Week 3: Enabling the Transactional Behavior of an App

Unit 1: The Enhanced Business Scenario

openSAP
open.sap.com

THE BEST RUN SAP

Topics

Week 3

01 The Enhanced Business Scenario

02 Defining the Basic Business Object Behavior

03 Creating the Business Object Behavior Projection

04 Understanding Entity Manipulation Language (EML)

05 Enhancing the BO Behavior with App-Specific Logic

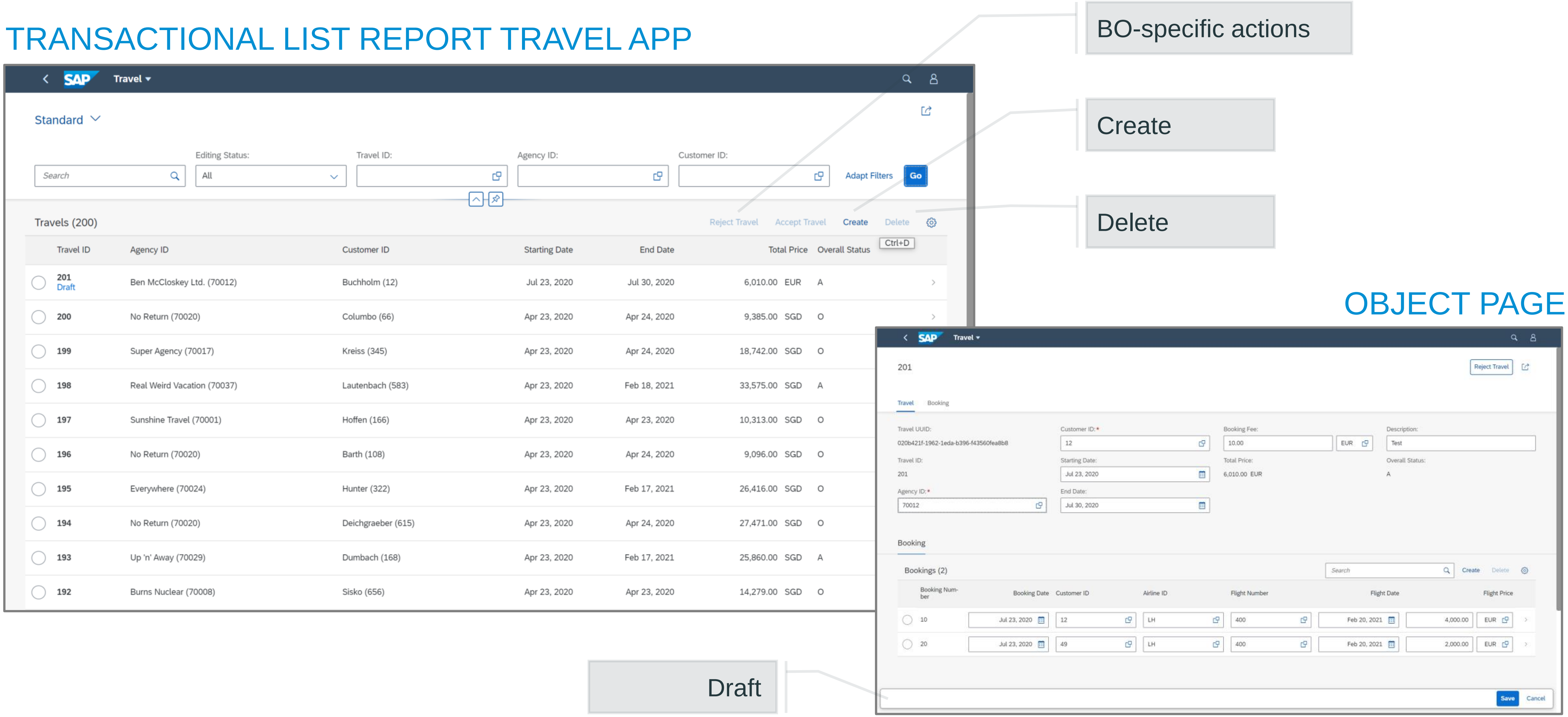
06 Implementing the Business Object Behavior

07 Enabling the Draft Handling

08 Troubleshooting Your SAP Fiori App

The Enhanced Business Scenario

Resulting SAP Fiori elements app



TRANSACTIONAL LIST REPORT TRAVEL APP

Editing Status: All

Travel ID:

Agency ID:

Customer ID:

Travels (200)

	Travel ID	Agency ID	Customer ID	Starting Date	End Date	Total Price		Overall Status
○	201 Draft	Ben McCloskey Ltd. (70012)	Buchholm (12)	Jul 23, 2020	Jul 30, 2020	6,010.00	EUR	A
○	200	No Return (70020)	Columbo (66)	Apr 23, 2020	Apr 24, 2020	9,385.00	SGD	O
○	199	Super Agency (70017)	Kreiss (345)	Apr 23, 2020	Apr 24, 2020	18,742.00	SGD	O
○	198	Real Weird Vacation (70037)	Lautenbach (583)	Apr 23, 2020	Feb 18, 2021	33,575.00	SGD	A
○	197	Sunshine Travel (70001)	Hoffen (166)	Apr 23, 2020	Apr 23, 2020	10,313.00	SGD	O
○	196	No Return (70020)	Barth (108)	Apr 23, 2020	Apr 24, 2020	9,096.00	SGD	O
○	195	Everywhere (70024)	Hunter (322)	Apr 23, 2020	Feb 17, 2021	26,416.00	SGD	O
○	194	No Return (70020)	Deichgraeber (615)	Apr 23, 2020	Apr 24, 2020	27,471.00	SGD	O
○	193	Up 'n' Away (70029)	Dumbach (168)	Apr 23, 2020	Feb 17, 2021	25,860.00	SGD	A
○	192	Burns Nuclear (70008)	Sisko (656)	Apr 23, 2020	Apr 23, 2020	14,279.00	SGD	O

BO-specific actions

Create

Delete

OBJECT PAGE

201

Travel UUID: 020b421f-1962-1eda-b396-f43560fea8b8

Travel ID: 201

Agency ID:* 70012

Customer ID:* 12

Starting Date: Jul 23, 2020

End Date: Jul 30, 2020

Booking Fee: 10.00 EUR

Total Price: 6,010.00 EUR

Description: Test

Overall Status: A

Booking

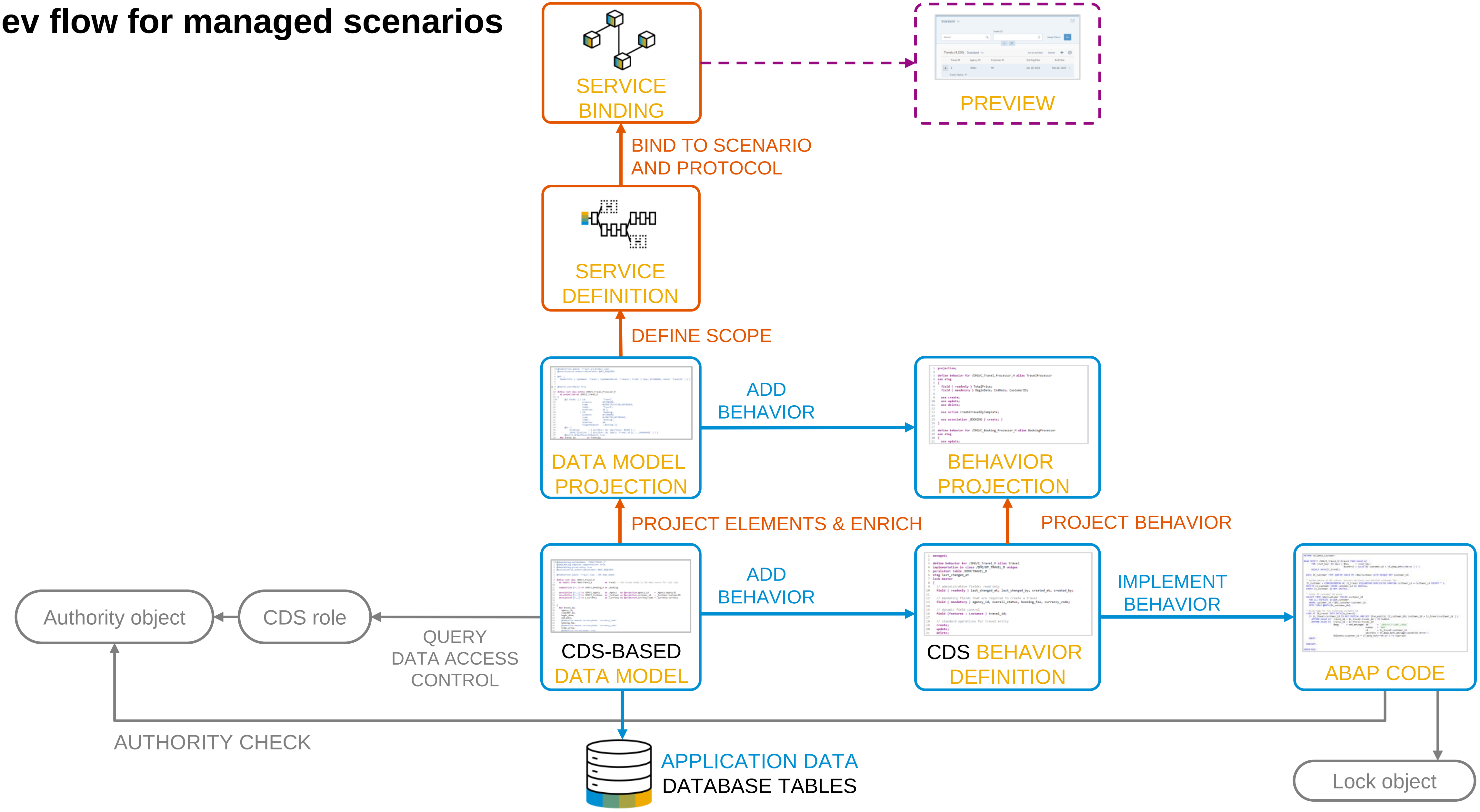
Bookings (2)

	Booking Number	Booking Date	Customer ID	Airline ID	Flight Number	Flight Date	Flight Price	
○	10	Jul 23, 2020	12	LH	400	Feb 20, 2021	4,000.00	EUR
○	20	Jul 23, 2020	49	LH	400	Feb 20, 2021	2,000.00	EUR

Draft

The Enhanced Business Scenario

Dev flow for managed scenarios



SERVICE BINDING

PREVIEW

BIND TO SCENARIO AND PROTOCOL

SERVICE DEFINITION

DEFINE SCOPE

DATA MODEL PROJECTION

ADD BEHAVIOR

BEHAVIOR PROJECTION

PROJECT ELEMENTS & ENRICH

PROJECT BEHAVIOR

Authority object

CDS role

QUERY DATA ACCESS CONTROL

CDS-BASED DATA MODEL

ADD BEHAVIOR

CDS BEHAVIOR DEFINITION

IMPLEMENT BEHAVIOR

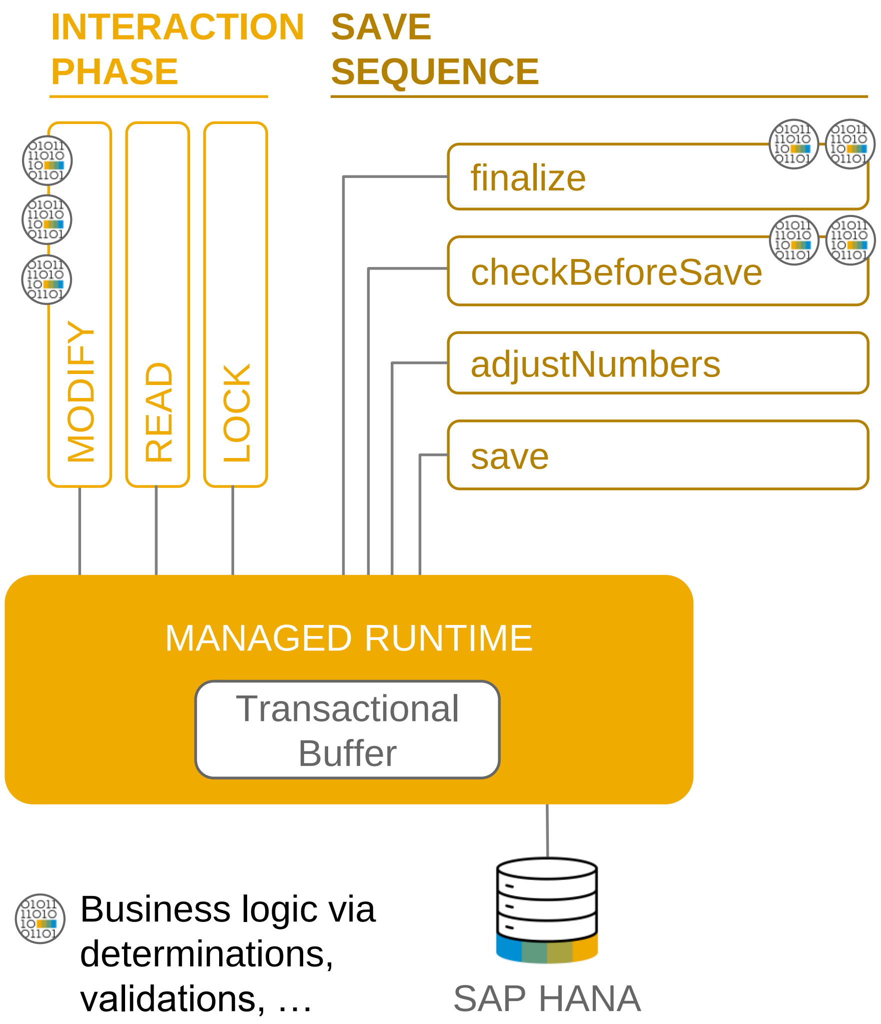
ABAP CODE

AUTHORITY CHECK

APPLICATION DATA DATABASE TABLES

Lock object

Managed business object runtime implementation



Application code

- Not yet available or fine granular reusable code available
- Technical implementation tasks taken over by BO infrastructure
- Standard CRUD operations available out-of-the-box
- Developer focuses on adding BO-specific business logic via dedicated code exits: determinations, validation and actions

Examples

- New applications on SAP Cloud Platform, ABAP environment and SAP S/4HANA

The Enhanced Business Scenario

Wrap-up

IN THIS UNIT, YOU LEARNED

- About the transactional enhancements of the business scenario we implement this week
- How the development flow will be in the upcoming units
- About the managed business object runtime implementation

NEXT UNIT

- Week 3 – Unit 2
  Defining the Basic Business Object Behavior

The Enhanced Business Scenario

Further reading

i Additional Material

ABAP RESTful APPLICATION PROGRAMMING MODEL INFORMATION PAGE

For more information, links to documentation, tutorials, and more, please visit the RAP at openSAP information page by following the link below

RAP at openSAP information page (week 3)

Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE’s or its affiliated companies’ strategy and possible future developments, products, and/or platforms, directions, and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

THE BEST RUN SAP®

Week 3: Enabling the Transactional Behavior of an App

Unit 2: Defining the Basic Business Object Behavior

Enabling the Transactional Behavior of an App

Topics

Week 3

01 The Enhanced Business Scenario

02 Defining the Basic Business Object Behavior

03 Creating the Business Object Behavior Projection

04 Understanding Entity Manipulation Language (EML)

05 Enhancing the BO Behavior with App-Specific Logic

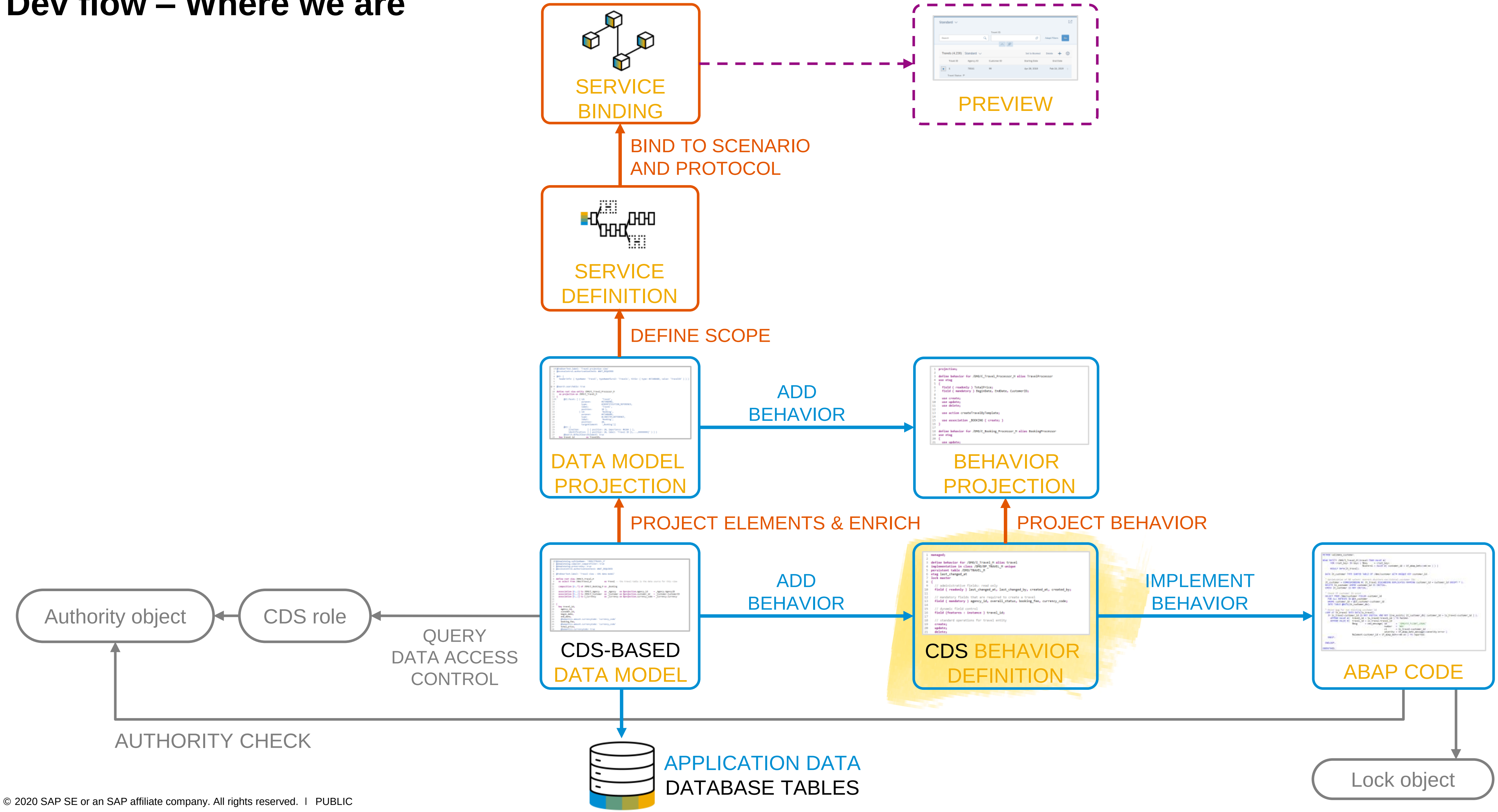
06 Implementing the Business Object Behavior

07 Enabling the Draft Handling

08 Troubleshooting Your SAP Fiori App

Defining the Basic Business Object Behavior

Dev flow – Where we are



SERVICE BINDING

PREVIEW

BIND TO SCENARIO AND PROTOCOL

SERVICE DEFINITION

DEFINE SCOPE

DATA MODEL PROJECTION

ADD BEHAVIOR

BEHAVIOR PROJECTION

PROJECT ELEMENTS & ENRICH

PROJECT BEHAVIOR

Authority object

CDS role

QUERY DATA ACCESS CONTROL

CDS-BASED DATA MODEL

ADD BEHAVIOR

CDS BEHAVIOR DEFINITION

IMPLEMENT BEHAVIOR

ABAP CODE

AUTHORITY CHECK

APPLICATION DATA DATABASE TABLES

Lock object

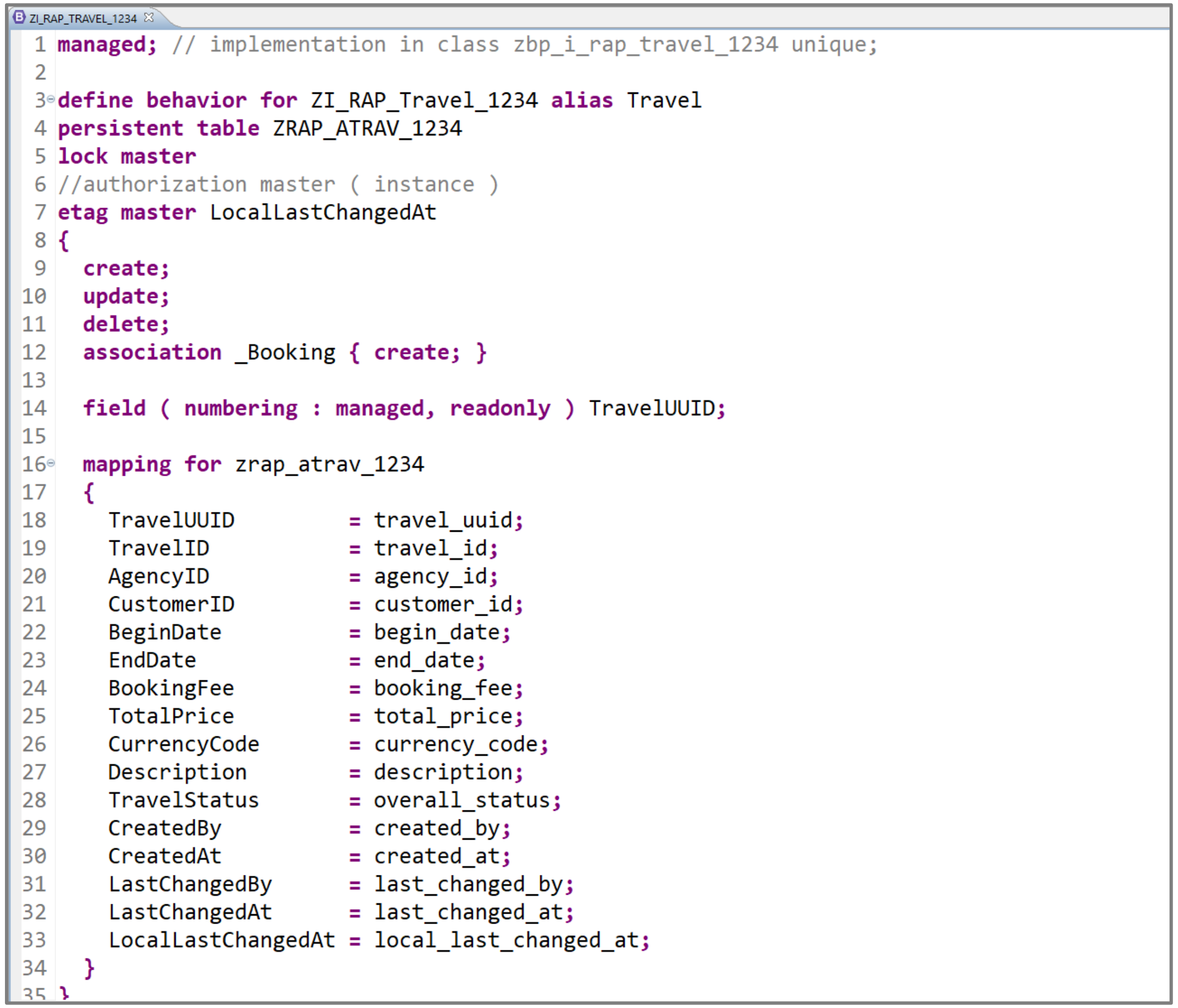
Defining the Basic Business Object Behavior

What we will achieve in this unit

Define table for persistency

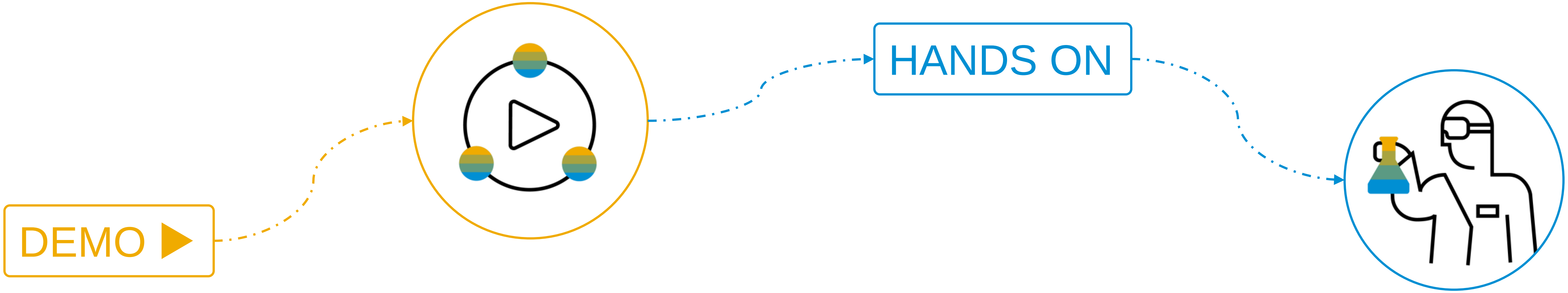
List of actions

Mapping between entity and persistent table



```
managed; // implementation in class zbp_i_rap_travel_1234 unique;

define behavior for ZI_RAP_Travel_1234 alias Travel
persistent table ZRAP_ATRAV_1234
lock master
//authorization master ( instance )
etag master LocalLastChangedAt
{
  create;
  update;
  delete;
  association _Booking { create; }

  field ( numbering : managed, readonly ) TravelUUID;

  mapping for zrap_atrav_1234
  {
    TravelUUID         = travel_uuid;
    TravelID           = travel_id;
    AgencyID           = agency_id;
    CustomerID         = customer_id;
    BeginDate          = begin_date;
    EndDate            = end_date;
    BookingFee         = booking_fee;
    TotalPrice         = total_price;
    CurrencyCode       = currency_code;
    Description        = description;
    TravelStatus       = overall_status;
    CreatedBy          = created_by;
    CreatedAt          = created_at;
    LastChangedBy      = last_changed_by;
    LastChangedAt      = last_changed_at;
    LocalLastChangedAt = local_last_changed_at;
  }
}
```

Demo



Creating and Defining the Transactional BO behavior

1. Create the BO behavior definition for the travel and booking entities

2. Define the standard transactional BO behavior of each entity

3. Test the enhanced SAP Fiori elements app

Defining the Basic Business Object Behavior

Wrap-up

IN THIS UNIT, YOU LEARNED

- About the transactional behavior of your app and how to define it utilizing ABAP Development Tools

NEXT UNIT

- Week 3 – Unit 3
  Creating the Business Object Behavior Projection

Defining the Basic Business Object Behavior

Further reading

i Additional Material

ABAP RESTful APPLICATION PROGRAMMING MODEL INFORMATION PAGE

For more information, links to documentation, tutorials, and more, please visit the RAP at openSAP information page by following the link below

RAP at openSAP information page (week 3)

Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platforms, directions, and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

THE BEST RUN SAP

Week 3: Enabling the Transactional Behavior of an App

Unit 3: Creating the Business Object Behavior Projection

openSAP
open.sap.com

THE BEST RUN SAP

Creating the Business Object Behavior Projection

Topics

Week 3

01 The Enhanced Business Scenario

02 Defining the Basic Business Object Behavior

03 Creating the Business Object Behavior Projection

04 Understanding Entity Manipulation Language (EML)

05 Enhancing the BO Behavior with App-Specific Logic

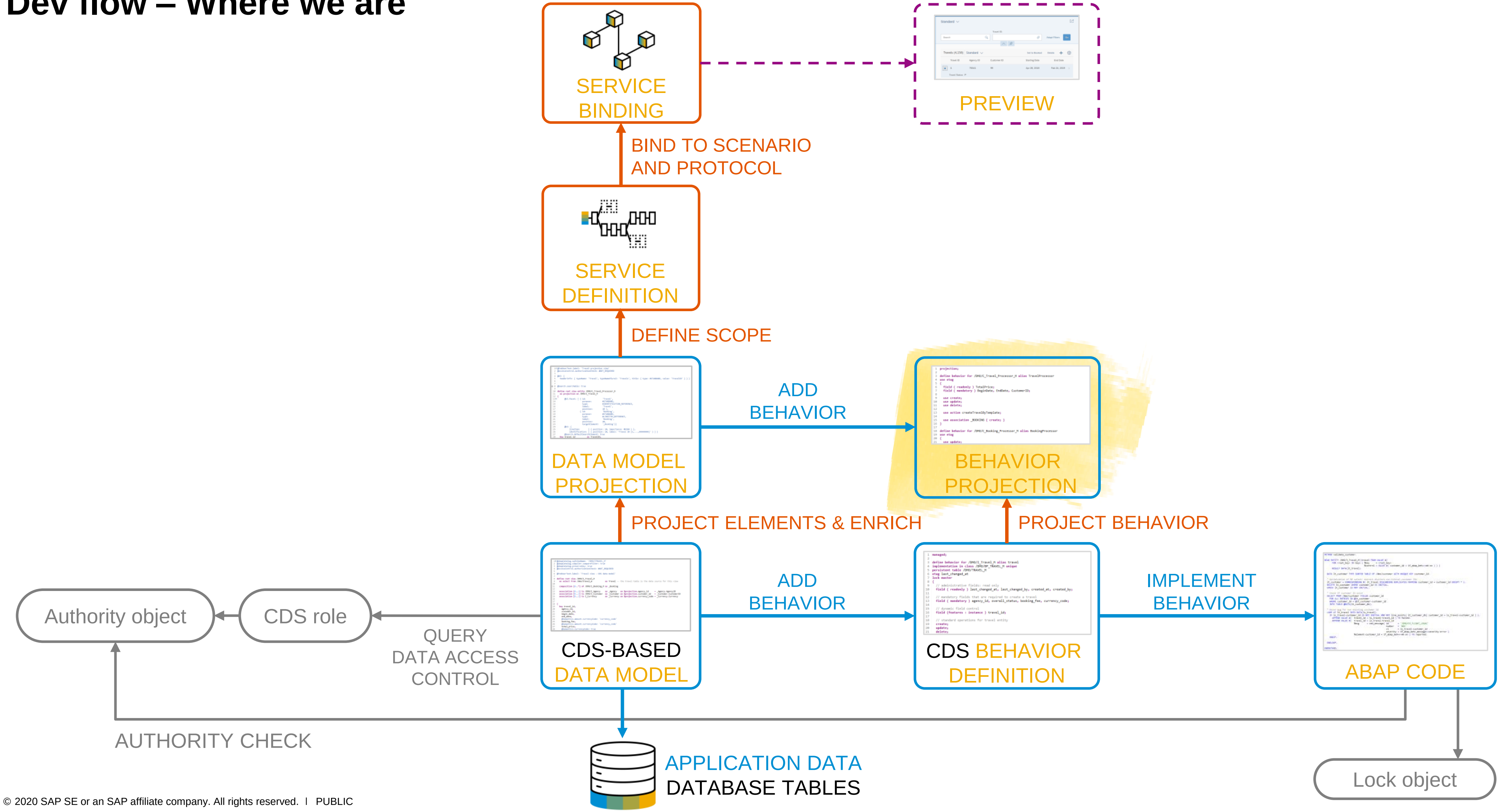
06 Implementing the Business Object Behavior

07 Enabling the Draft Handling

08 Troubleshooting Your SAP Fiori App

Creating the Business Object Behavior Projection

Dev flow – Where we are



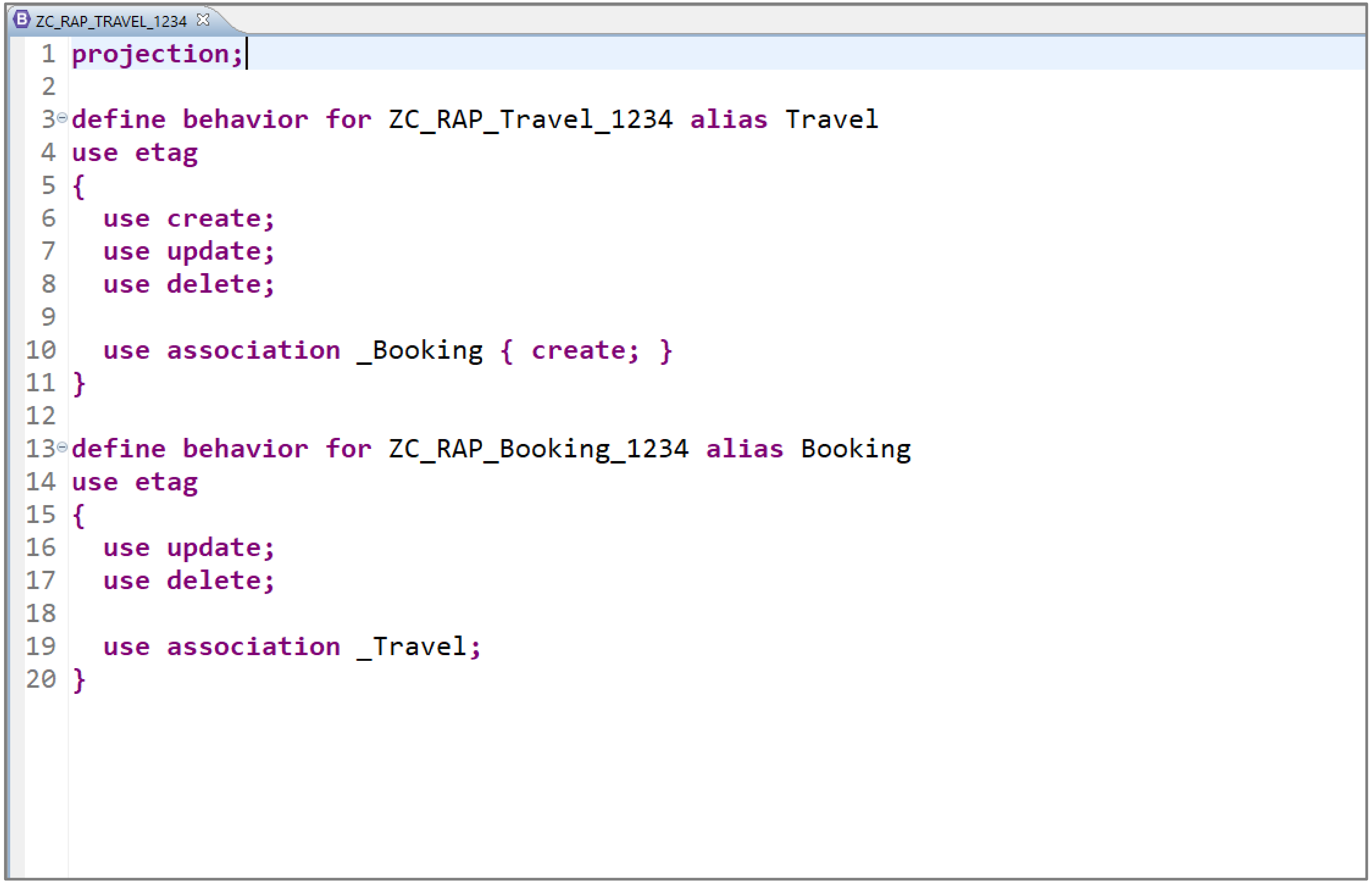
```
projection;

define behavior for /DMO/C_Travel_Processor_M alias TravelProcessor
use etag
{
  field ( readonly ) TotalPrice;
  field ( mandatory ) BeginDate, EndDate, CustomerID;

  use create;
  use update;
  use delete;

  use action createTravelByTemplate;

  use association _BOOKING { create; }
}

define behavior for /DMO/C_Booking_Processor_M alias BookingProcessor
use etag
{
  use update;
```

```
managed;

define behavior for /DMO/I_Travel_M alias travel
implementation in class /DMO/BP_TRAVEL_M unique
persistent table /DMO/TRAVEL_M
etag last_changed_at
lock master
{
  // administrative fields: read only
  field ( readonly ) last_changed_at, last_changed_by, created_at, created_by;

  // mandatory fields that are required to create a travel
  field ( mandatory ) agency_id, overall_status, booking_fee, currency_code;

  // dynamic field control
  field (features : instance ) travel_id;

  // standard operations for travel entity
  create;
  update;
  delete;
```

Creating the Business Object Behavior Projection

What we will achieve in this unit

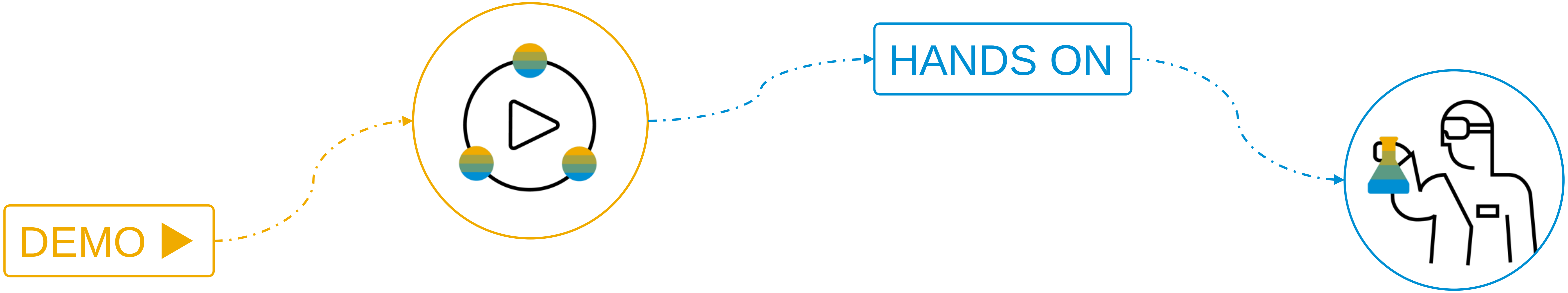
Projected features from the base behavior definition



```
projection;

define behavior for ZC_RAP_Travel_1234 alias Travel
use etag
{
  use create;
  use update;
  use delete;

  use association _Booking { create; }
}

define behavior for ZC_RAP_Booking_1234 alias Booking
use etag
{
  use update;
  use delete;

  use association _Travel;
}
```

Creating the Business Object Behavior Projection

Demo



Creating and Defining the Transactional BO Behavior

1. Create the BO behavior projection for the travel and booking entities
2. Project the transactional BO behavior of each entity
3. Test the enhanced SAP Fiori elements app

Creating the Business Object Behavior Projection

Wrap-up

IN THIS UNIT, YOU LEARNED

- What a business object behavior projection is and how it can be defined

NEXT UNIT

- Week 3 – Unit 4
  Understanding Entity Manipulation Language (EML)

Creating the Business Object Behavior Projection

Further reading

i Additional Material

ABAP RESTful APPLICATION PROGRAMMING MODEL INFORMATION PAGE

For more information, links to documentation, tutorials, and more, please visit the RAP at openSAP information page by following the link below

RAP at openSAP information page (week 3)

Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platforms, directions, and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

THE BEST RUN SAP

Week 3: Enabling the Transactional Behavior of an App

Unit 4: Understanding Entity Manipulation Language (EML)

openSAP
open.sap.com

THE BEST RUN SAP

Understanding Entity Manipulation Language (EML)

Topics

Week 3

01 The Enhanced Business Scenario

02 Defining the Basic Business Object Behavior

03 Creating the Business Object Behavior Projection

04 Understanding Entity Manipulation Language (EML)

05 Enhancing the BO Behavior with App-Specific Logic

06 Implementing the Business Object Behavior

07 Enabling the Draft Handling

08 Troubleshooting Your SAP Fiori App

EML at a glance

ENTITY MANIPULATION LANGUAGE

EXTENSION OF THE ABAP LANGUAGE
with an SQL-like syntax

CONTROL THE TRANSACTIONAL BO BEHAVIOR IN RAP CONTEXT

DIRECT API-BASED ACCESS TO RAP BOs

STANDARD EML API
for type-safe read and modifying access to RAP BOs

GENERIC EML API
for generic integration of RAP BOs into other frameworks

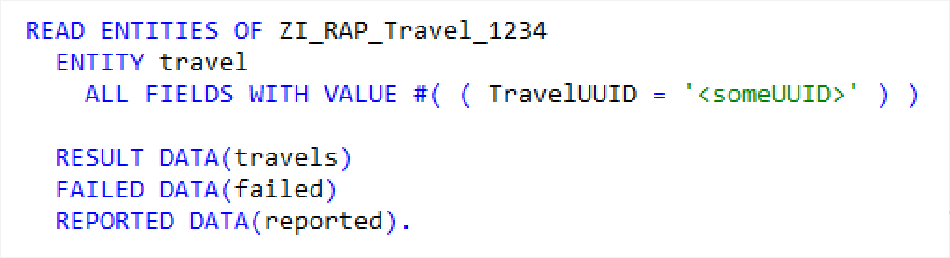
DATA CONSISTENCY ENSURED BY DATABASE LUW
COMMIT operation required to persist changes

Understanding Entity Manipulation Language (EML)

READ operation

READ ACCESS TO RAP BOs

EXAMPLE



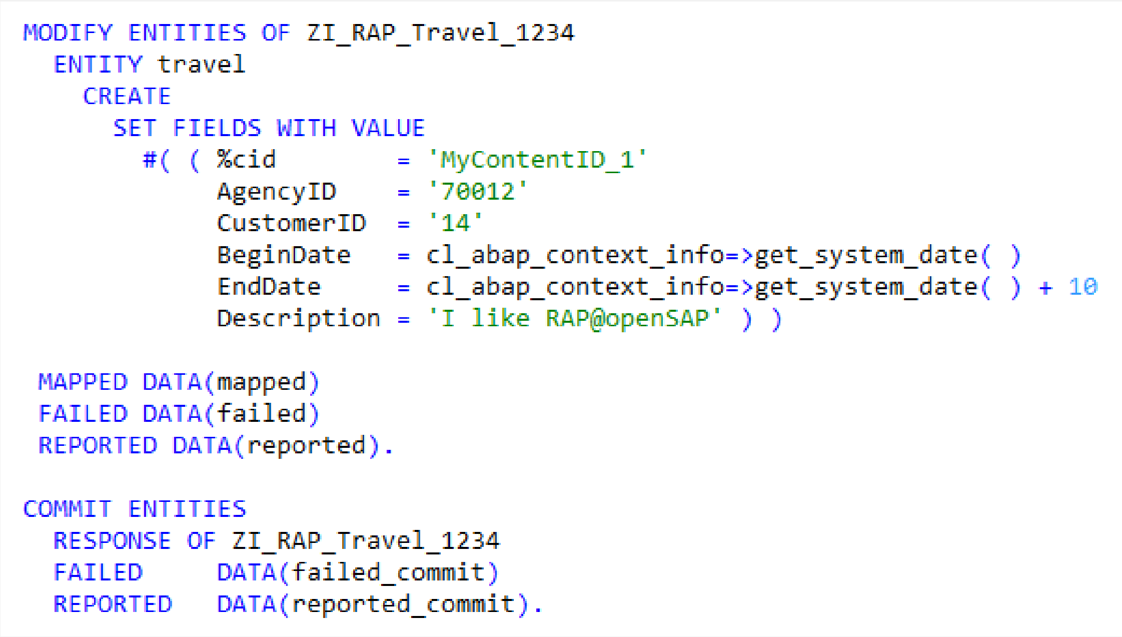
```
READ ENTITIES OF ZI_RAP_Travel_1234
  ENTITY travel
    ALL FIELDS WITH VALUE #( ( TravelUUID = '<someUUID>' ) )

  RESULT DATA(travels)
  FAILED DATA(failed)
  REPORTED DATA(reported).
```

Failure handling (failed & reported)

Understanding Entity Manipulation Language (EML)

MODIFY – CREATE operation

EXAMPLE



```
MODIFY ENTITIES OF ZI_RAP_Travel_1234
  ENTITY travel
    CREATE
      SET FIELDS WITH VALUE
        #( ( %cid        = 'MyContentID_1'
             AgencyID    = '70012'
             CustomerID  = '14'
             BeginDate   = cl_abap_context_info=>get_system_date( )
             EndDate     = cl_abap_context_info=>get_system_date( ) + 10
             Description = 'I like RAP@openSAP' ) )

 MAPPED DATA(mapped)
 FAILED DATA(failed)
 REPORTED DATA(reported).

COMMIT ENTITIES
  RESPONSE OF ZI_RAP_Travel_1234
  FAILED     DATA(failed_commit)
  REPORTED   DATA(reported_commit).
```

MODIFY ACCESS TO RAP BOs

Failure handling (failed & reported)

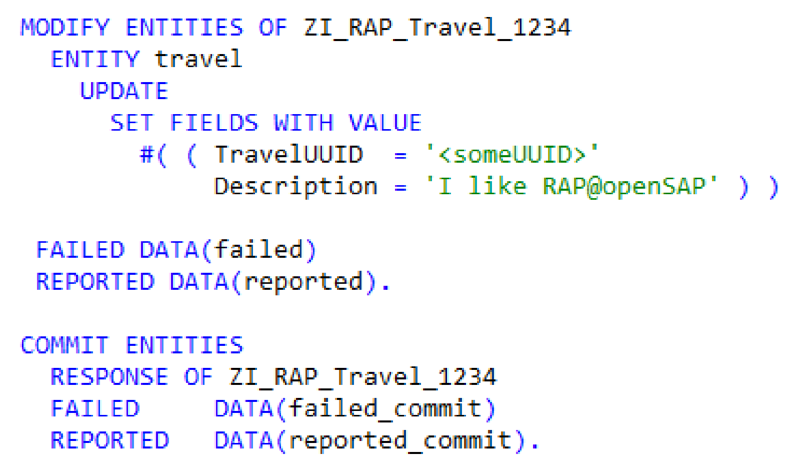
COMMIT statement for proper LUW handling

Understanding Entity Manipulation Language (EML)

MODIFY – UPDATE operation

MODIFY ACCESS TO RAP BOs

EXAMPLE



```
MODIFY ENTITIES OF ZI_RAP_Travel_1234
  ENTITY travel
    UPDATE
      SET FIELDS WITH VALUE
        #( ( TravelUUID  = '<someUUID>'
             Description = 'I like RAP@openSAP' ) )

 FAILED DATA(failed)
 REPORTED DATA(reported).

COMMIT ENTITIES
  RESPONSE OF ZI_RAP_Travel_1234
  FAILED     DATA(failed_commit)
  REPORTED   DATA(reported_commit).
```

Failure handling (failed & reported)

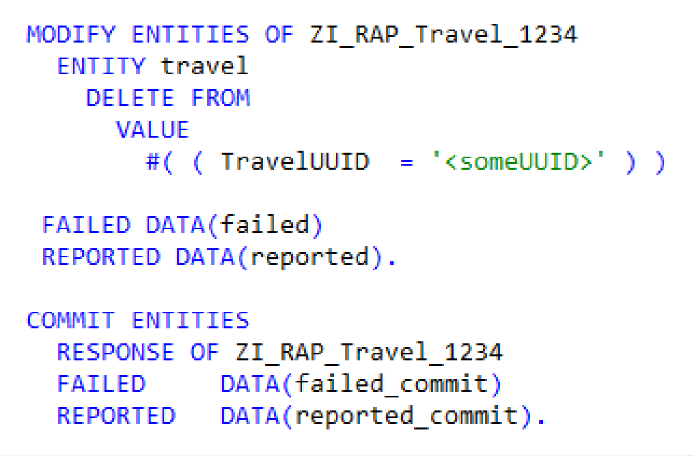
COMMIT statement for proper LUW handling

Understanding Entity Manipulation Language (EML)

MODIFY – DELETE operation

DELETE ACCESS TO RAP BOs

EXAMPLE



```
MODIFY ENTITIES OF ZI_RAP_Travel_1234
  ENTITY travel
    DELETE FROM
      VALUE
        #( ( TravelUUID  = '<someUUID>' ) )

 FAILED DATA(failed)
 REPORTED DATA(reported).

COMMIT ENTITIES
  RESPONSE OF ZI_RAP_Travel_1234
  FAILED     DATA(failed_commit)
  REPORTED   DATA(reported_commit).
```

Failure handling (failed & reported)

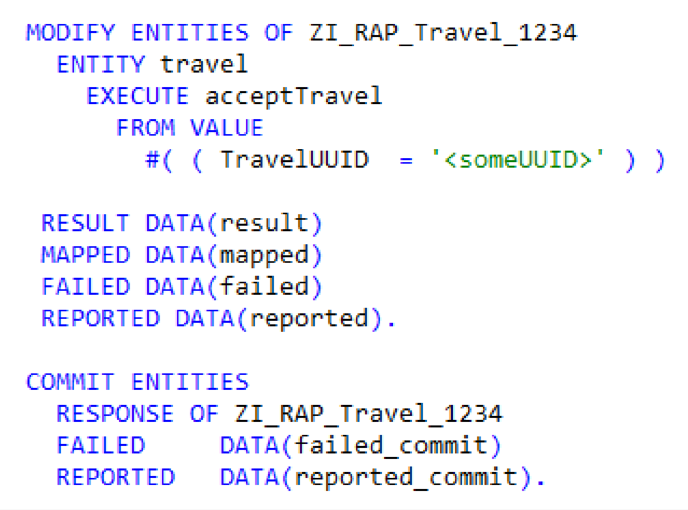
COMMIT statement for proper LUW handling

Understanding Entity Manipulation Language (EML)

MODIFY – EXECUTE ACTION operation

EXECUTION OF RAP BO-SPECIFIC ACTIONS

EXAMPLE



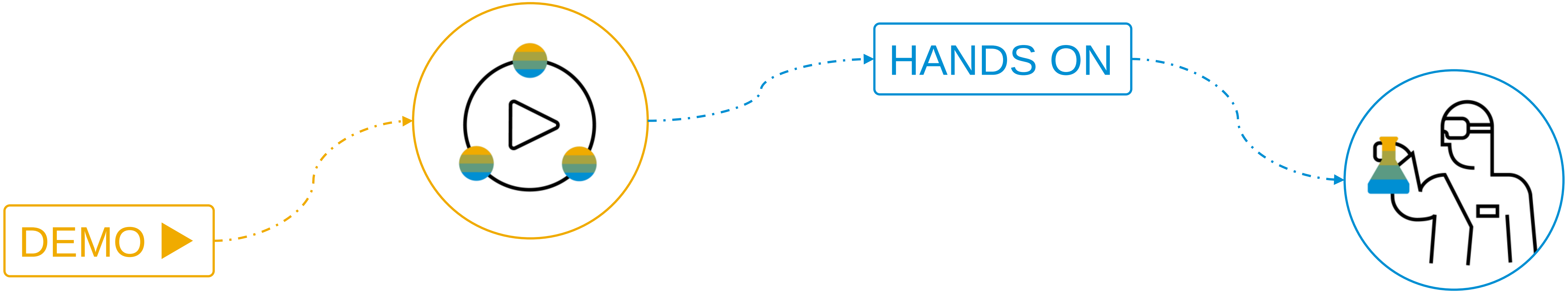
```
MODIFY ENTITIES OF ZI_RAP_Travel_1234
  ENTITY travel
    EXECUTE acceptTravel
      FROM VALUE
        #( ( TravelUUID  = '<someUUID>' ) )

 RESULT DATA(result)
 MAPPED DATA(mapped)
 FAILED DATA(failed)
 REPORTED DATA(reported).

COMMIT ENTITIES
  RESPONSE OF ZI_RAP_Travel_1234
  FAILED     DATA(failed_commit)
  REPORTED   DATA(reported_commit).
```

Failure handling (failed & reported)

COMMIT statement for proper LUW handling

Understanding Entity Manipulation Language (EML)

Demo



DEMO ▶

HANDS ON

Practice and Understand the EML Syntax

1. Read an existing travel business object instance
2. Update a given travel business object instance
3. Delete a travel business object instance

Understanding Entity Manipulation Language (EML)

Wrap-up

IN THIS UNIT, YOU LEARNED

- What Entity Manipulation Language is all about and how it helps to make the development of transactional apps fast and easy

NEXT UNIT

- Week 3 – Unit 5
  Enhancing the BO Behavior with App-Specific Logic

Understanding Entity Manipulation Language (EML)

Further reading

i Additional Material

ABAP RESTful APPLICATION PROGRAMMING MODEL INFORMATION PAGE

For more information, links to documentation, tutorials, and more, please visit the RAP at openSAP information page by following the link below

RAP at openSAP information page (week 3)

Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

Week 3: Enabling the Transactional Behavior of an App

Unit 5: Enhancing the Business Object Behavior with App-Specific Logic

openSAP
open.sap.com

THE BEST RUN SAP

Enhancing the Business Object Behavior with App-Specific Logic

Topics

Week 3

01 The Enhanced Business Scenario

02 Defining the Basic Business Object Behavior

03 Creating the Business Object Behavior Projection

04 Understanding Entity Manipulation Language (EML)

05 Enhancing the BO Behavior with App-Specific Logic

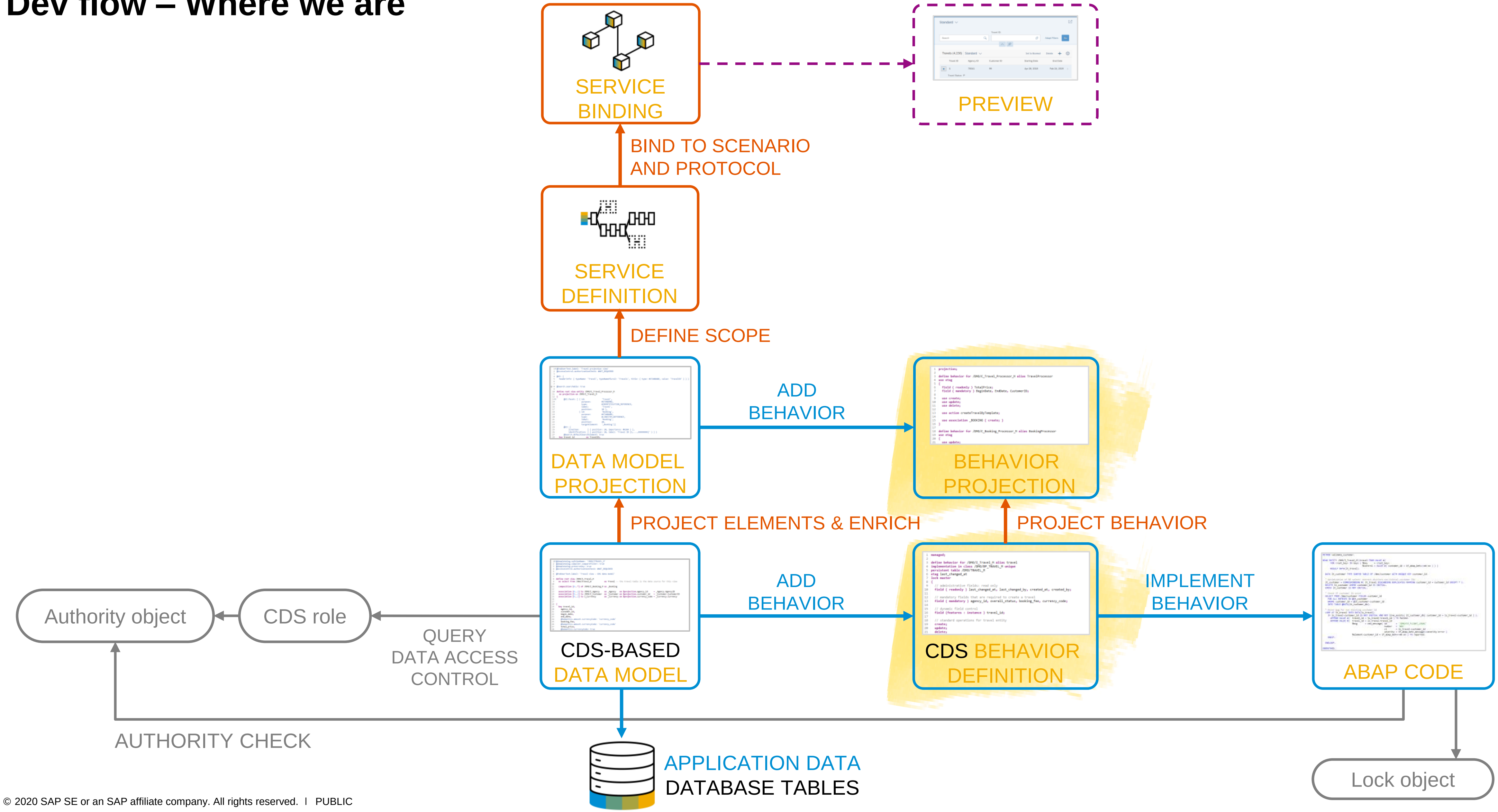
06 Implementing the Business Object Behavior

07 Enabling the Draft Handling

08 Troubleshooting Your SAP Fiori App

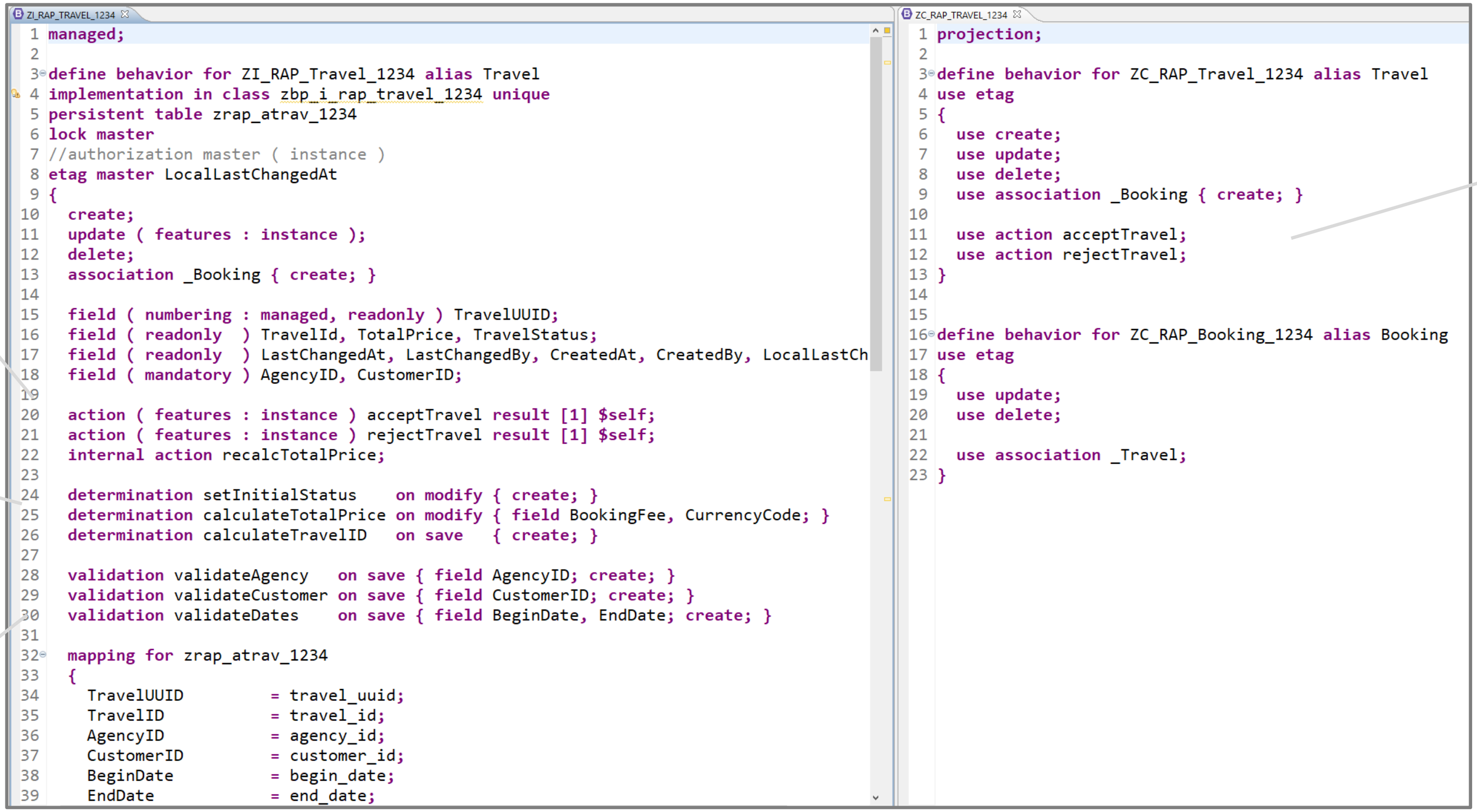
Enhancing the Business Object Behavior with App-Specific Logic

Dev flow – Where we are



Enhancing the Business Object Behavior with App-Specific Logic

What we will achieve in this unit



```
managed;

define behavior for ZI_RAP_Travel_1234 alias Travel
implementation in class zbp_i_rap_travel_1234 unique
persistent table zrap_atrav_1234
lock master
//authorization master ( instance )
etag master LocalLastChangedAt
{
  create;
  update ( features : instance );
  delete;
  association _Booking { create; }

  field ( numbering : managed, readonly ) TravelUUID;
  field ( readonly  ) TravelId, TotalPrice, TravelStatus;
  field ( readonly  ) LastChangedAt, LastChangedBy, CreatedAt, CreatedBy, LocalLastCh
  field ( mandatory ) AgencyID, CustomerID;

  action ( features : instance ) acceptTravel result [1] $self;
  action ( features : instance ) rejectTravel result [1] $self;
  internal action recalcTotalPrice;

  determination setInitialStatus    on modify { create; }
  determination calculateTotalPrice on modify { field BookingFee, CurrencyCode; }
  determination calculateTravelID   on save   { create; }

  validation validateAgency   on save { field AgencyID; create; }
  validation validateCustomer on save { field CustomerID; create; }
  validation validateDates    on save { field BeginDate, EndDate; create; }

  mapping for zrap_atrav_1234
  {
    TravelUUID          = travel_uuid;
    TravelID            = travel_id;
    AgencyID            = agency_id;
    CustomerID          = customer_id;
    BeginDate           = begin_date;
    EndDate             = end_date;
```

```
projection;

define behavior for ZC_RAP_Travel_1234 alias Travel
use etag
{
  use create;
  use update;
  use delete;
  use association _Booking { create; }

  use action acceptTravel;
  use action rejectTravel;
}


define behavior for ZC_RAP_Booking_1234 alias Booking
use etag
{
  use update;
  use delete;

  use association _Travel;
}
```

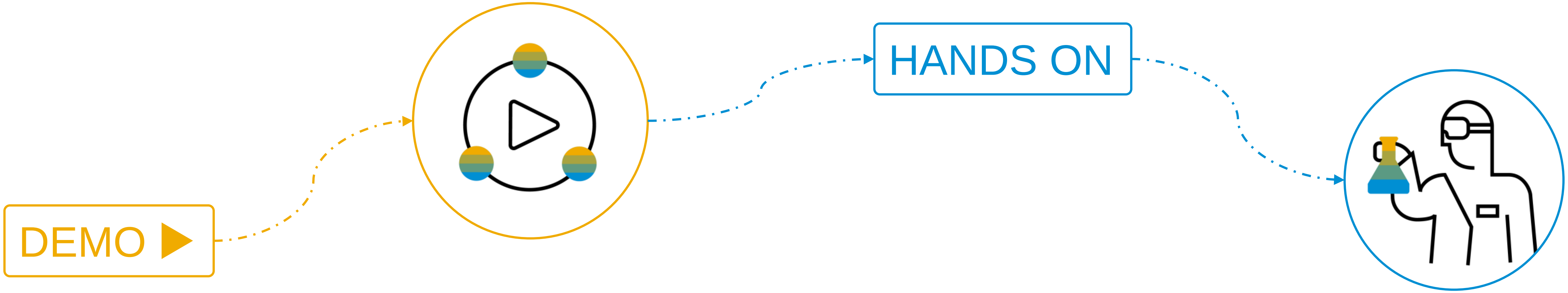
Definition of actions

Definition of determinations

Definition of validations

Projection of actions

Demo



Enhancing the BO Behavior with App-Specific Logic

1. Enhance the BO behavior definition with validation, action, and determination

2. Project the enhanced transactional BO behavior of both entities

3. Test the enhanced SAP Fiori elements App

Enhancing the Business Object Behavior with App-Specific Logic

Wrap-up

IN THIS UNIT, YOU LEARNED

- How to integrate your own app-specific logic to your app

NEXT UNIT

- Week 3 – Unit 6
  Implementing the Business Object Behavior

Enhancing the Business Object Behavior with App-Specific Logic

Further reading

i Additional Material

ABAP RESTful APPLICATION PROGRAMMING MODEL INFORMATION PAGE

For more information, links to documentation, tutorials, and more, please visit the RAP at openSAP information page by following the link below

RAP at openSAP information page (week 3)

Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platforms, directions, and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

THE BEST RUN SAP

Week 3: Enabling the Transactional Behavior of an App

Unit 6: Implementing the Business Object Behavior

Implementing the Business Object Behavior

Topics

Week 3

01 The Enhanced Business Scenario

02 Defining the Basic Business Object Behavior

03 Creating the Business Object Behavior Projection

04 Understanding Entity Manipulation Language (EML)

05 Enhancing the BO Behavior with App-Specific Logic

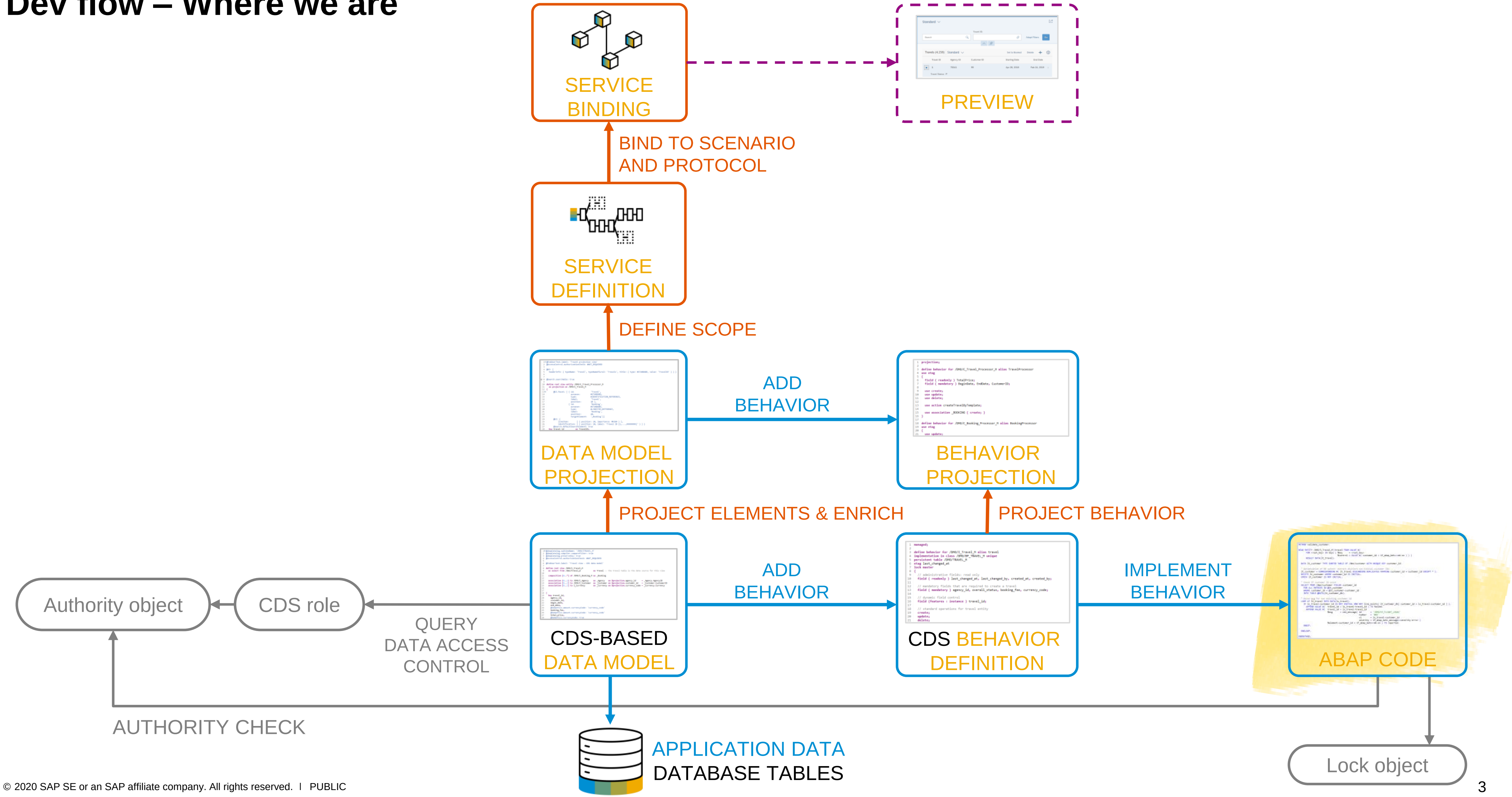
06 Implementing the Business Object Behavior

07 Enabling the Draft Handling

08 Troubleshooting Your SAP Fiori App

Implementing the Business Object Behavior

Dev flow – Where we are



SERVICE BINDING

PREVIEW

BIND TO SCENARIO AND PROTOCOL

SERVICE DEFINITION

DEFINE SCOPE

DATA MODEL PROJECTION

ADD BEHAVIOR

BEHAVIOR PROJECTION

```
projection;

define behavior for /DMO/C_Travel_Processor_M alias TravelProcessor
use etag
{
  field ( readonly ) TotalPrice;
  field ( mandatory ) BeginDate, EndDate, CustomerID;

  use create;
  use update;
  use delete;

  use action createTravelByTemplate;

  use association _BOOKING { create; }
}

define behavior for /DMO/C_Booking_Processor_M alias BookingProcessor
use etag
{
  use update;
```

PROJECT ELEMENTS & ENRICH

PROJECT BEHAVIOR

Authority object

CDS role

QUERY
DATA ACCESS
CONTROL

CDS-BASED DATA MODEL

ADD BEHAVIOR

CDS BEHAVIOR DEFINITION

```
managed;

define behavior for /DMO/I_Travel_M alias travel
implementation in class /DMO/BP_TRAVEL_M unique
persistent table /DMO/TRAVEL_M
etag last_changed_at
lock master
{
  // administrative fields: read only
  field ( readonly ) last_changed_at, last_changed_by, created_at, created_by;

  // mandatory fields that are required to create a travel
  field ( mandatory ) agency_id, overall_status, booking_fee, currency_code;

  // dynamic field control
  field (features : instance ) travel_id;

  // standard operations for travel entity
  create;
  update;
  delete;
```

IMPLEMENT BEHAVIOR

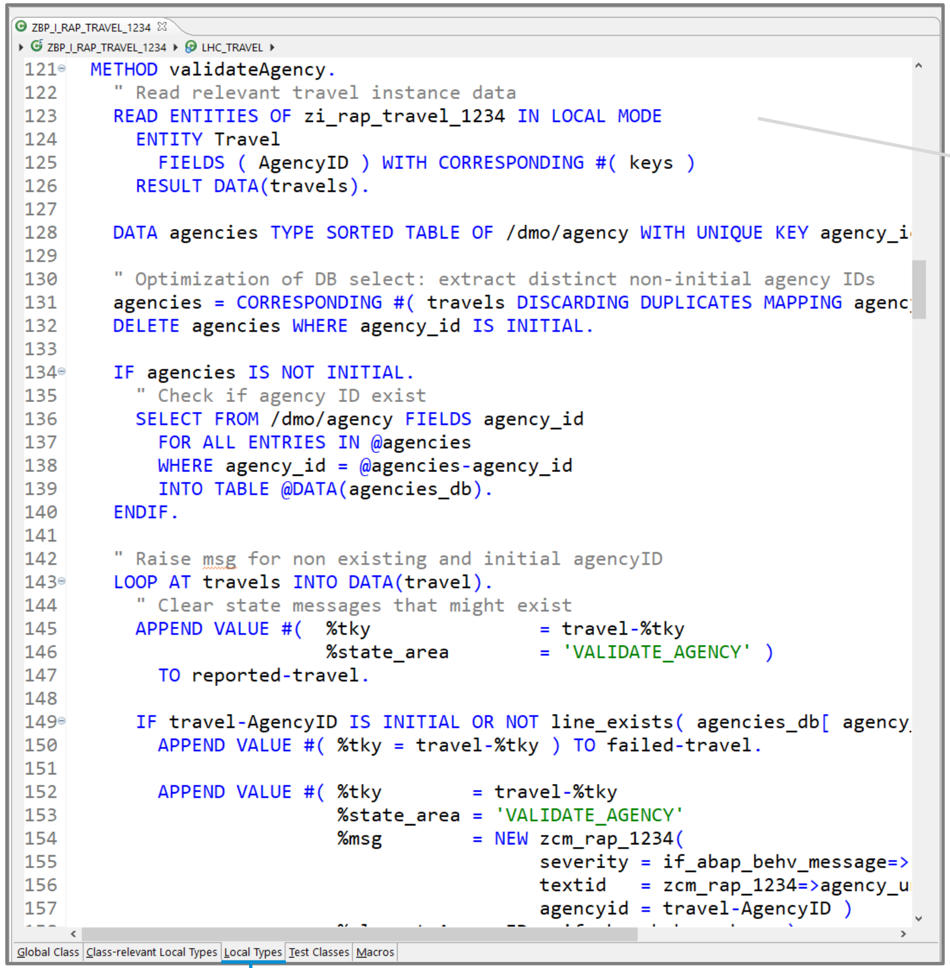
ABAP CODE

AUTHORITY CHECK

APPLICATION DATA
DATABASE TABLES

Lock object

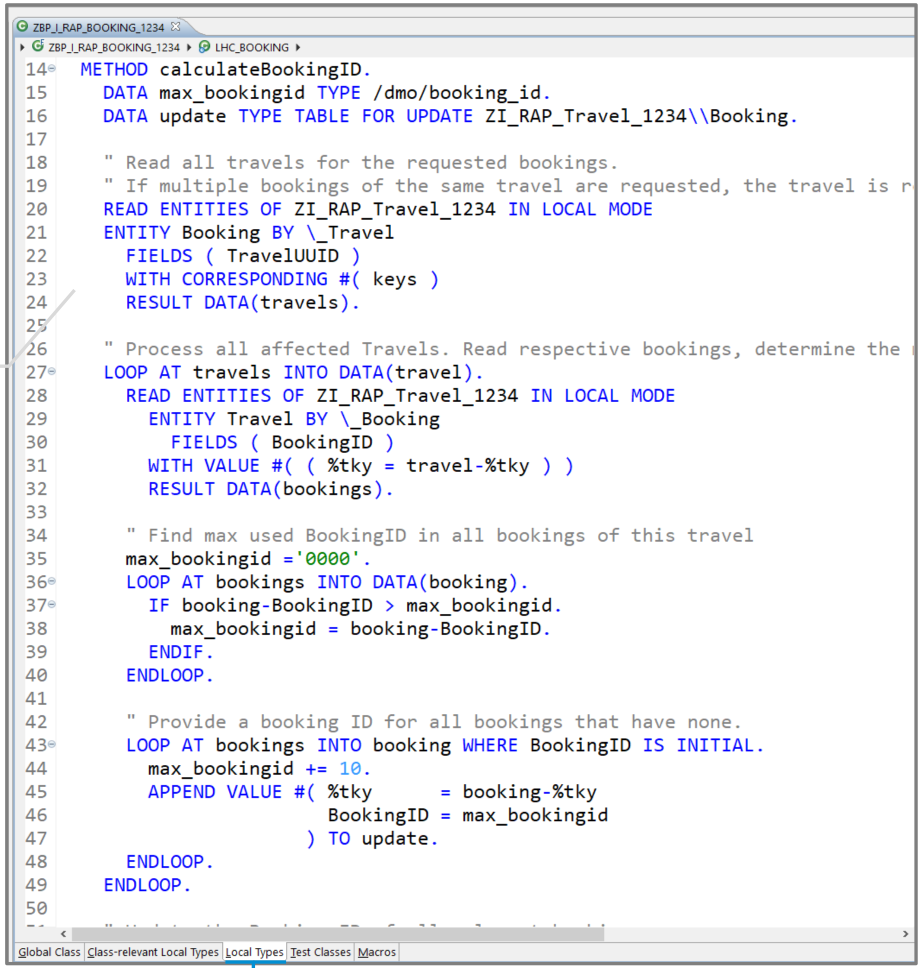
What we will achieve in this unit


ZBP_I_RAP_TRAVEL_1234
ZBP_I_RAP_TRAVEL_1234 ▸ LHC_TRAVEL ▸
Global Class | Class-relevant Local Types | Local Types | Test Classes | Macros

```
METHOD validateAgency.
  " Read relevant travel instance data
  READ ENTITIES OF zi_rap_travel_1234 IN LOCAL MODE
    ENTITY Travel
      FIELDS ( AgencyID ) WITH CORRESPONDING #( keys )
    RESULT DATA(travels).

  DATA agencies TYPE SORTED TABLE OF /dmo/agency WITH UNIQUE KEY agency_i

  " Optimization of DB select: extract distinct non-initial agency IDs
  agencies = CORRESPONDING #( travels DISCARDING DUPLICATES MAPPING agenc
  DELETE agencies WHERE agency_id IS INITIAL.

  IF agencies IS NOT INITIAL.
    " Check if agency ID exist
    SELECT FROM /dmo/agency FIELDS agency_id
      FOR ALL ENTRIES IN @agencies
      WHERE agency_id = @agencies-agency_id
      INTO TABLE @DATA(agencies_db).
  ENDIF.

  " Raise msg for non existing and initial agencyID
  LOOP AT travels INTO DATA(travel).
    " Clear state messages that might exist
    APPEND VALUE #(  %tky               = travel-%tky
                     %state_area        = 'VALIDATE_AGENCY' )
      TO reported-travel.

    IF travel-AgencyID IS INITIAL OR NOT line_exists( agencies_db[ agency
      APPEND VALUE #( %tky = travel-%tky ) TO failed-travel.

      APPEND VALUE #( %tky        = travel-%tky
                      %state_area = 'VALIDATE_AGENCY'
                      %msg        = NEW zcm_rap_1234(
                                        severity = if_abap_behv_message=>
                                        textid   = zcm_rap_1234=>agency_u
                                        agencyid = travel-AgencyID )
```

Implementation of validation

Implementation of determination


ZBP_I_RAP_BOOKING_1234
ZBP_I_RAP_BOOKING_1234 ▸ LHC_BOOKING ▸
Global Class | Class-relevant Local Types | Local Types | Test Classes | Macros

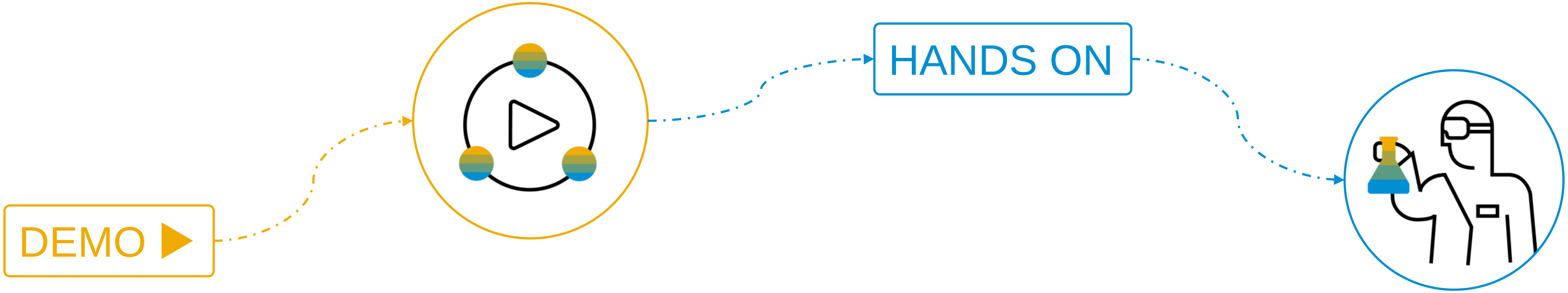
```
METHOD calculateBookingID.
  DATA max_bookingid TYPE /dmo/booking_id.
  DATA update TYPE TABLE FOR UPDATE ZI_RAP_Travel_1234\\Booking.

  " Read all travels for the requested bookings.
  " If multiple bookings of the same travel are requested, the travel is r
  READ ENTITIES OF ZI_RAP_Travel_1234 IN LOCAL MODE
  ENTITY Booking BY \_Travel
    FIELDS ( TravelUUID )
    WITH CORRESPONDING #( keys )
    RESULT DATA(travels).

  " Process all affected Travels. Read respective bookings, determine the
  LOOP AT travels INTO DATA(travel).
    READ ENTITIES OF ZI_RAP_Travel_1234 IN LOCAL MODE
      ENTITY Travel BY \_Booking
        FIELDS ( BookingID )
      WITH VALUE #( ( %tky = travel-%tky ) )
      RESULT DATA(bookings).

    " Find max used BookingID in all bookings of this travel
    max_bookingid ='0000'.
    LOOP AT bookings INTO DATA(booking).
      IF booking-BookingID > max_bookingid.
        max_bookingid = booking-BookingID.
      ENDIF.
    ENDLOOP.

    " Provide a booking ID for all bookings that have none.
    LOOP AT bookings INTO booking WHERE BookingID IS INITIAL.
      max_bookingid += 10.
      APPEND VALUE #( %tky      = booking-%tky
                      BookingID = max_bookingid
                    ) TO update.
    ENDLOOP.
  ENDLOOP.
```

Implementation in local types

Implementation in local types

Implementing the Business Object Behavior

Demo



Implement the BO-specific Behavior

1. Create the BO behavior pool for the travel entity
2. Define and implement validations for the travel entity
3. Implement the dynamic feature control for fields and operations
4. Test the enhanced SAP Fiori elements app

Implementing the Business Object Behavior

Wrap-up

IN THIS UNIT, YOU LEARNED

- How to create the BO behavior pool class and how to implement the validations for your app in ABAP
- How feature control enhances the user experience and how to implement it

NEXT UNIT

- Week 3 – Unit 7
  Enabling the Draft Handling

Implementing the Business Object Behavior

Further reading

i Additional Material

ABAP RESTful APPLICATION PROGRAMMING MODEL INFORMATION PAGE

For more information, links to documentation, tutorials, and more, please visit the RAP at openSAP information page by following the link below

RAP at openSAP information page (week 3)

Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platforms, directions, and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

THE BEST RUN SAP

Week 3: Enabling the Transactional Behavior of an App

Unit 7: Enabling the Draft Handling

openSAP
open.sap.com

THE BEST RUN SAP

Enabling the Draft Handling

Topics

Week 3

01 The Enhanced Business Scenario

02 Defining the Basic Business Object Behavior

03 Creating the Business Object Behavior Projection

04 Understanding Entity Manipulation Language (EML)

05 Enhancing the BO Behavior with App-Specific Logic

06 Implementing the Business Object Behavior

07 Enabling the Draft Handling

08 Troubleshooting Your SAP Fiori App

Enabling the Draft Handling

RESTful architecture and draft handling

ENABLERS FOR CLOUD AND MODERN UX

CLOUD ENVIRONMENT EXPECTS HIGH AVAILABILITY, CONTINUOUS DELIVERY, AND LOW TCO

RESTful avoids problematic server stickiness by introducing a stateless communication

Draft fills the gap between stateless communication and stateful application

MODERN USER EXPERIENCE REQUIRES MULTI-DEVICE SUPPORT AND DATA LOSS PREVENTION WITHOUT CONNECTION TIMEOUTS

Draft persists the state device-independently in a non-process-relevant way

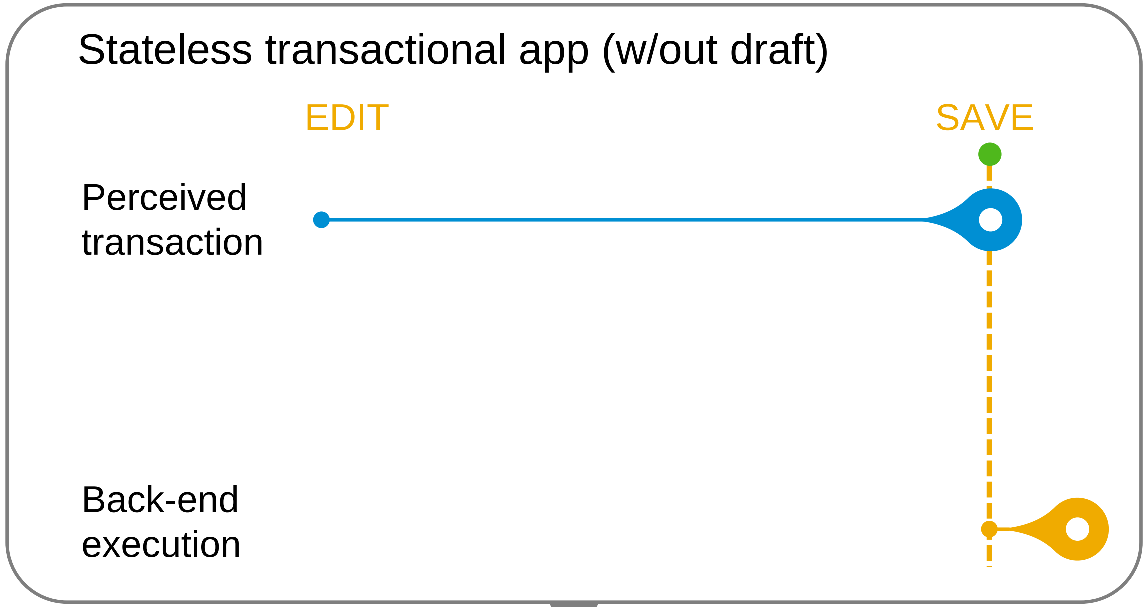
RESTful makes the draft available as an addressable resource

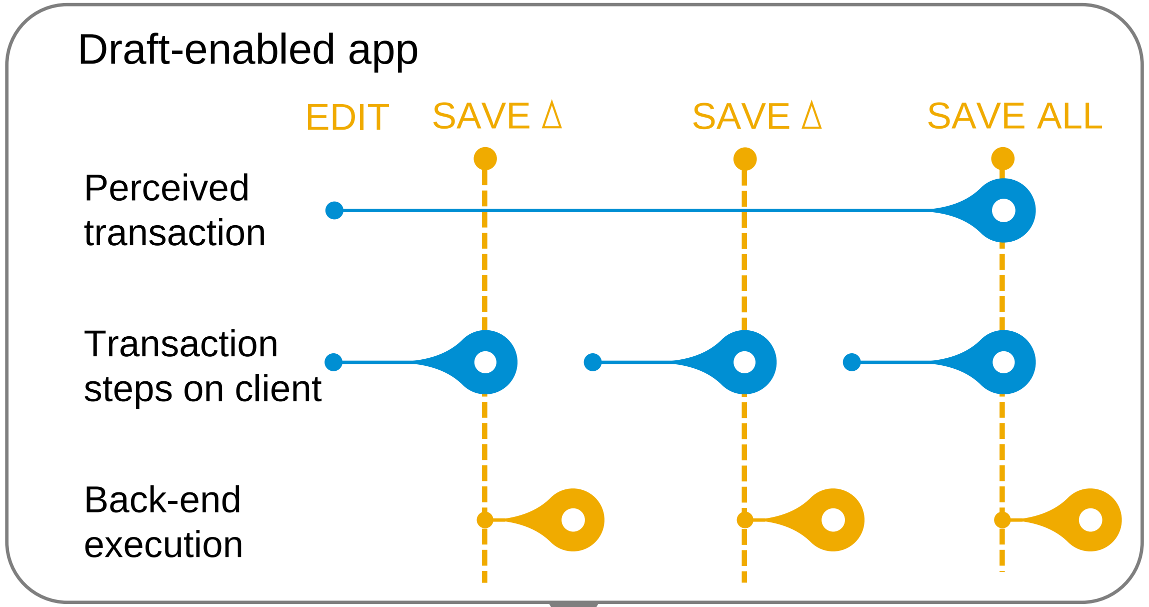
GUIDING PRINCIPLE

Draft is the persisted transactional buffer

Enabling the Draft Handling

Draft – Stateful applications with stateless communication protocol





IMPACT
for end user

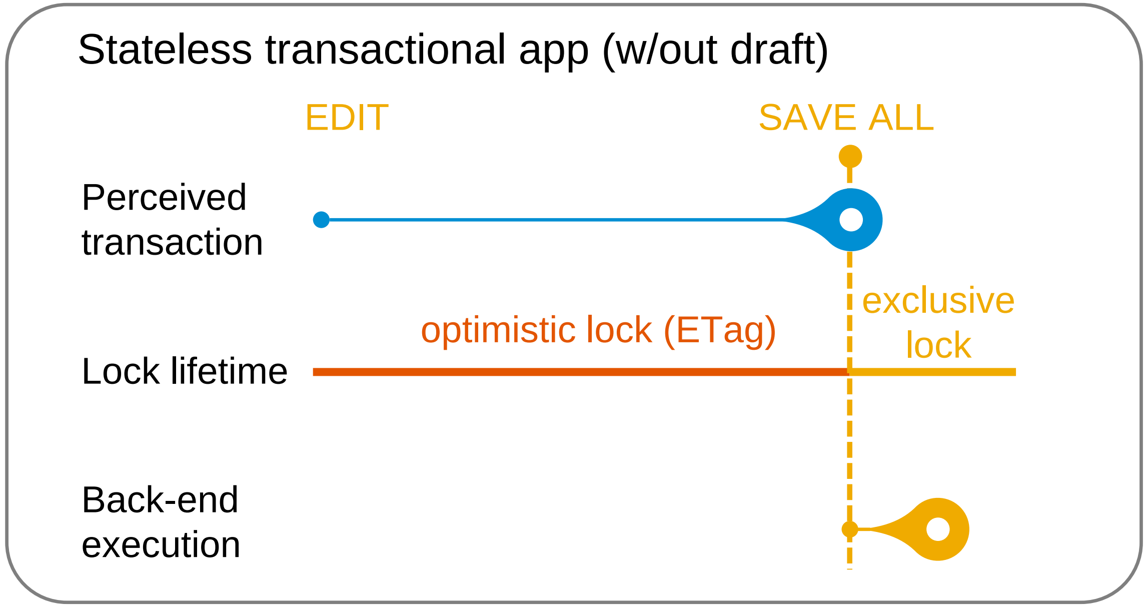
No feedback (e.g. messages, feature control) until SAVE is triggered

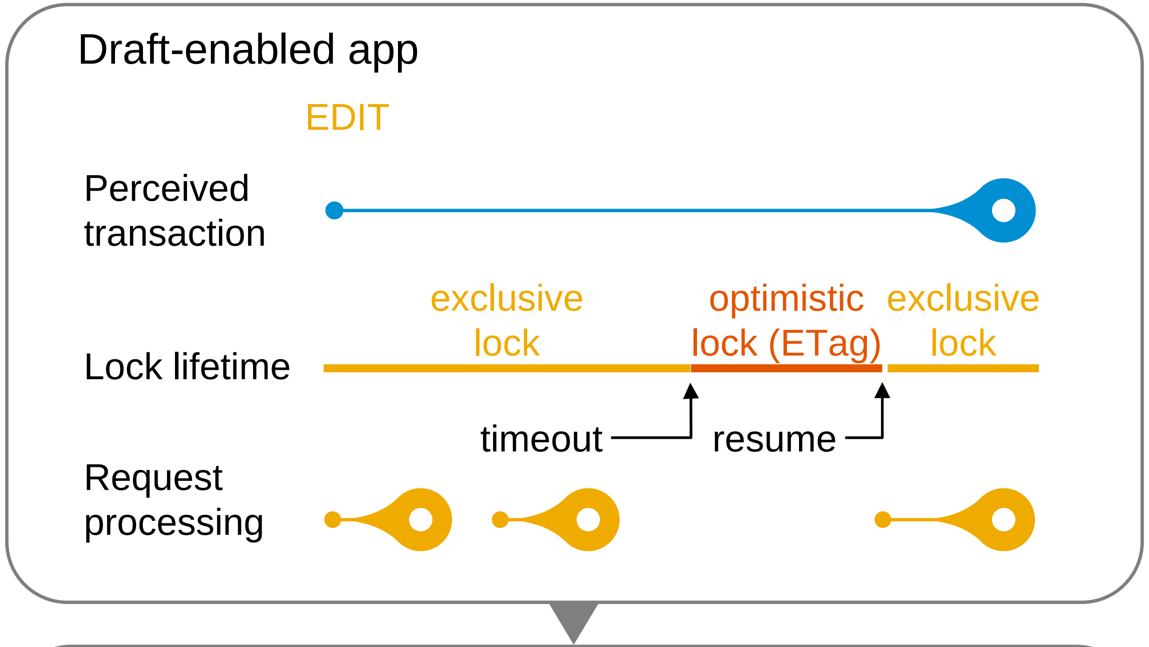
IMPACT
for end user

Early feedback from calculations and validations in the back end including feature control

Draft automatically supports data loss prevention, continuous work, and device switch

Draft – Impact for concurrency handling: Locks





End user perceived transaction distributed over a long period, so locks can‘t be held all the time

Locks are not bound to a session and have their own timeout

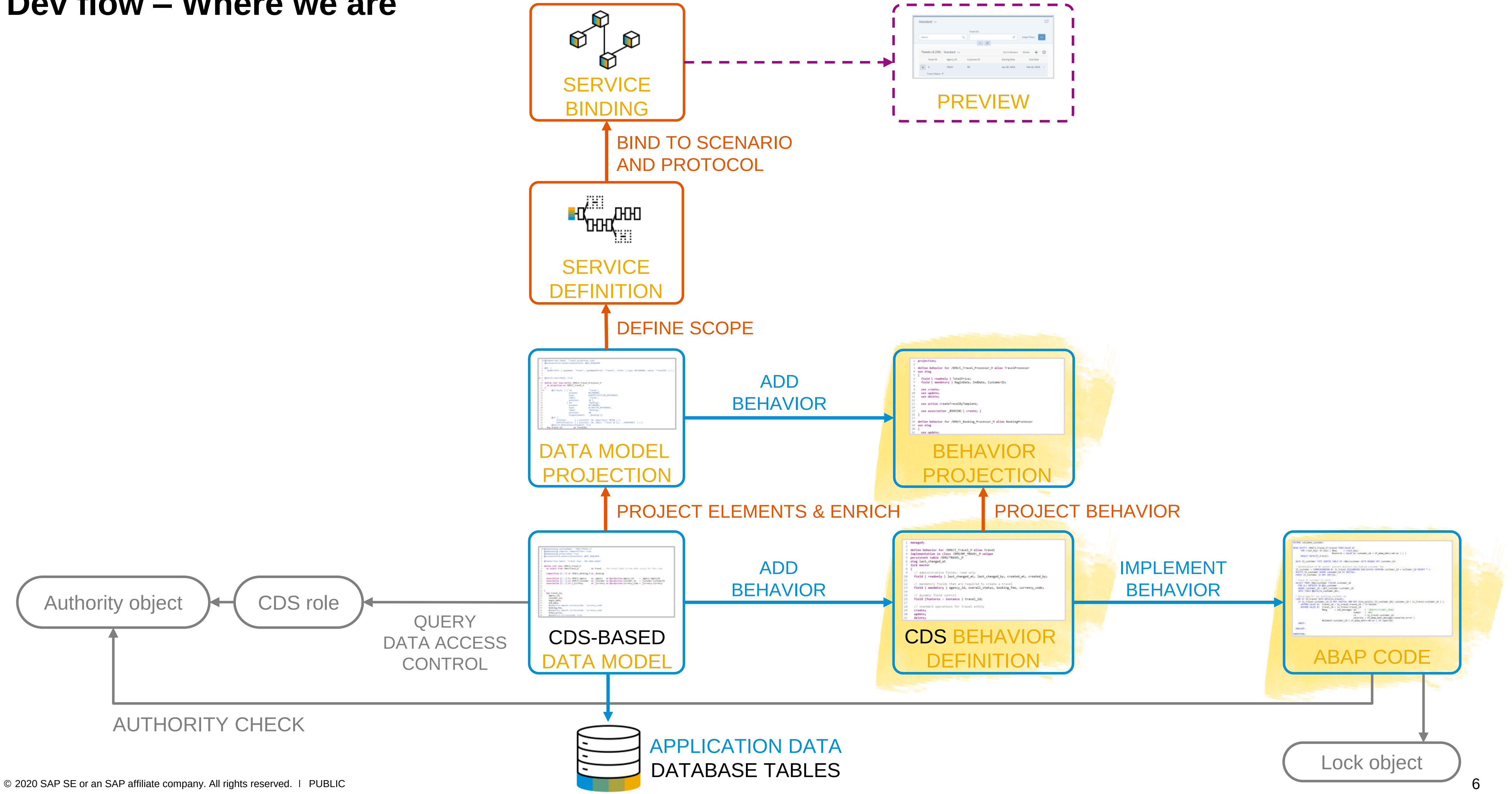
After timeout of exclusive locks an optimistic lock is used

Resume re-acquires the exclusive locks

Request processing is executed in a single ABAP session

Enabling the Draft Handling

Dev flow – Where we are



Enabling the Draft Handling

What we will achieve in this unit

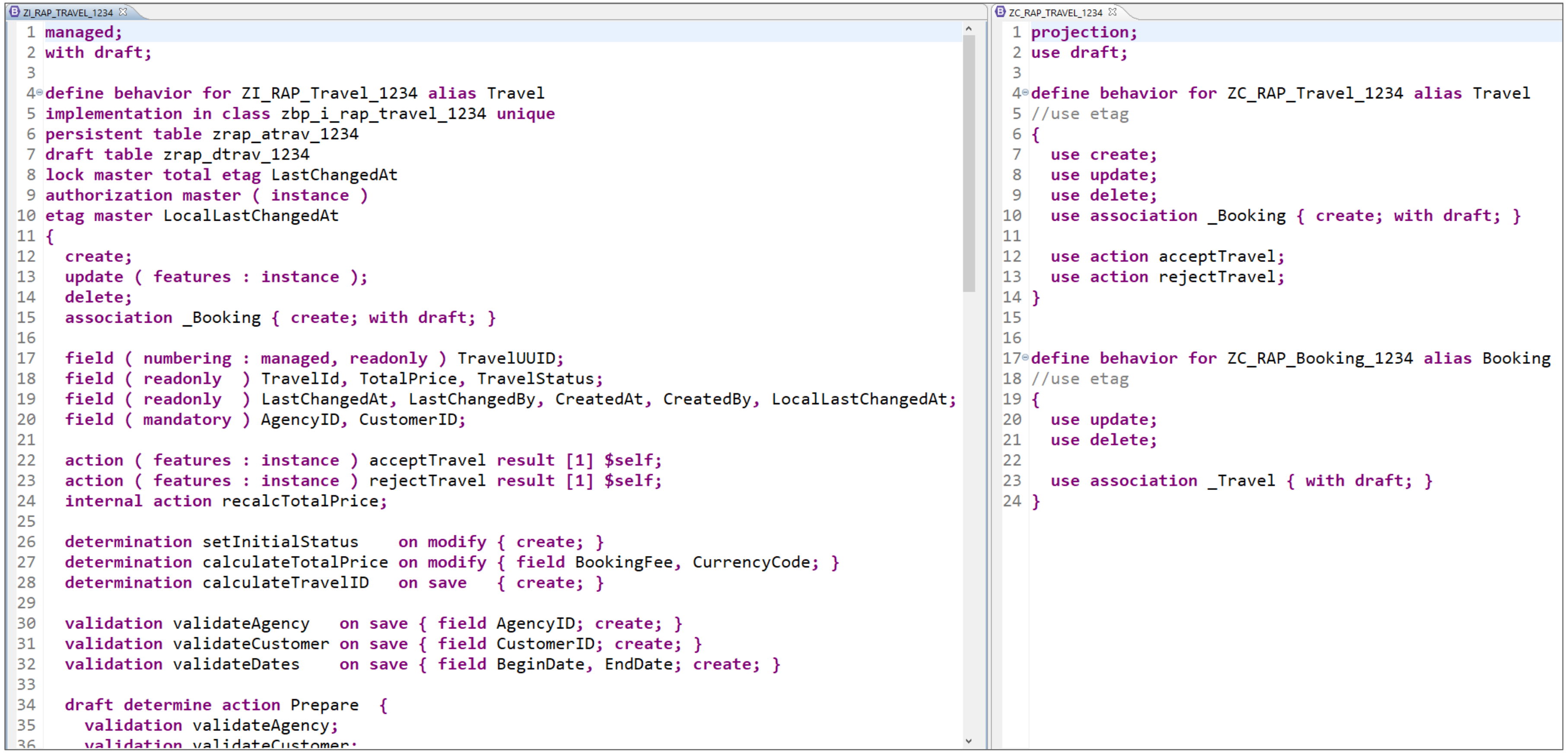
Enable draft for this business object

Define draft table

Define total etag

Enable draft for associations

Projection draft enablement

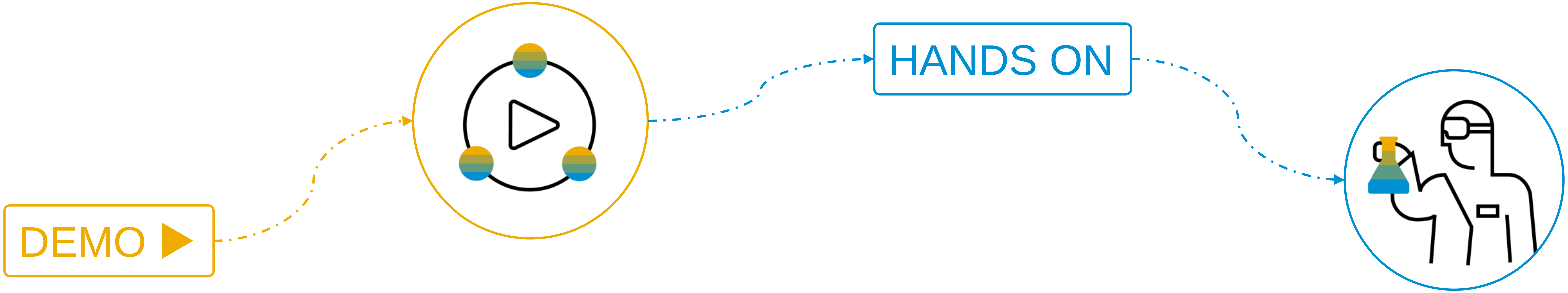


```
managed;
with draft;

define behavior for ZI_RAP_Travel_1234 alias Travel
implementation in class zbp_i_rap_travel_1234 unique
persistent table zrap_atrav_1234
draft table zrap_dtrav_1234
lock master total etag LastChangedAt
authorization master ( instance )
etag master LocalLastChangedAt
{
  create;
  update ( features : instance );
  delete;
  association _Booking { create; with draft; }

  field ( numbering : managed, readonly ) TravelUUID;
  field ( readonly  ) TravelId, TotalPrice, TravelStatus;
  field ( readonly  ) LastChangedAt, LastChangedBy, CreatedAt, CreatedBy, LocalLastChangedAt;
  field ( mandatory ) AgencyID, CustomerID;

  action ( features : instance ) acceptTravel result [1] $self;
  action ( features : instance ) rejectTravel result [1] $self;
  internal action recalcTotalPrice;

  determination setInitialStatus    on modify { create; }
  determination calculateTotalPrice on modify { field BookingFee, CurrencyCode; }
  determination calculateTravelID   on save   { create; }

  validation validateAgency   on save { field AgencyID; create; }
  validation validateCustomer on save { field CustomerID; create; }
  validation validateDates    on save { field BeginDate, EndDate; create; }

  draft determine action Prepare  {
    validation validateAgency;
    validation validateCustomer;
```

```
projection;
use draft;

define behavior for ZC_RAP_Travel_1234 alias Travel
//use etag
{
  use create;
  use update;
  use delete;
  use association _Booking { create; with draft; }

  use action acceptTravel;
  use action rejectTravel;
}


define behavior for ZC_RAP_Booking_1234 alias Booking
//use etag
{
  use update;
  use delete;

  use association _Travel { with draft; }
}
```

Enabling the Draft Handling

Demo



Enable the draft handling in the BO behavior

1. Adjust the CDS data models of the travel and booking entities
2. Enable the draft handling in BO behavior definitions
3. Adjust the business logic in BO behavior implementations
4. Test the enhanced SAP Fiori elements app

Enabling the Draft Handling

Wrap-up

IN THIS UNIT, YOU LEARNED

- The purpose of draft handling
- How to enable and implement the draft handling in an OData-based SAP Fiori UI service

NEXT UNIT

- Week 3 – Unit 8
  Troubleshooting Your SAP Fiori App

Enabling the Draft Handling

Further reading

ABAP RESTful APPLICATION PROGRAMMING MODEL INFORMATION PAGE

For more information, links to documentation, tutorials, and more, please visit the RAP at openSAP information page by following the link below

RAP at openSAP information page (week 3)

Thank you.

Contact information:

open@sap.com

openSAP
open.sap.com

THE BEST RUN SAP

Follow all of SAP

www.sap.com/contactsap

THE BEST RUN SAP

Week 3: Enabling the Transactional Behavior of an App

Unit 8: Troubleshooting Your SAP Fiori App

Troubleshooting Your SAP Fiori App

Topics

Week 3

01 The Enhanced Business Scenario

02 Defining the Basic Business Object Behavior

03 Creating the Business Object Behavior Projection

04 Understanding Entity Manipulation Language (EML)

05 Enhancing the BO Behavior with App-Specific Logic

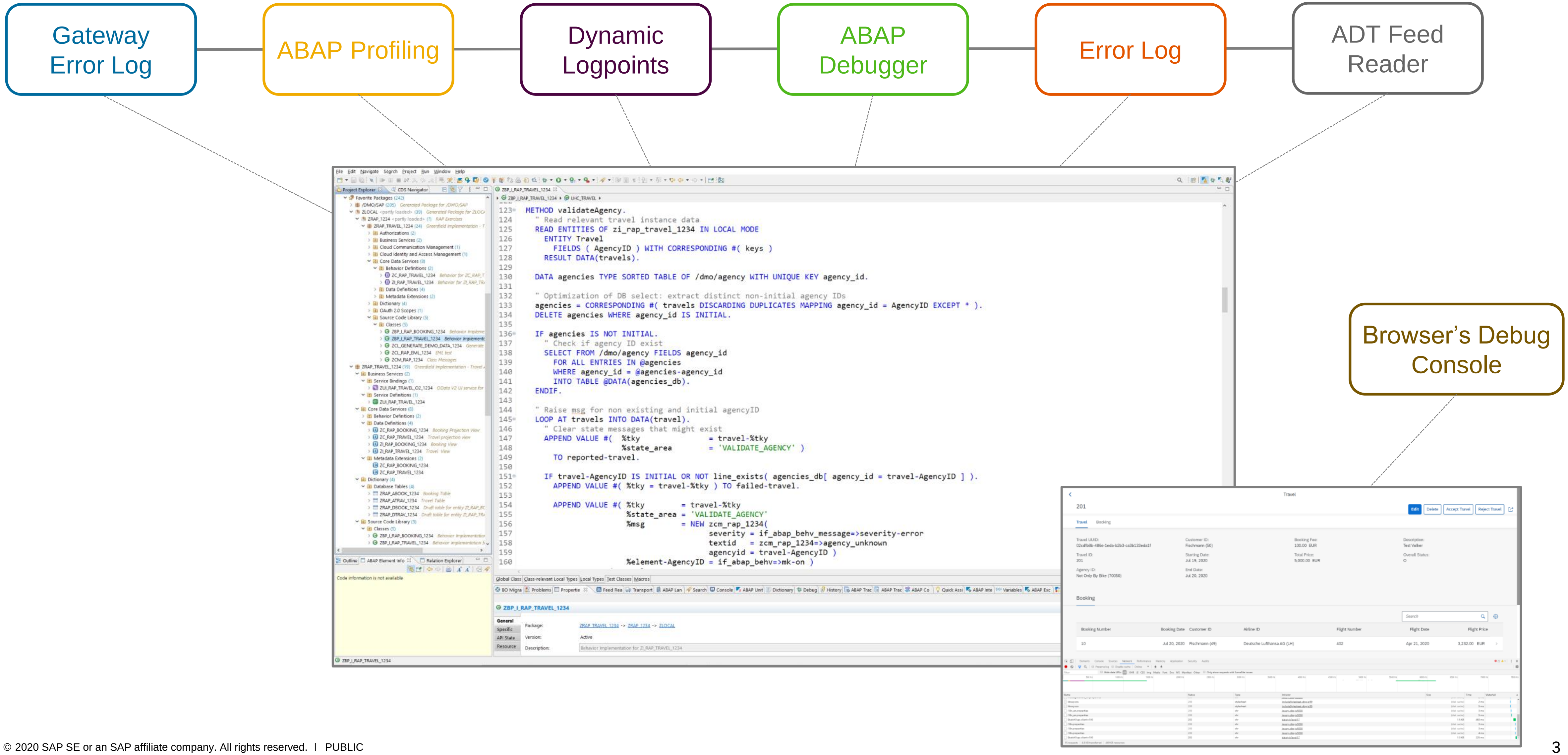
06 Implementing the Business Object Behavior

07 Enabling the Draft Handling

08 Troubleshooting Your SAP Fiori App

Troubleshooting Your SAP Fiori App

ADT troubleshooting tools for ABAP



Gateway Error Log

ABAP Profiling

Dynamic Logpoints

ABAP Debugger

Error Log

ADT Feed Reader

Browser’s Debug Console

ZBP_I_RAP_TRAVEL_1234 ▸ LHC_TRAVEL ▸

```
METHOD validateAgency.
  " Read relevant travel instance data
  READ ENTITIES OF zi_rap_travel_1234 IN LOCAL MODE
    ENTITY Travel
      FIELDS ( AgencyID ) WITH CORRESPONDING #( keys )
    RESULT DATA(travels).

  DATA agencies TYPE SORTED TABLE OF /dmo/agency WITH UNIQUE KEY agency_id.

  " Optimization of DB select: extract distinct non-initial agency IDs
  agencies = CORRESPONDING #( travels DISCARDING DUPLICATES MAPPING agency_id = AgencyID EXCEPT * ).
  DELETE agencies WHERE agency_id IS INITIAL.

  IF agencies IS NOT INITIAL.
    " Check if agency ID exist
    SELECT FROM /dmo/agency FIELDS agency_id
      FOR ALL ENTRIES IN @agencies
      WHERE agency_id = @agencies-agency_id
      INTO TABLE @DATA(agencies_db).
  ENDIF.

  " Raise msg for non existing and initial agencyID
  LOOP AT travels INTO DATA(travel).
    " Clear state messages that might exist
    APPEND VALUE #(  %tky                 = travel-%tky
                     %state_area          = 'VALIDATE_AGENCY' )
      TO reported-travel.

    IF travel-AgencyID IS INITIAL OR NOT line_exists( agencies_db[ agency_id = travel-AgencyID ] ).
      APPEND VALUE #( %tky = travel-%tky ) TO failed-travel.

      APPEND VALUE #( %tky        = travel-%tky
                      %state_area = 'VALIDATE_AGENCY'
                      %msg        = NEW zcm_rap_1234(
                                        severity = if_abap_behv_message=>severity-error
                                        textid   = zcm_rap_1234=>agency_unknown
                                        agencyid = travel-AgencyID )
                      %element-AgencyID = if_abap_behv=>mk-on )
```

ZBP_I_RAP_TRAVEL_1234

Package: ZRAP_TRAVEL_1234 -> ZRAP_1234 -> ZLOCAL

Version: Active

Description: Behavior Implementation for ZI_RAP_TRAVEL_1234

Travel

201

Travel UUID: 02cdfb8b-486e-1eda-b2b3-ca3b133eda1f

Travel ID: 201

Agency ID: Not Only By Bike (70050)

Customer ID: Fischmann (50)

Starting Date: Jul 19, 2020

End Date: Jul 20, 2020

Booking Fee: 100.00 EUR

Total Price: 5,000.00 EUR

Description: Test Volker

Overall Status: O

Booking

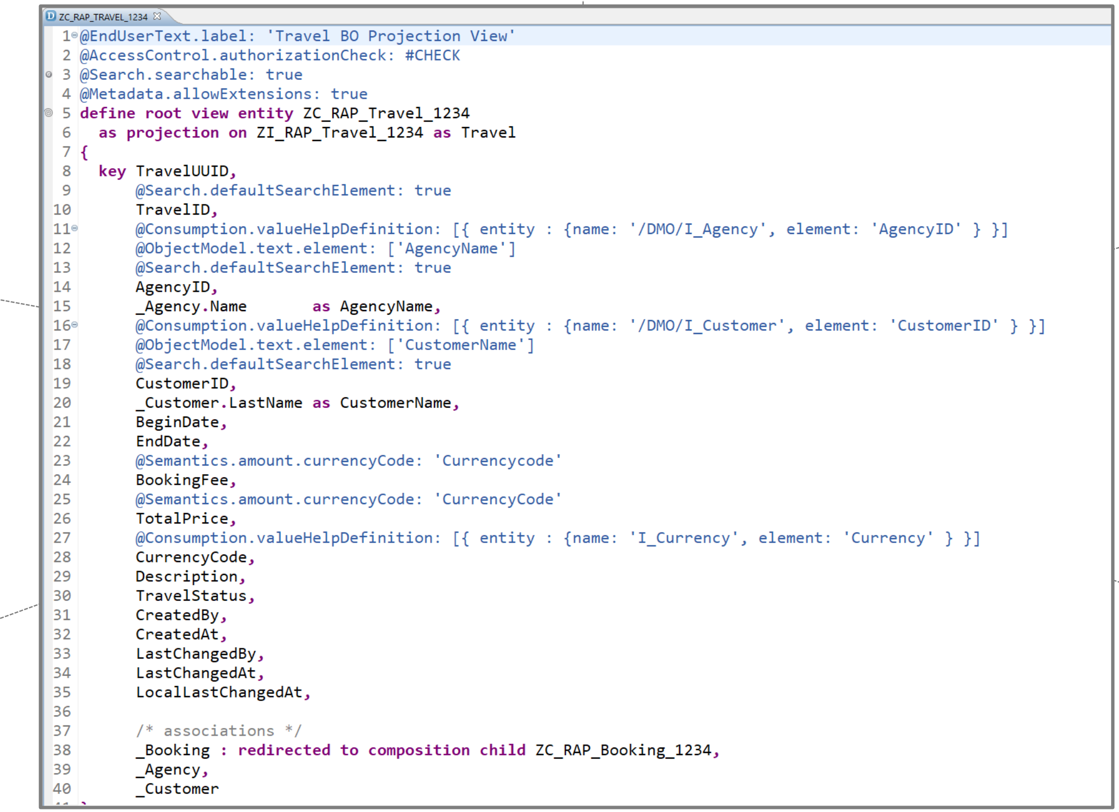
Booking Number	Booking Date	Customer ID	Airline ID	Flight Number	Flight Date	Flight Price
10	Jul 20, 2020	Fischmann (49)	Deutsche Lufthansa AG (LH)	402	Apr 21, 2020	3,232.00 EUR

ADT troubleshooting tools for ABAP CDS

Data Preview

Activation Graph

Dictionary Log

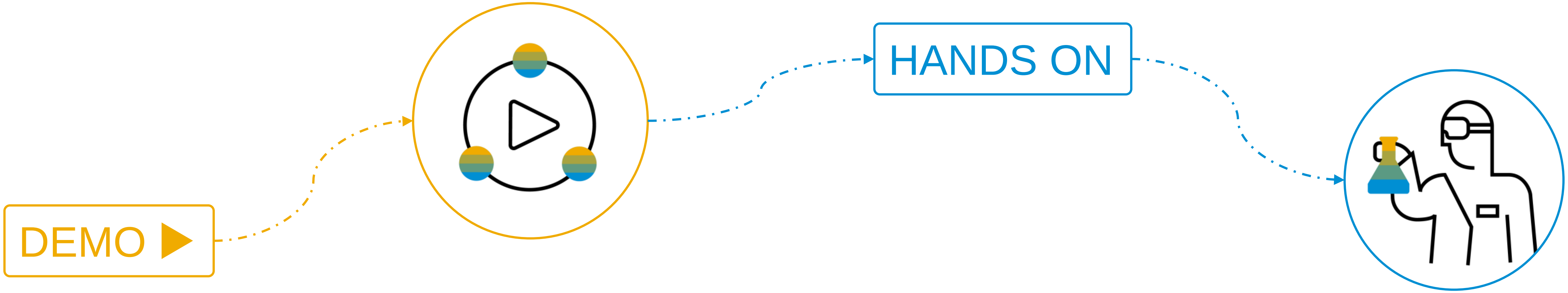


```
@EndUserText.label: 'Travel BO Projection View'
@AccessControl.authorizationCheck: #CHECK
@Search.searchable: true
@Metadata.allowExtensions: true
define root view entity ZC_RAP_Travel_1234
  as projection on ZI_RAP_Travel_1234 as Travel
{
  key TravelUUID,
      @Search.defaultSearchElement: true
      TravelID,
      @Consumption.valueHelpDefinition: [{ entity : {name: '/DMO/I_Agency', element: 'AgencyID' } }]
      @ObjectModel.text.element: ['AgencyName']
      @Search.defaultSearchElement: true
      AgencyID,
      _Agency.Name       as AgencyName,
      @Consumption.valueHelpDefinition: [{ entity : {name: '/DMO/I_Customer', element: 'CustomerID' } }]
      @ObjectModel.text.element: ['CustomerName']
      @Search.defaultSearchElement: true
      CustomerID,
      _Customer.LastName as CustomerName,
      BeginDate,
      EndDate,
      @Semantics.amount.currencyCode: 'Currencycode'
      BookingFee,
      @Semantics.amount.currencyCode: 'CurrencyCode'
      TotalPrice,
      @Consumption.valueHelpDefinition: [{ entity : {name: 'I_Currency', element: 'Currency' } }]
      CurrencyCode,
      Description,
      TravelStatus,
      CreatedBy,
      CreatedAt,
      LastChangedBy,
      LastChangedAt,
      LocalLastChangedAt,

      /* associations */
      _Booking : redirected to composition child ZC_RAP_Booking_1234,
      _Agency,
      _Customer
```

Annotation Propagation

Active Annotations

Demo



Troubleshoot Your SAP Fiori App

1. Examples for using ABAP troubleshooting tools
2. How to use CDS troubleshooting tools

Troubleshooting Your SAP Fiori App

Wrap-up

IN THIS UNIT, YOU LEARNED

- What troubleshooting tools are available in ABAP and CDS and how these tools can be used to analyze your app

NEXT UNIT

- Week 4
  Dealing with Existing Code

Troubleshooting Your SAP Fiori App

Further reading

i Additional Material

ABAP RESTful APPLICATION PROGRAMMING MODEL INFORMATION PAGE

For more information, links to documentation, tutorials, and more, please visit the RAP at openSAP information page by following the link below

RAP at openSAP information page (week 3)

Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company.

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platforms, directions, and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

See www.sap.com/copyright for additional trademark information and notices.

THE BEST RUN SAP