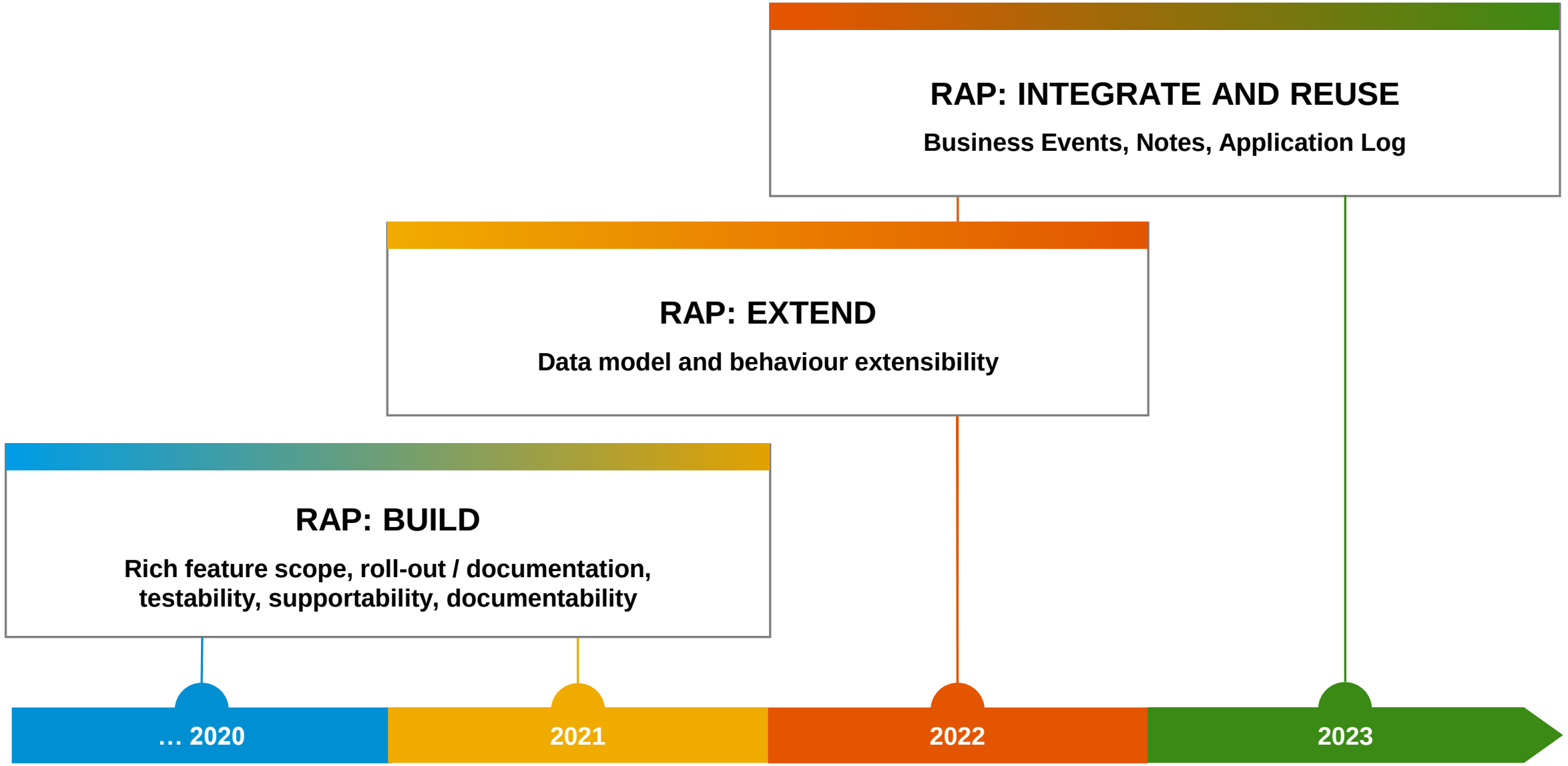
RAP vs. SEGW

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

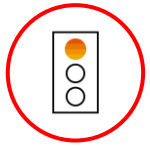
ENROLL NOW!
https://open.sap.com/courses/cp13

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

*** Best practice Fiori apps, E2E documentation**

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

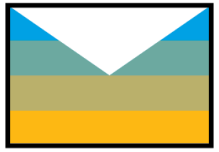
You are invited to network, collaborate and learn while enjoying these community offerings:

Questions and answers

Blogging

Recognition Program

Personalization



Direct Messaging

Coffee Corner

Events

Expert Pages