

UV 911.411

Enterprise Computing

WS2025/2026

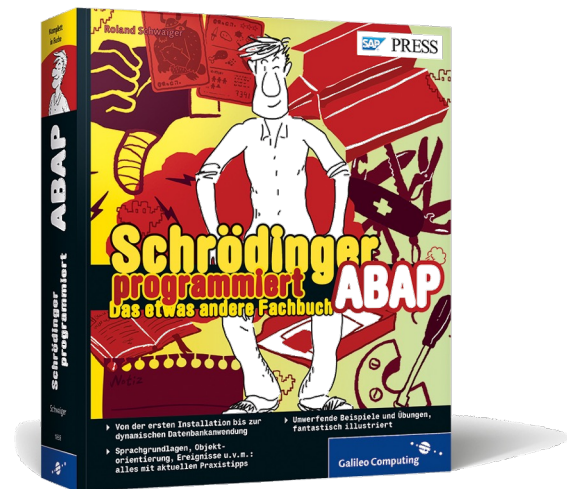
SW-Development with SAP-ABAP

Dr. Schwaiger Roland
NoR GmbH

Introduction



roland.schwaiger@nor.gmbh



The book to read (German)

Dr. Roland Schwaiger

Located

Bad Dürrenberg, Hallein, AT

Background

Mathematics (University Salzburg)

Computer Sciences (University Salzburg, Bowling Green State University)

Project & Process Management (SMBS – University of Salzburg Business School)

Profession

Co-Founder and -CEO **NoR GmbH**

SAP Technical Consultant (Cert. SAP Development Consultant, SAP Cloud Developer)

SAP Trainer

Project Coach (Cert. Scrum Master)

Software Architect

Software Developer (SAP AG, Walldorf, DE and Customer Development Projects)

Author (check out Amazon and/or www.citeseer.com)

Lecturer (University Salzburg, FH Salzburg)

Passion

Learning

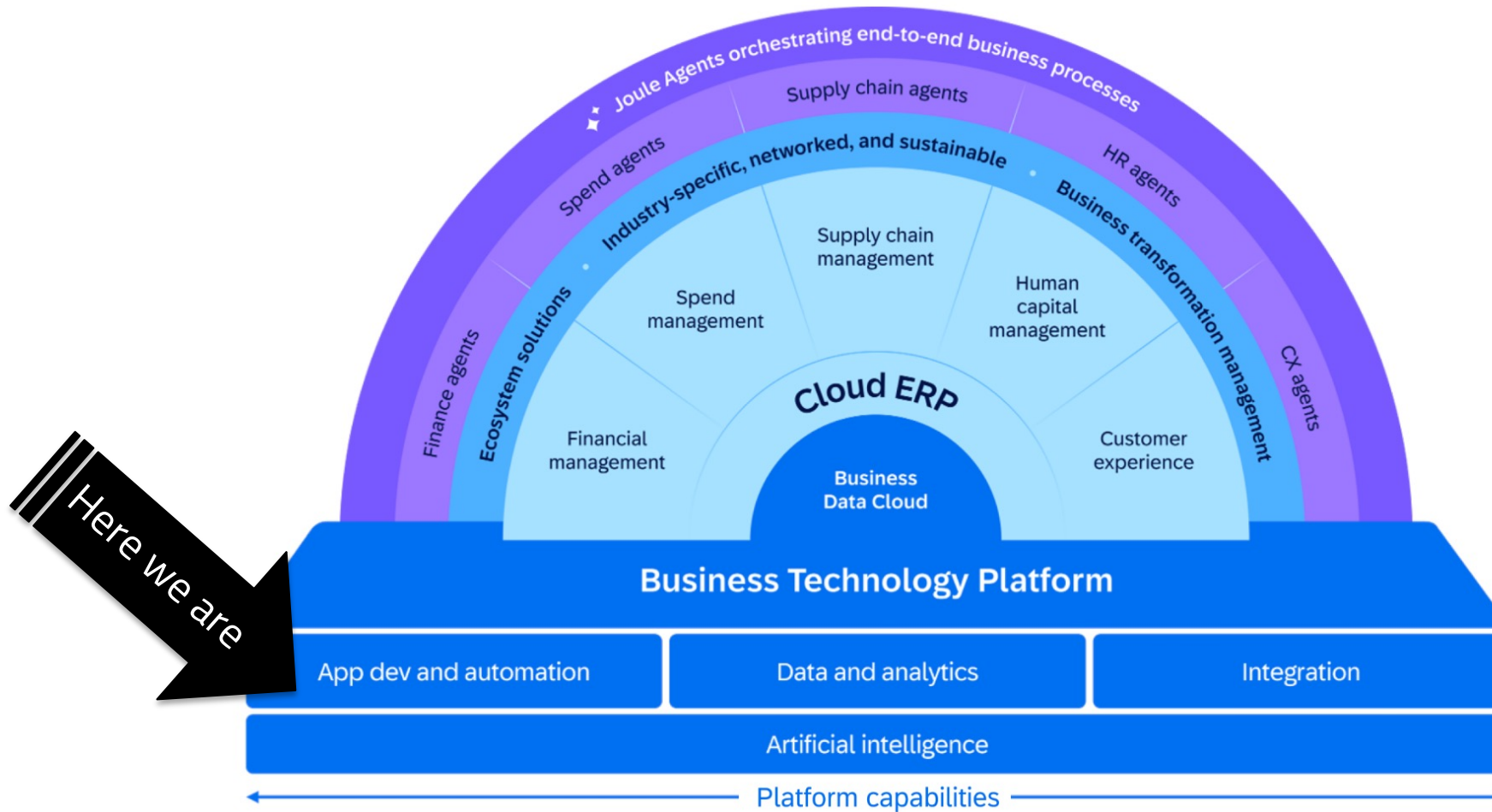
Definition

“ **Enterprise computing** involves the development, deployment and maintenance of the information systems required for survival and success in today's business climate.”

Thus SAP is one instance of the class of enterprise computing

Yen-Ping Shan & Ralph H. Earle,
Enterprise Computing with Objects,
Addison-Wesley, 1998.

Motivation



Course Overview



Infrastructure BTP, Cockpit, Subaccount,
Entitlements, Cloud Foundry, ABAP Environment

ABAP Cloud Program Package, OO
Console Application

Dev Tools Eclipse ADT, Git, BAS, Class Builder,
ABAP Editor, Dictionary

Data Types and Data Object
DATA, TYPES, Tables, Structures, References

ABAP OO Class, Object, Method, Attribute,
Exception

RAP Development CDS, Behavior,
Service, Binding, UI Annotation

Database Create Tables, Read, Update, Delete

Finalizing the Course

The completion of the course takes place within the course block. This consists of:

- Implementation of repository objects as part of the course
- Final examination
 - Theoretical exam with ten questions covering the course content
 - Development of repository objects



Motivation

Motivation

What does SAP do?

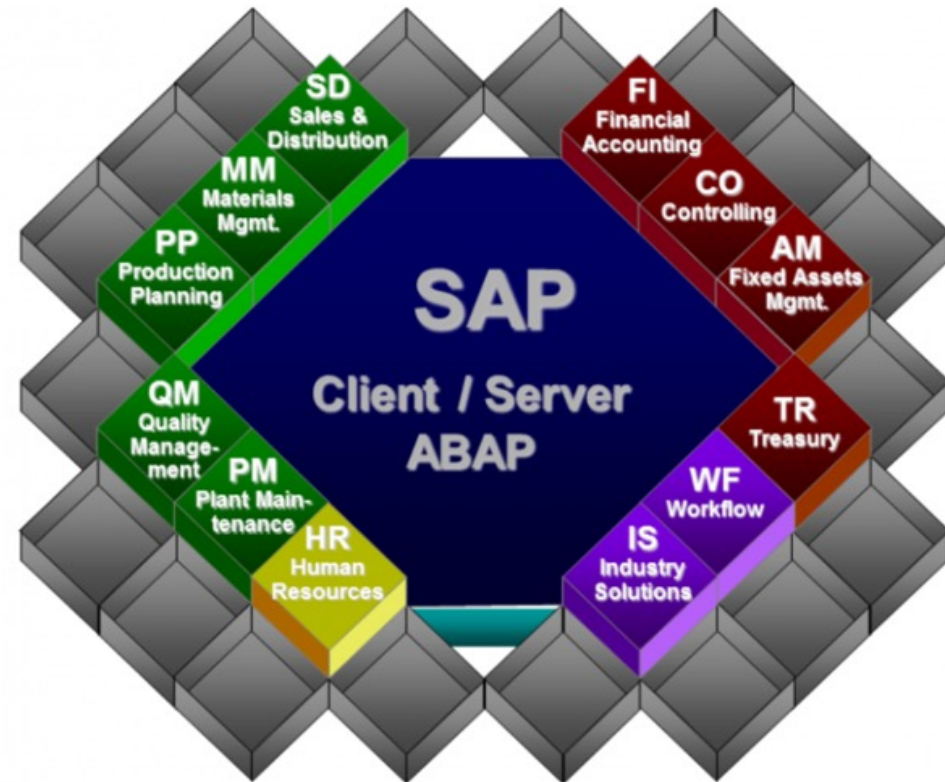
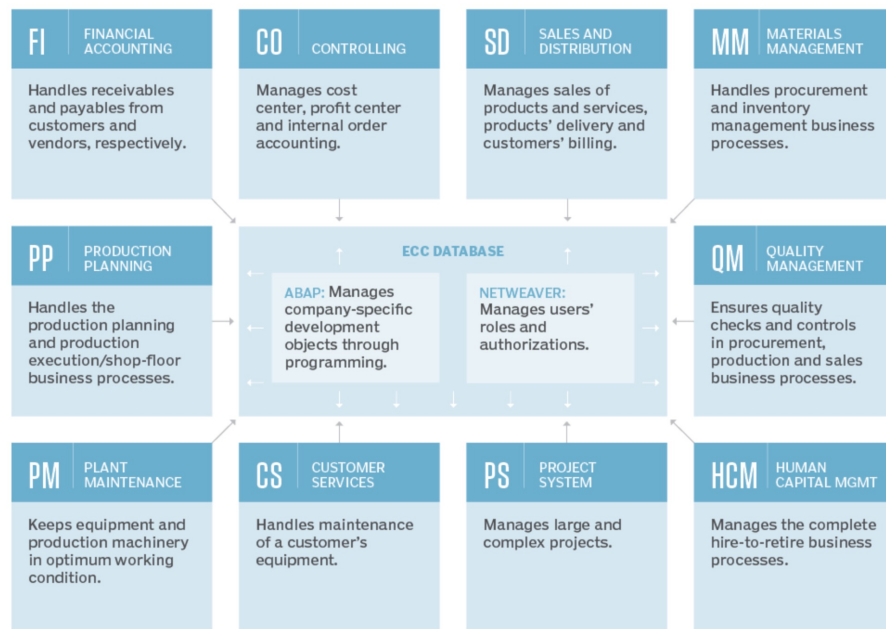


SAP helps companies and organizations of all sizes and industries run their businesses profitably, adapt continuously, and grow sustainably [Definition by SAP].

Motivation SAP ECC

SAP ERP Central Component is the umbrella term for SAP’s functional and technical software components that make up the ERP system.
ECC is also known as SAP ERP.

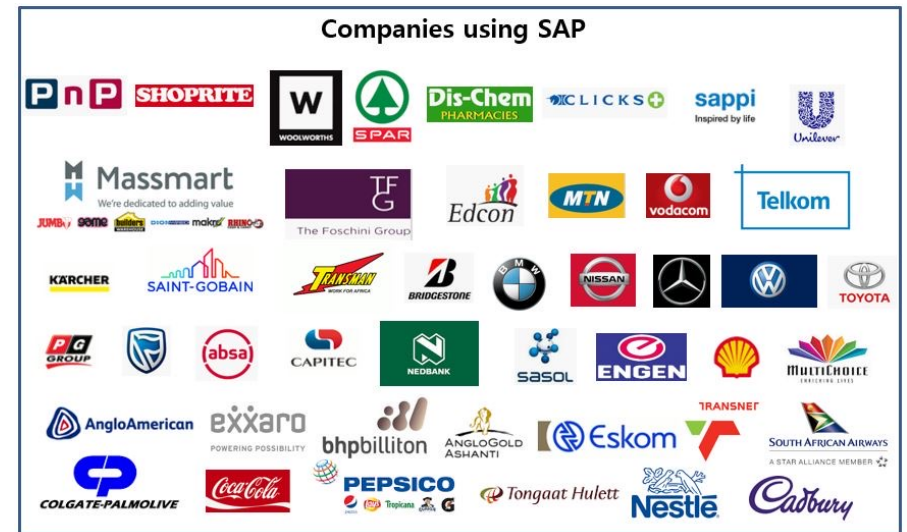
<https://erproof.com/how-does-sap-work/>



<https://www.techtarget.com/searchsap/definition/SAP>

Motivation

Who is using SAP? e.g.



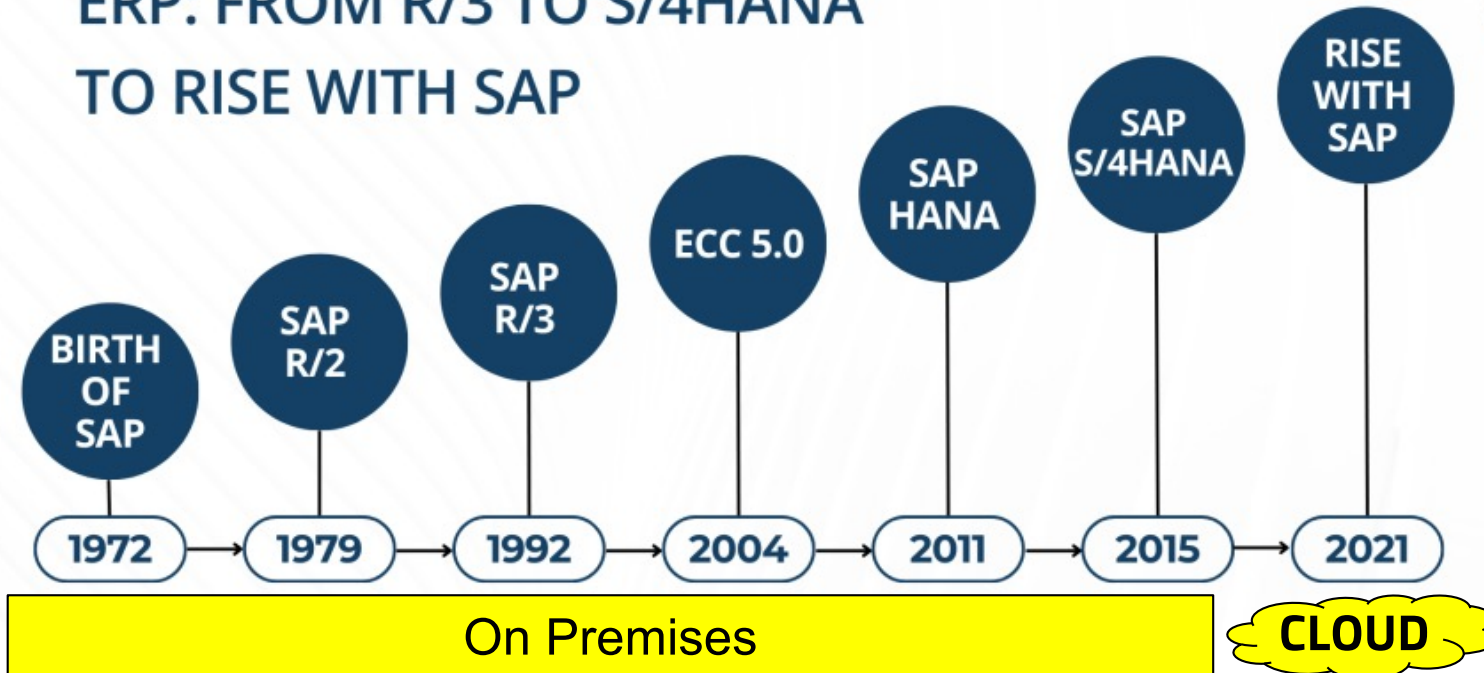
<https://www.thomsondata.com/customer-base/sap.php>

Evolution of SAP



ITEANZ TECHNOLOGIES

...
**THE EVOLUTION OF SAP
ERP: FROM R/3 TO S/4HANA
TO RISE WITH SAP**



Evolution of SAP



- The first version of SAP's **enterprise software** was a financial accounting system named **RF** or better known as **R/1** . (The "R" was for "Real-time data processing ")
- This was replaced by **R/2** at the end of the 1970s. **SAP R/2** was a **mainframe based** business application software suite that was very successful in the 1980s and early 1990s
- The client server based solution **SAP R/3** was official launched on July 6 , 1992. (The 3 stands for the **three layer architecture**)
- In 2008 SAP HANA, an in-memory, column-oriented RDBMS was introduced .

Motivation

SAP



SAP history

- "SAP system analysis and program development" was founded in 1972 by 5 former IBM engineers
- In 1976 the SAP GmbH was founded, they moved to Walldorf
- In 1988 the SAP Aktiengesellschaft *Systems, Applications and Products in Data Processing* was founded

Motivation

SAP History



Metric	Updated SAP Figures (approx.)
Global ranking	Largest software company in Europe; among top globally
Employees (total)	~109,973 (2024)
Revenue (IFRS)	~€34.18 bn (2024), ~€36.8 bn (2025 est.)
Customers	~425,000 – 440,000 worldwide
Countries/Subsidiaries	Operations in 180+ countries
Development Centers	~19 major SAP Labs centers globally

Motivation

SAP







SAP is...

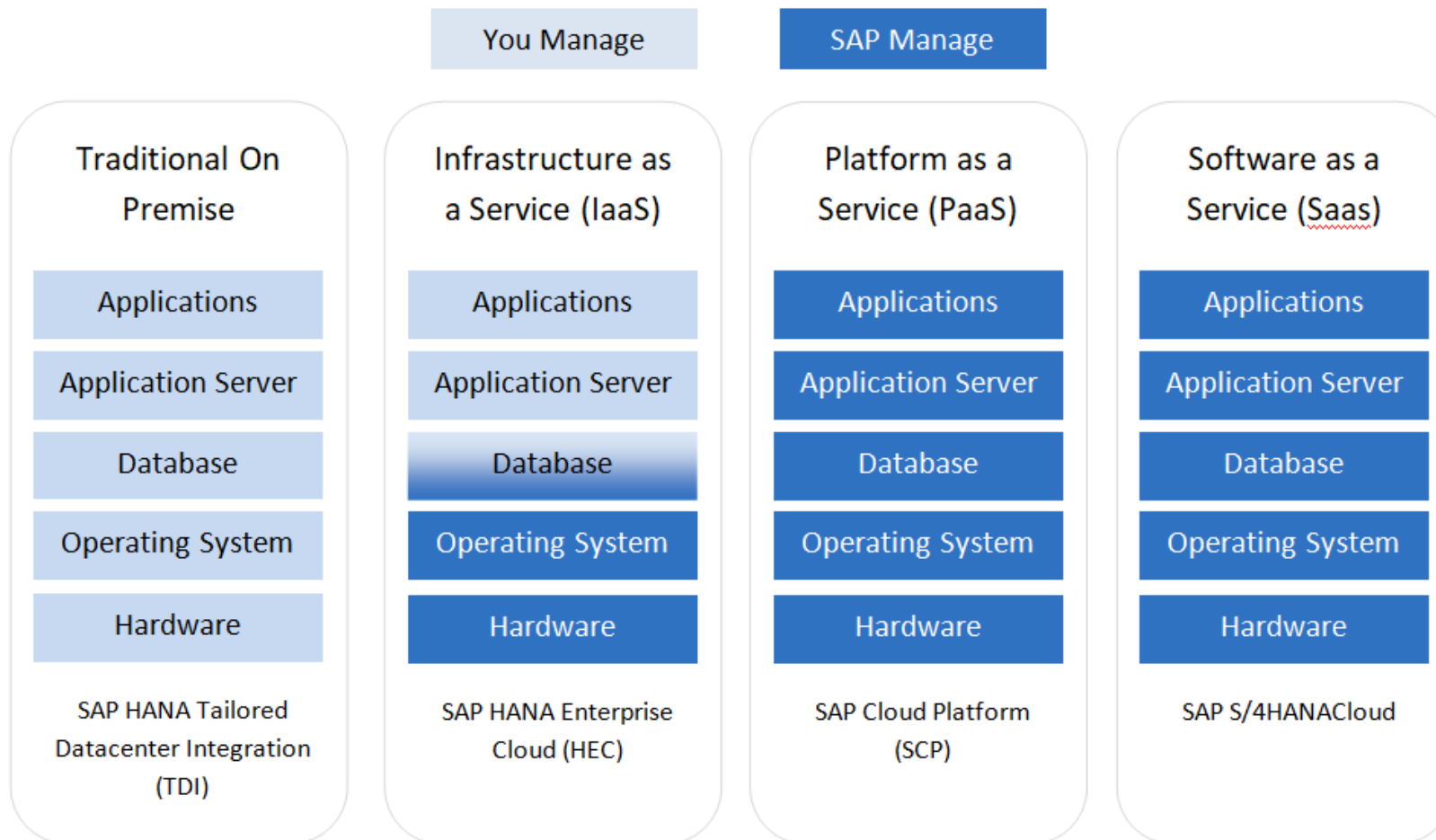
- the corporation SAP
- the family of products of the SAP, referred to as **THE SAP SYSTEMS**

Terms

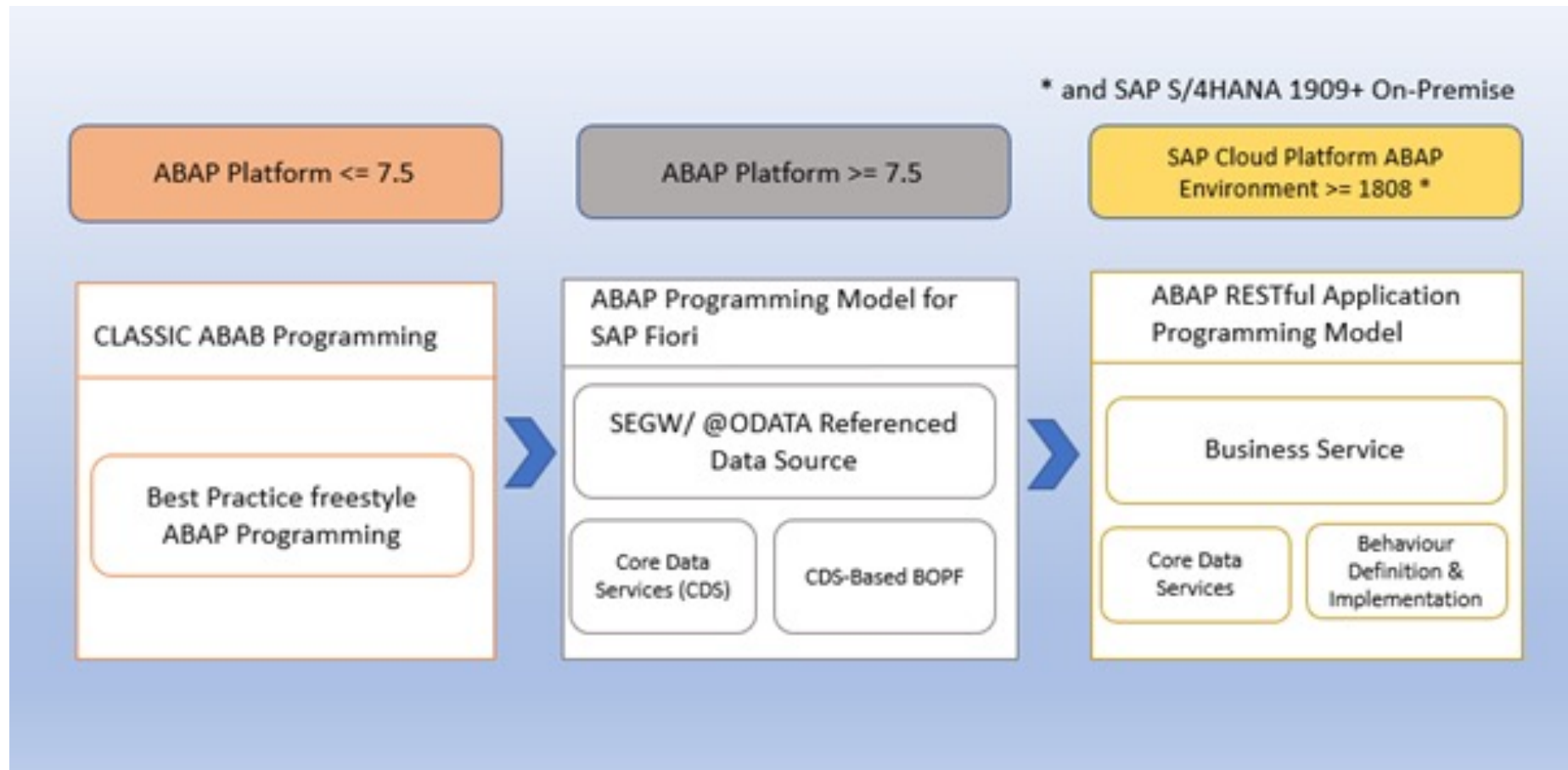
Talking about Cloud

As a Service		As a Product	
Public Cloud	Private Cloud		
		On Premise	
 <p>RISE with SAP S/4HANA Cloud</p> <ul style="list-style-type: none"> • Lowest TCO & highest Cloud value • Comprehensive ERP scope & some industries • Flexibility within standards • RISE business transformation services • Single contract • Implementation: Greenfield 	 <p>SAP S/4HANA Cloud, extended edition</p> <ul style="list-style-type: none"> • Cloud value with full ERP & industry scope • Traditional flexibility with some restrictions (no modifications) • Hosted on SAP HANA Enterprise Cloud • Multiple contracts • Implementation: Greenfield 	 <p>RISE with SAP S/4HANA Cloud, private edition</p> <ul style="list-style-type: none"> • Cloud value with full ERP & industry scope • Traditional flexibility (incl. modifications) • Hosted on SAP HANA Enterprise Cloud • RISE business transformation services • Single Contract • Implementation: Brownfield, Greenfield 	 <p>SAP S/4HANA On Premise</p> <ul style="list-style-type: none"> • Total control and customization with full ERP & industry scope • Traditional flexibility • Multiple contracts • Implementation: Brownfield, Greenfield

Talking about Responsibility

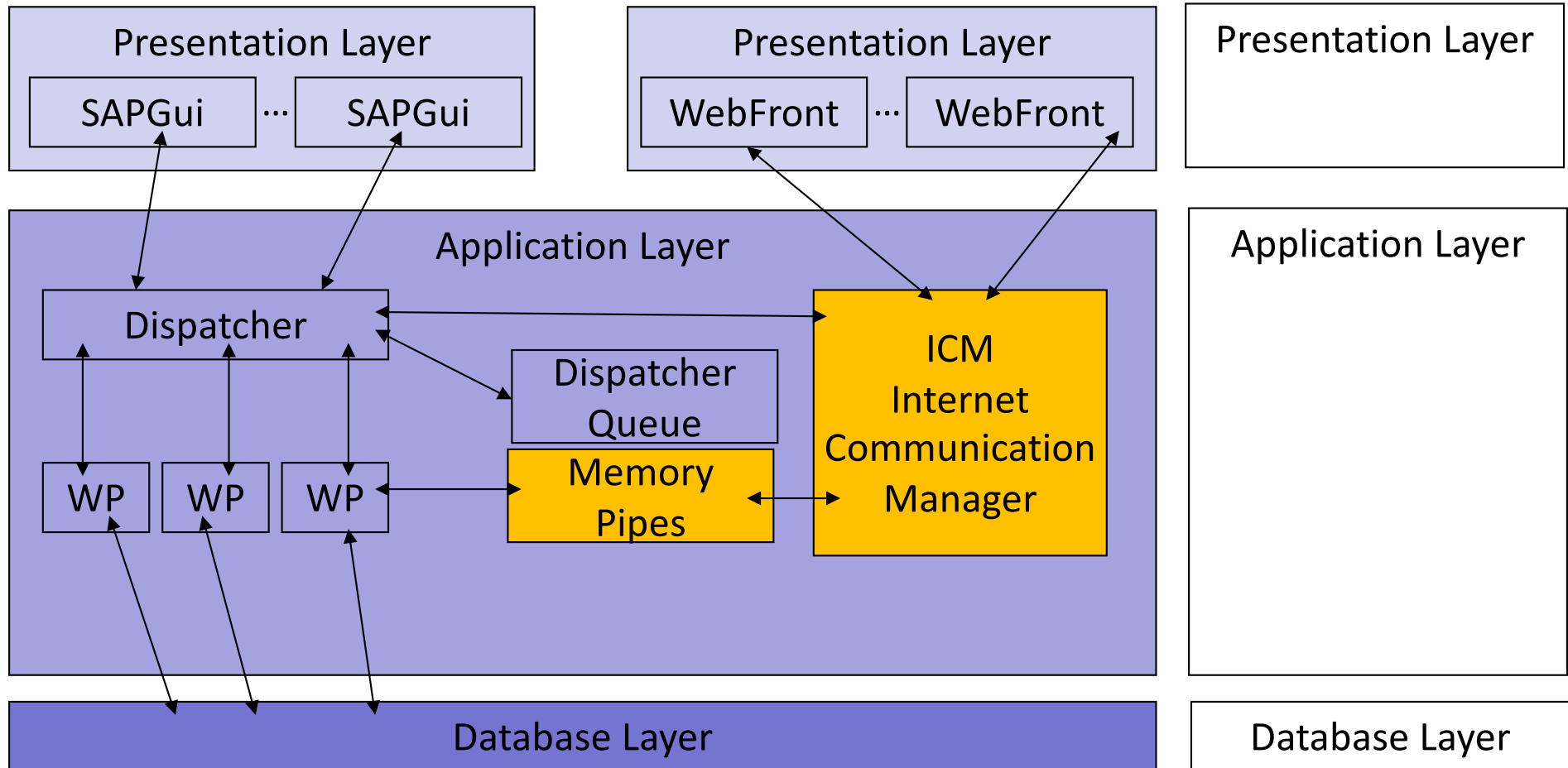


Development Model Evolution



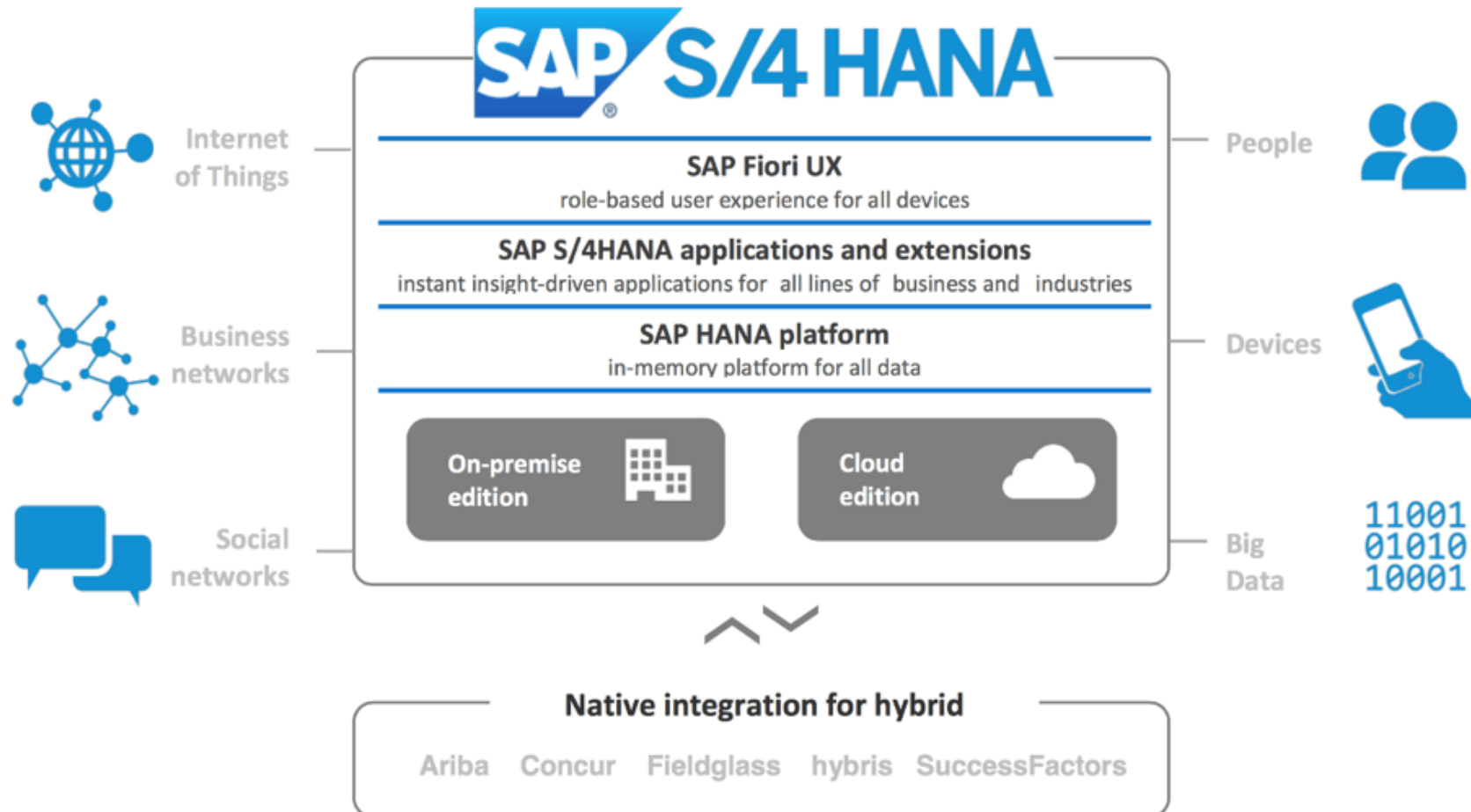
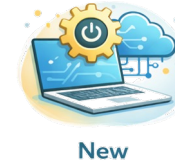
Terms

Application Platform ABAP onPremises



Terms

SAP System Examples



© SAP

Terms

SAP integration options



OnPremises

Classic

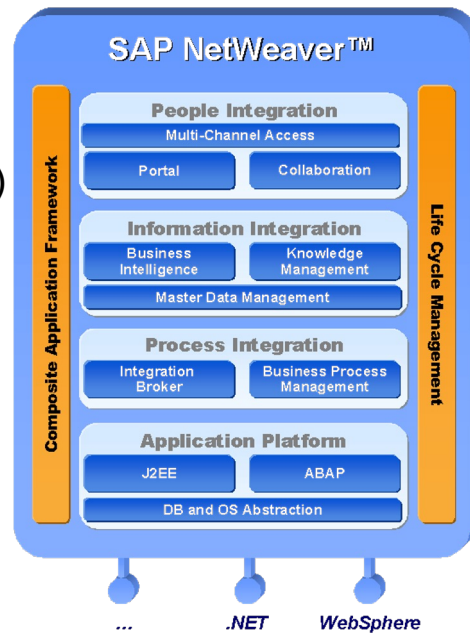
- Java Connector (Jco)
- Flash Islands (Flex)
- Silverlight Islands (MS)

Current

- Web Services
- OData Rest Services
- ...

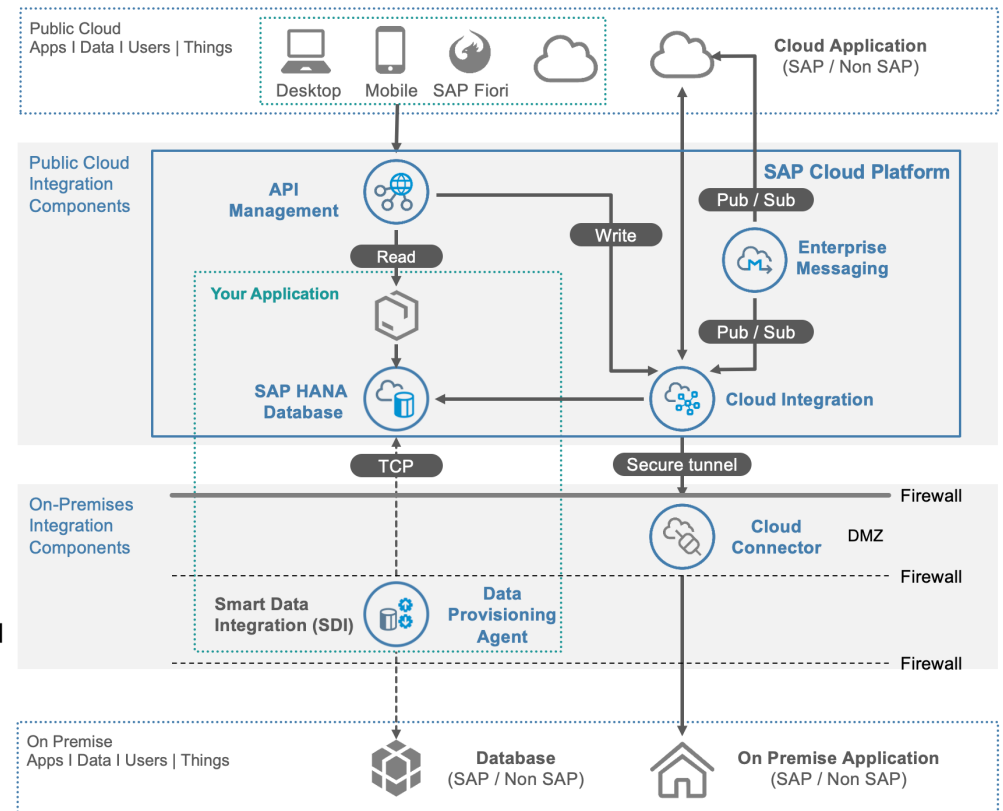
Further

- Operation mode server or client



Quelle: [7]

Cloud

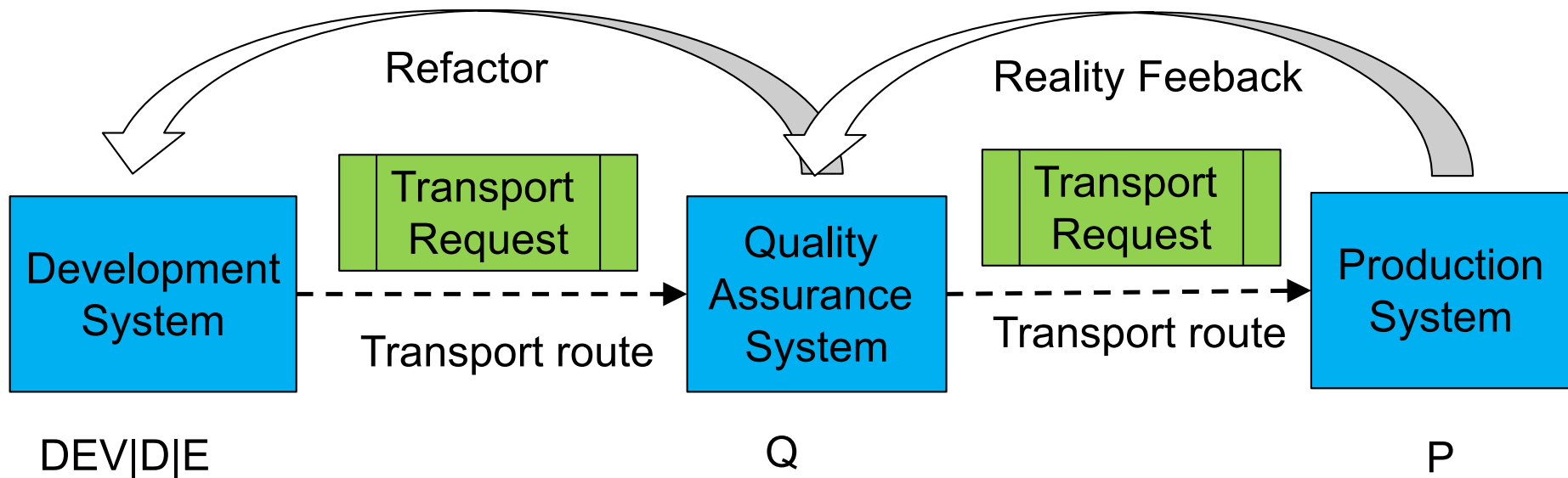


© SAP

Terms

From Development to Production

Classics

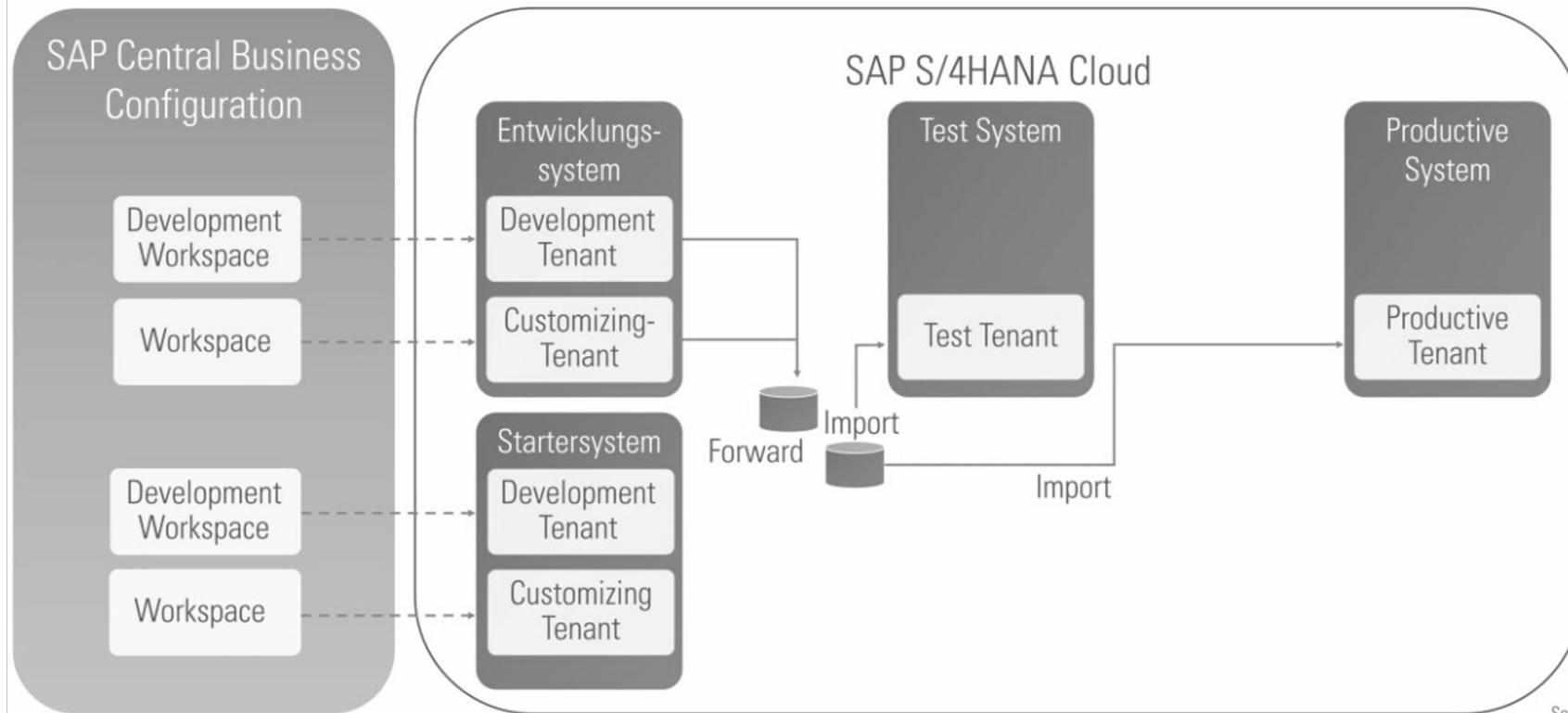


Terms

From Development to Production



S/4HANA Public Cloud 3-System-Landscape

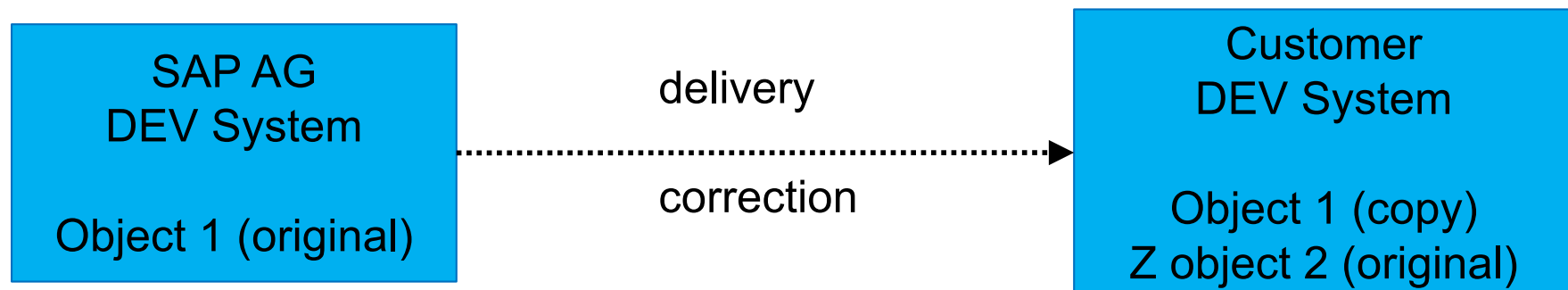


Source: SAP SE

Terms

SAP upgrade

- Original (from SAP or customer)
- Copy (Is the original in the subsequent system)
- Modification (change to the copy of SAP)
- Modification Adjustment (comparison of modifications with the new SAP deliveries)



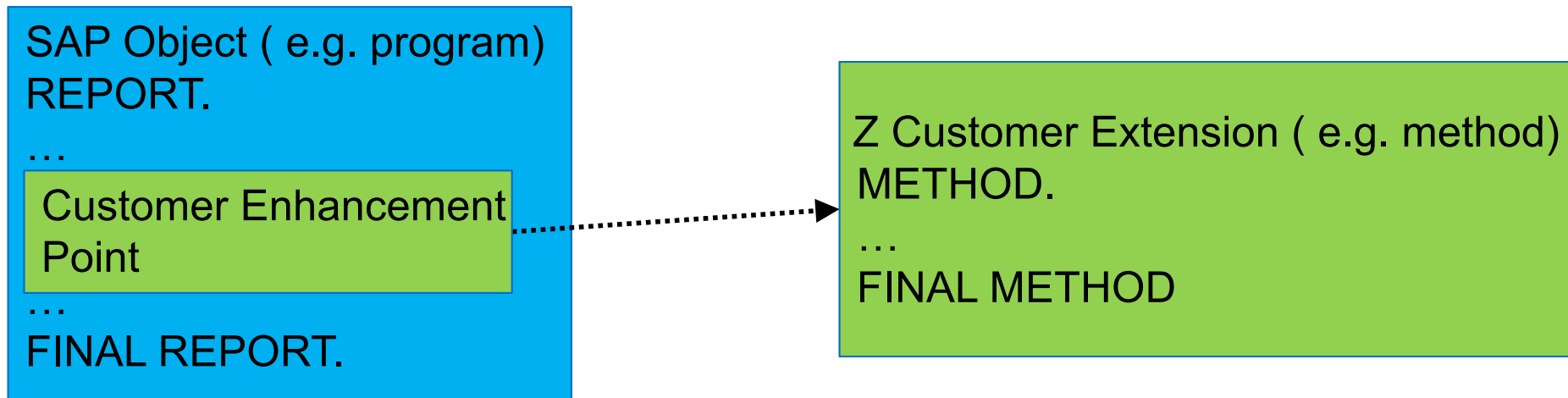
Terms

Extensibility



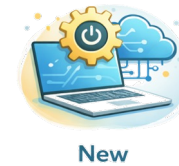
SAP Extensibility

- Principle (SAP prepares an extension possibility for customer implementation = customer development object in the customer namespace, like Z*)
- Enhancement Framework



Terms

Extensibility



SAP S/4HANA Cloud Public Edition offers support for the following three main extensibility options:

- **Key User Extensibility**
 - Use key user extensibility for quick, simple extensions.
 - Ensure key users are familiar with the business processes to maximize efficiency.

- **Developer Extensibility**
 - Utilizing the SAP S/4HANA Cloud ABAP Environment and standard ADT tools, it allows for the development of cloud-ready, upgrade-stable custom ABAP code.

- **Side-by-Side Extensibility**
 - Side-by-side extensibility lets you build custom applications or extend existing ones, using the SAP Business Technology Platform (BTP).
 - On SAP BTP, you can develop using your preferred runtime or language, such as ABAP, Java, or Node.js.
 - Use side-by-side extensibility for loosely coupled scenarios.
 - Ensure seamless integration using SAP API Business Hub.

▪

Terms

Standard vs. Custom



SAP standard vs. customer development

- SAP delivers **standard functionality**
- Customers implement their own processes to realize a competitive advantage, i.e. **customer development** (custom development)

Terms

SAP GUI/Logon

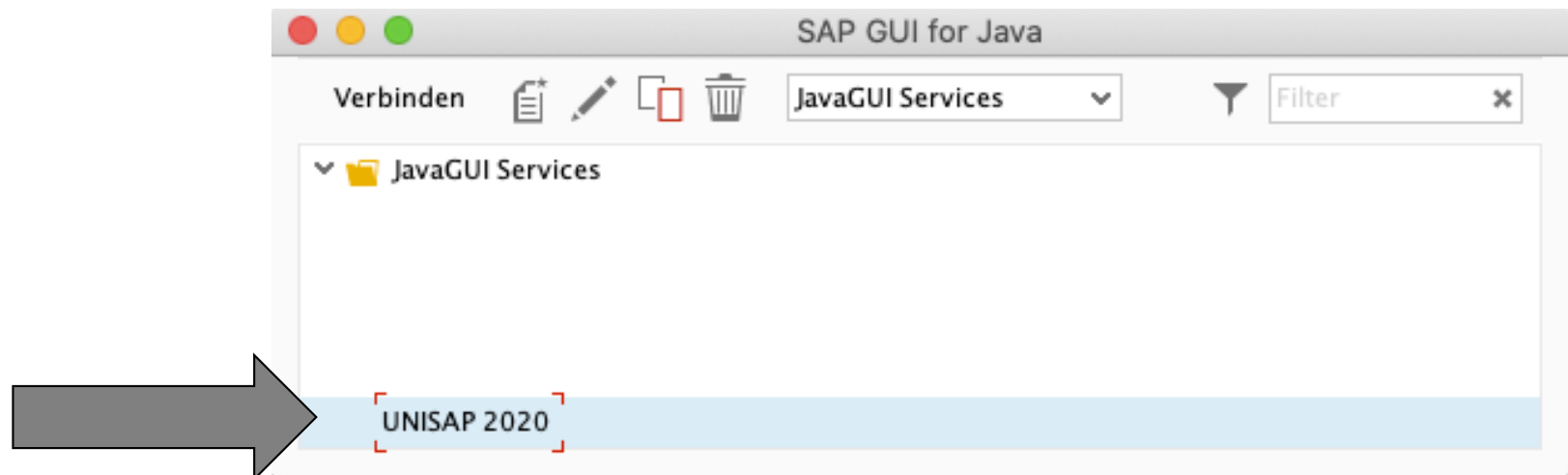


GUI for User & Developer

- Open SAP Logon
- Double Click your System to open SAP GUI



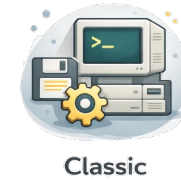
SAPLogon



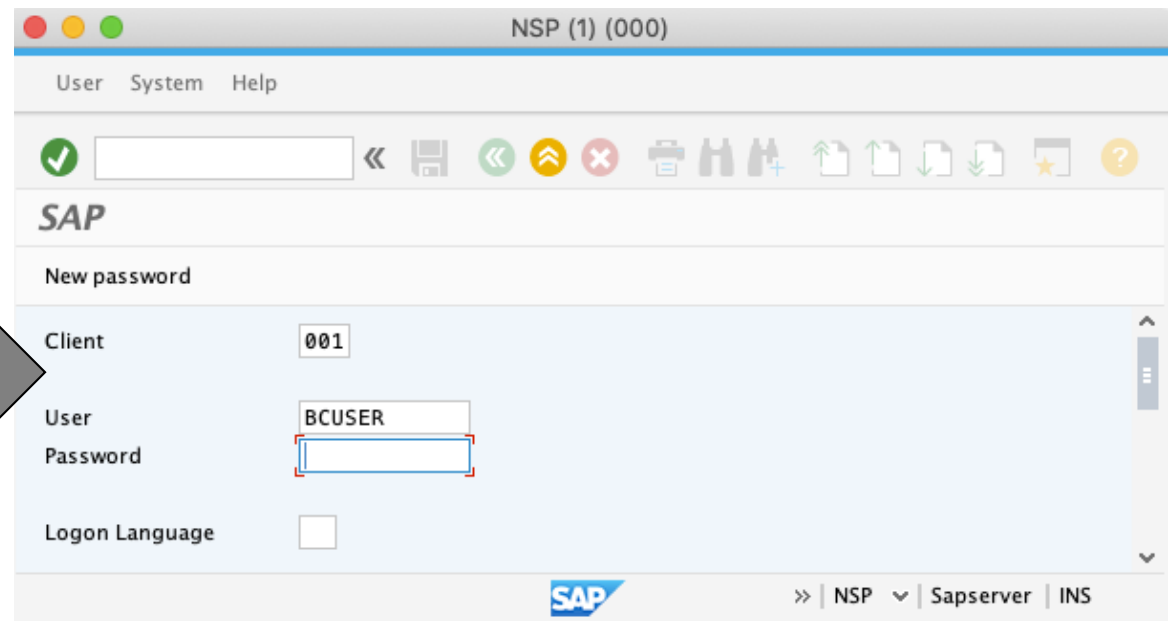
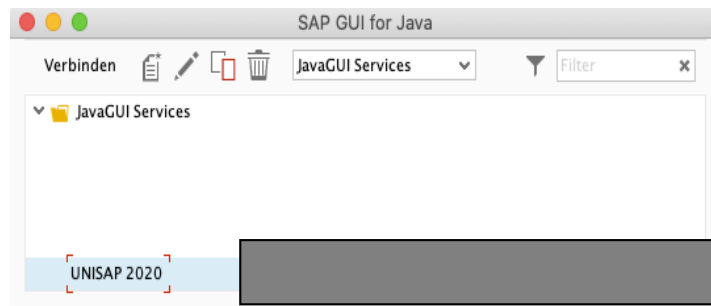
<https://developers.sap.com/trials-downloads.html>

Terms

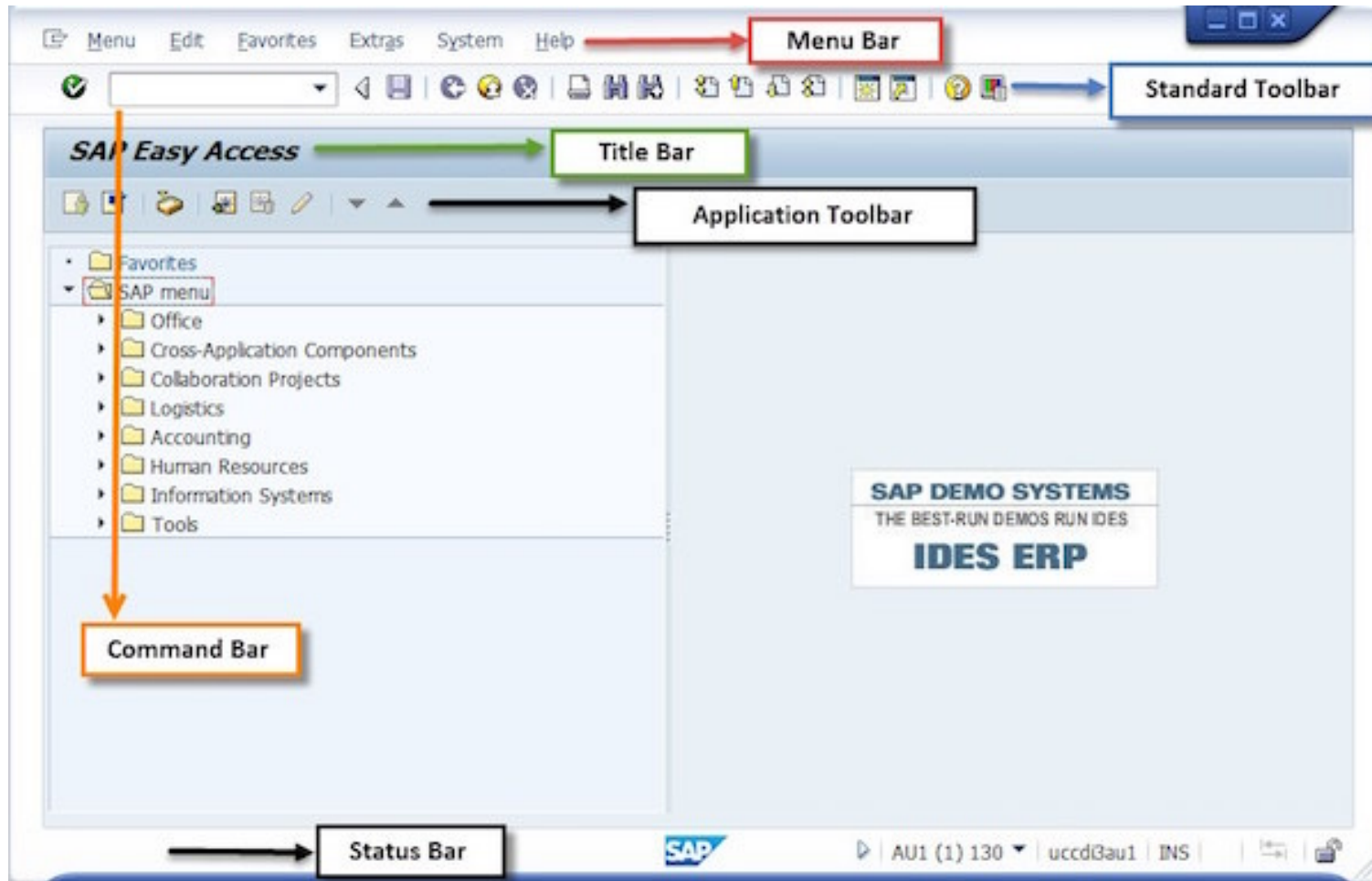
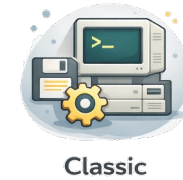
SAP GUI Logon



SAP Logon via SAP GUI



Terms SAP GUI



© SAP

Terms

Transaction Code



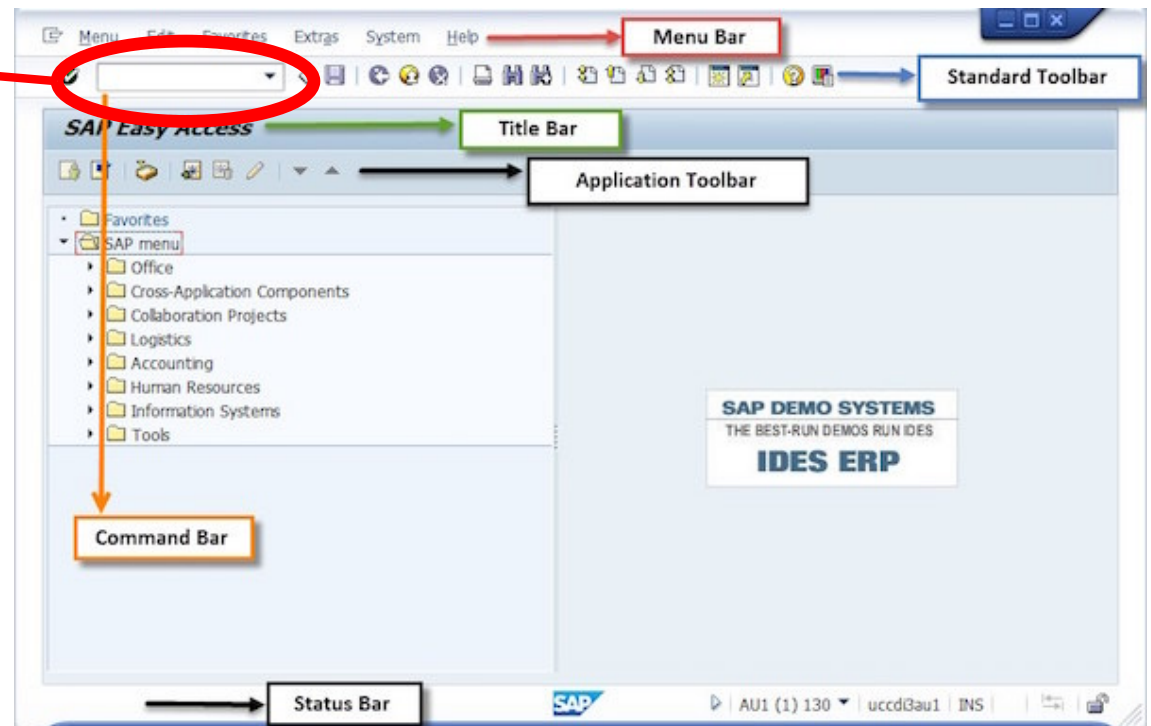
- A transaction code in SAP is a shortcut to functionality, like SE80.
- It has nothing to do with a classical transaction on DB level
- You insert the transaction code in the Command Bar.

Terms SE80

THE most important transaction code :

SE80

Object Navigator



Terms

Prefix transaction code



Prefix for the transaction code :

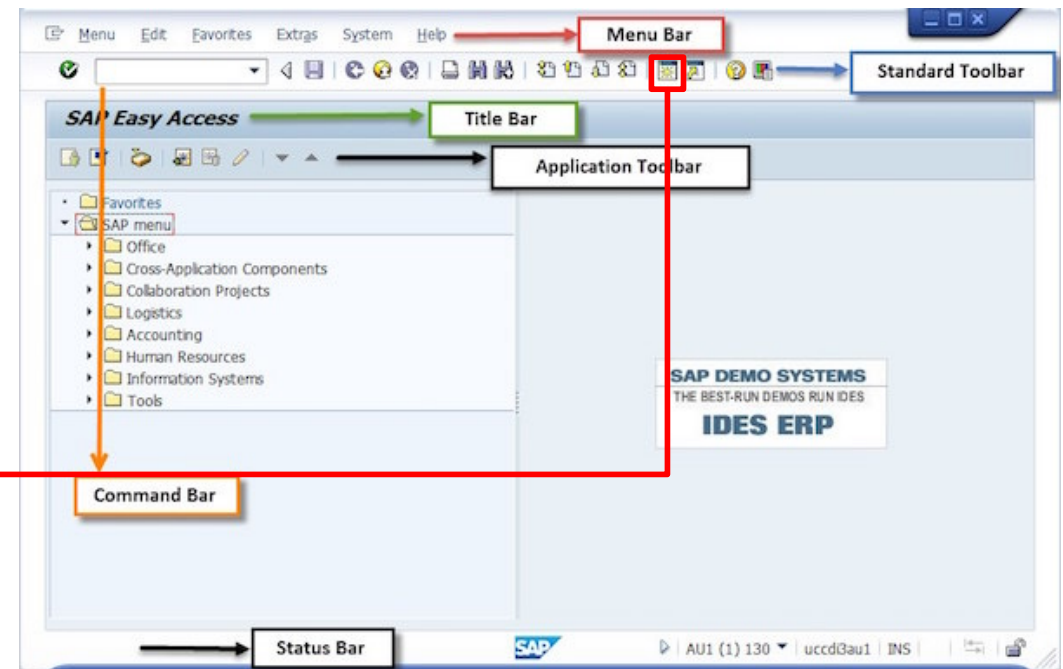
Prefix + transaction,
e.g. /nSE80 = Object Navigator

Prefix:

- /n = clear actual window and start transaction
- /o = new window

Direct Input:

- /nex = close all windows



Terms

Eclipse ADT



- Additional development environment: Eclipse
- Perspective: ABAP Development Tools (ADT)

<https://tools.hana.ondemand.com/> _ _

Type	Test Case ID	Finding Type	Impact	Location	Package	System ID	Mitigation
Flaw	#0029	Disclosure of Critical Data (Customized)	High	ZVF_USERDEMO	ZVF_DEMO	R66	n/a
Flaw	#0080	Missing AUTHORITY-CHECK in Reports	High	ZVF_USERDEMO	ZVF_DEMO	R66	Add an autho...
Warning	#0201	Missing sy-subrc Check after SELECT ... INTO / APPENDII	Low	ZVF_USERDEMO	ZVF_DEMO	R66	Add a sy-subr...
Warning	#0062	Usage of SELECT *	Medium	ZVF_USERDEMO	ZVF_DEMO	R66	n/a
Warning	#0064	Missing WHERE Restriction in SELECT Statement	High	ZVF_USERDEMO	ZVF_DEMO	R66	n/a
Warning	#0089	Usage of LOOP AT itab INTO	Medium	ZVF_USERDEMO	ZVF_DEMO	R66	n/a

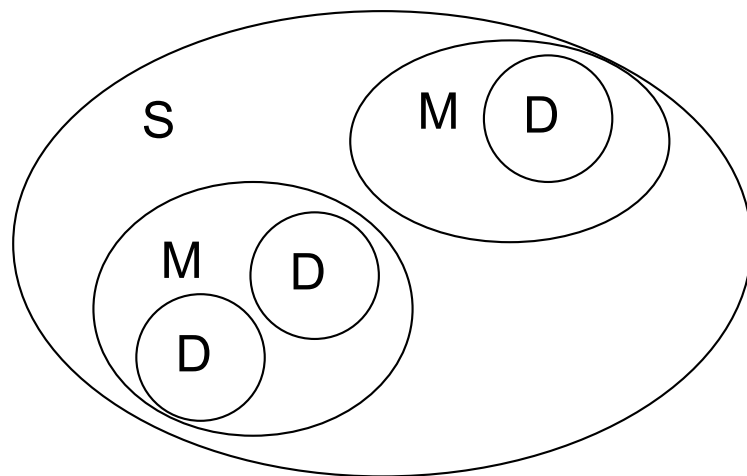
Terms

Package

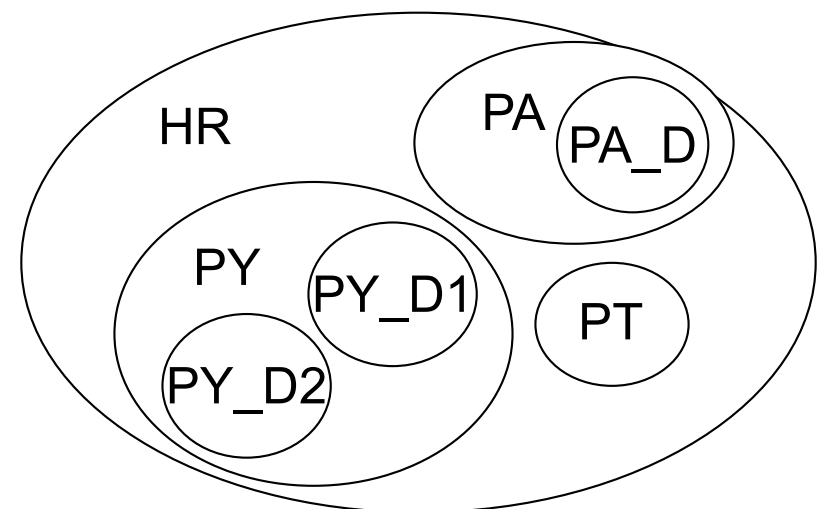
Development organization packages (~Java

Package)

- Structural package (S)
- Main package (M)
- Development package (D)



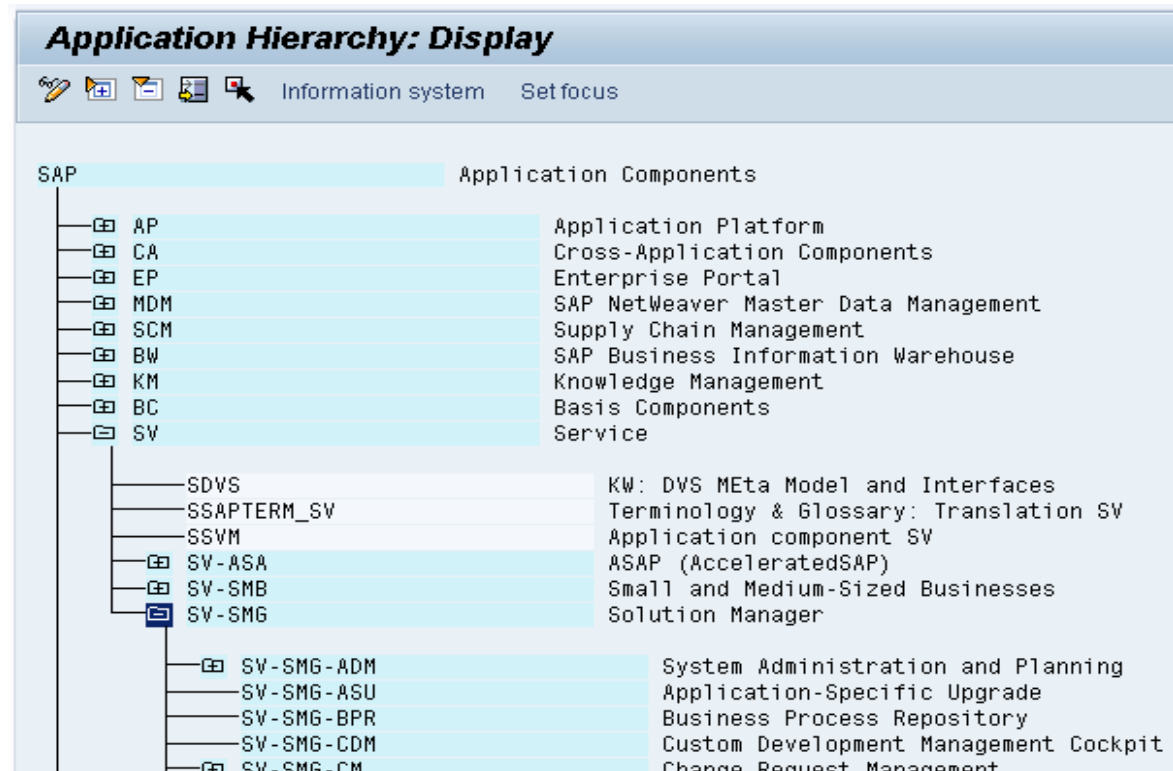
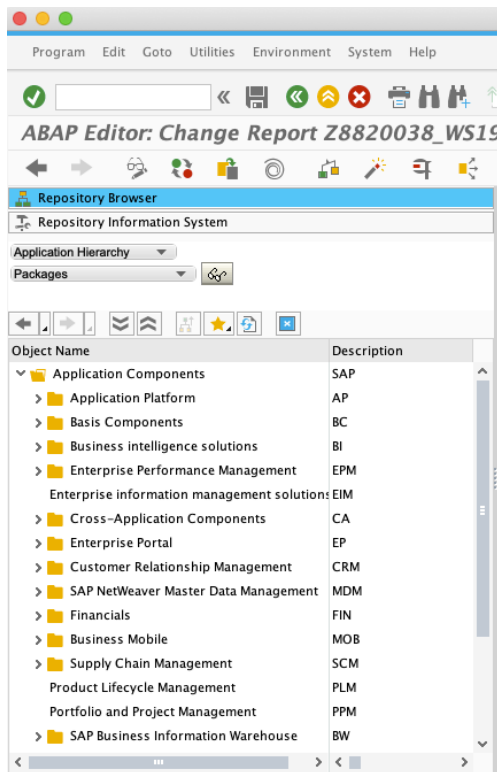
Example HR



Terms

Enterprise business area

Development organization application component ~
Enterprise Business Area



Terms

Software component



Development organization software component

This describes a set of **development objects** that can only be **delivered together** .

Exception : Packages that are not to be delivered to customers must, however, be assigned to the

Terms

Transport layer



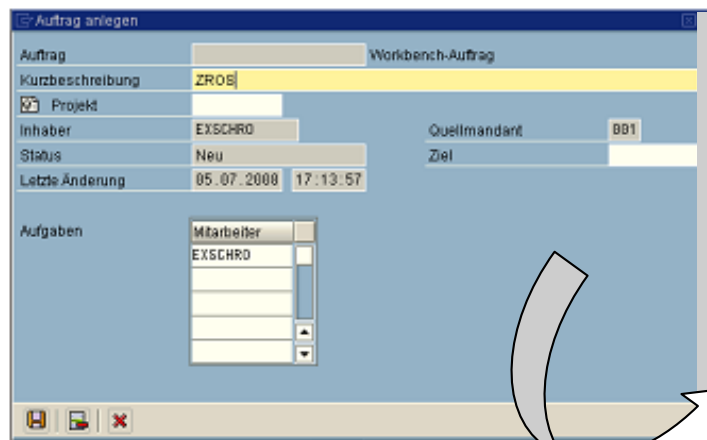
Development organization transport layer

All development projects that are carried out in an SAP system and transported on the same transport routes are combined into a **transport layer** .

Terms

Transport request

Development organization **transport request**



Projektleiter:

CARSON

Team:

CARSON

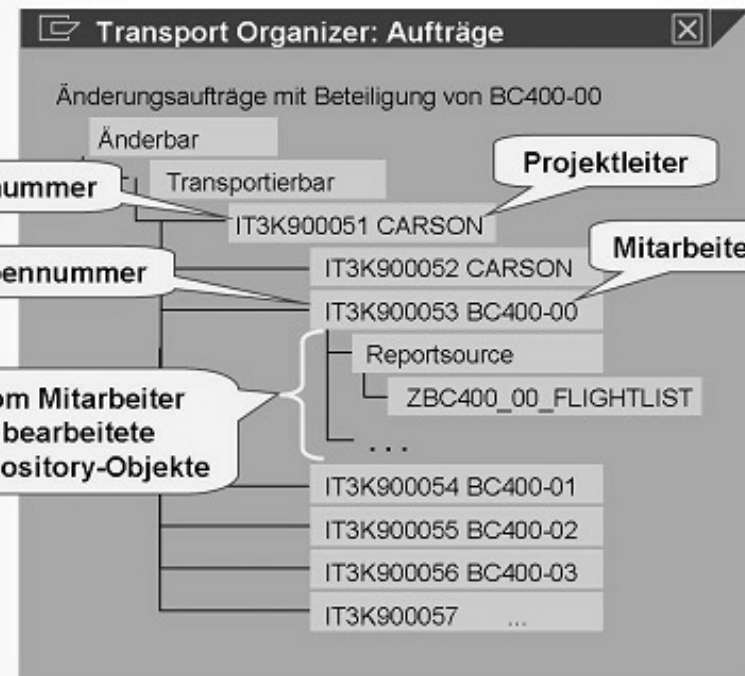
BC400-00

BC400-01

BC400-02

BC400-03

...



Transaction SE09

Terms Packages



Development organization : Based on package,
e.g. Local Package \$TMP

Package Saved

Properties Subpackages Package Hierarchy

Basic Data

Short Description	Temporary Objects (never transported!)		
Person Responsible	SAP		
Created by	SAP	Created On	<input type="text"/>
Last Changed By	SAP	Changed on	<input type="text"/>
Application Component	<input type="text"/>		

Package Properties

Superpackage

Main Package
 Adding further objects not possible

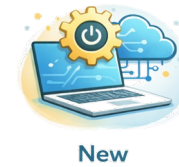
Transport Attributes

Transport Layer

Software Component

Record Object Changes in Transport Requests

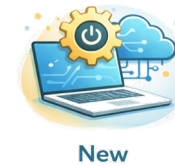
Terms Packages



Development organization : Based on package,
e.g. Local Package ZLOCAL

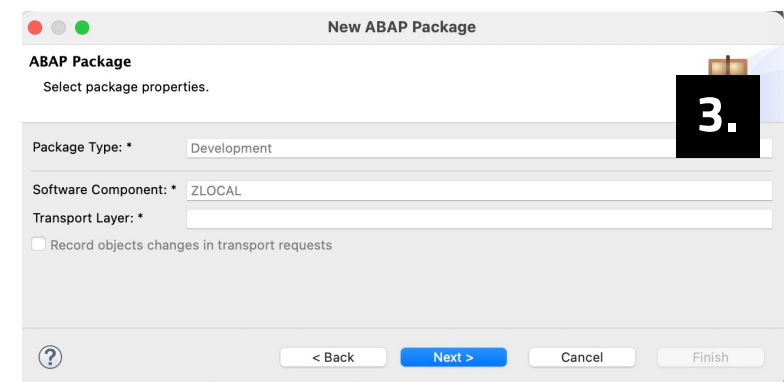
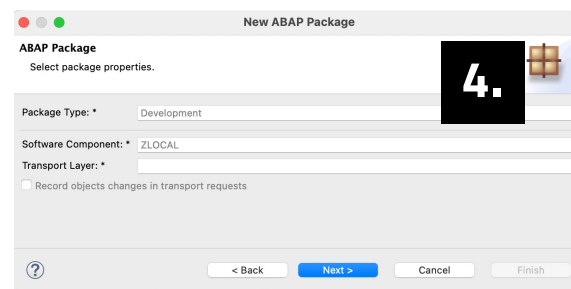
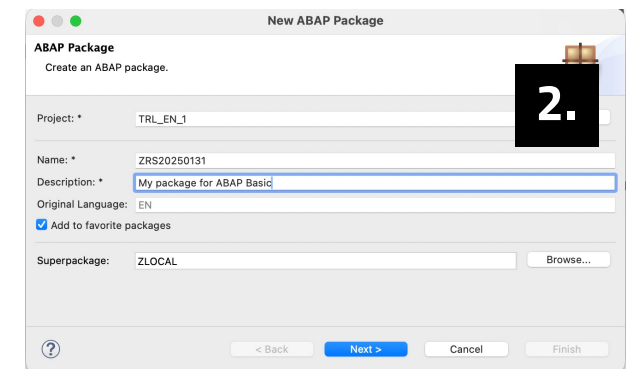
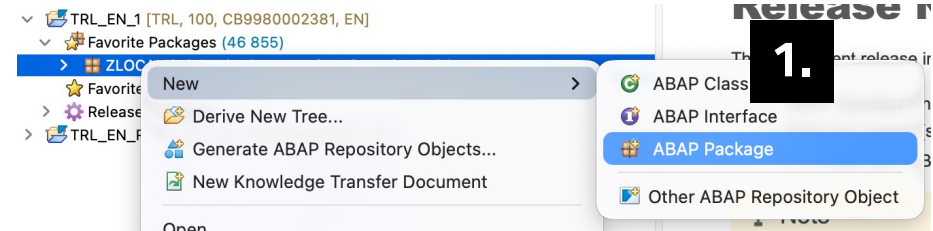


Terms Do it yourself!



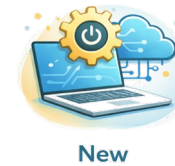
Create a development package below ZLOCAL:
Name = Z<MATNR>





1. Use the context menu in ADT on ZLOCAL
2. Fill out the creation form
3. Save



Terms


Do it yourself!




- ▼  TRL_EN_1 [TRL, 100, CB9980002381, EN]
- ▼  Favorite Packages (46 854)
 - >  ZLOCAL (46 855) *Generated Package for ZLOCAL*
 - >  ZRS20250131 (0) *My package for ABAP Basic*

Package: ZRS20250131

General Data

Responsible: (3e84efd6-b7c2-4889-aed2-96fd78ee704c)
Default ABAP Language Version: 

Package Properties

Superpackage: 
Package Type:
 Adding further objects not possible
 Package encapsulated

Transport Properties

Transport Layer:
Software Component:
 Record objects changes in transport requests

Terms

Namespaces



SAP delivers its own software developments to the customer. The customer can develop their own software. In principle, there can be **name overlaps** between **SAP objects** and **customer objects** .

There are **two variants** (namespaces) to protect customer objects.

1. If development objects **begin with Z or Y** , they are implicitly protected because SAP does not deliver objects beginning with Z or Y (exception: customer exits)
2. **Reserved namespace** from SAP . This has 10 digits and begins with "/" and ends with "/". This namespace must be prefixed to the development objects. This variant is particularly interesting for third-party providers.

Terms

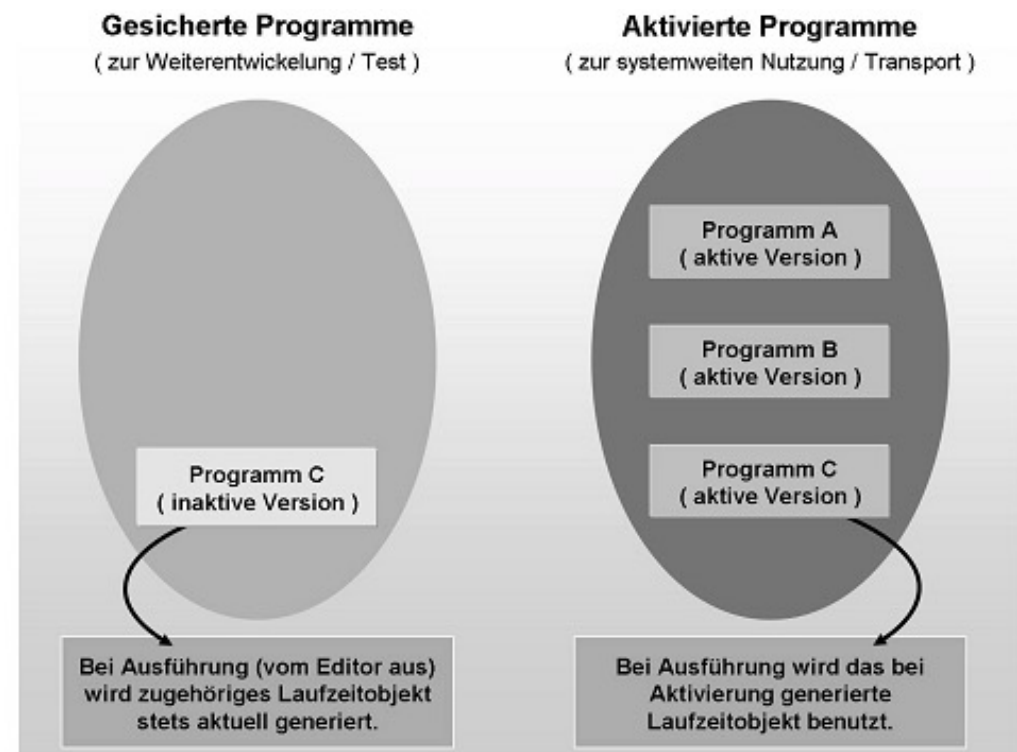
Activation State

- **Repository objects**

- Activity States
- Program Types
- Function Modules
- ...

- **Dictionary objects**

- Activity States
- Domain
- Data Elements
- Structure
- Transparent Table

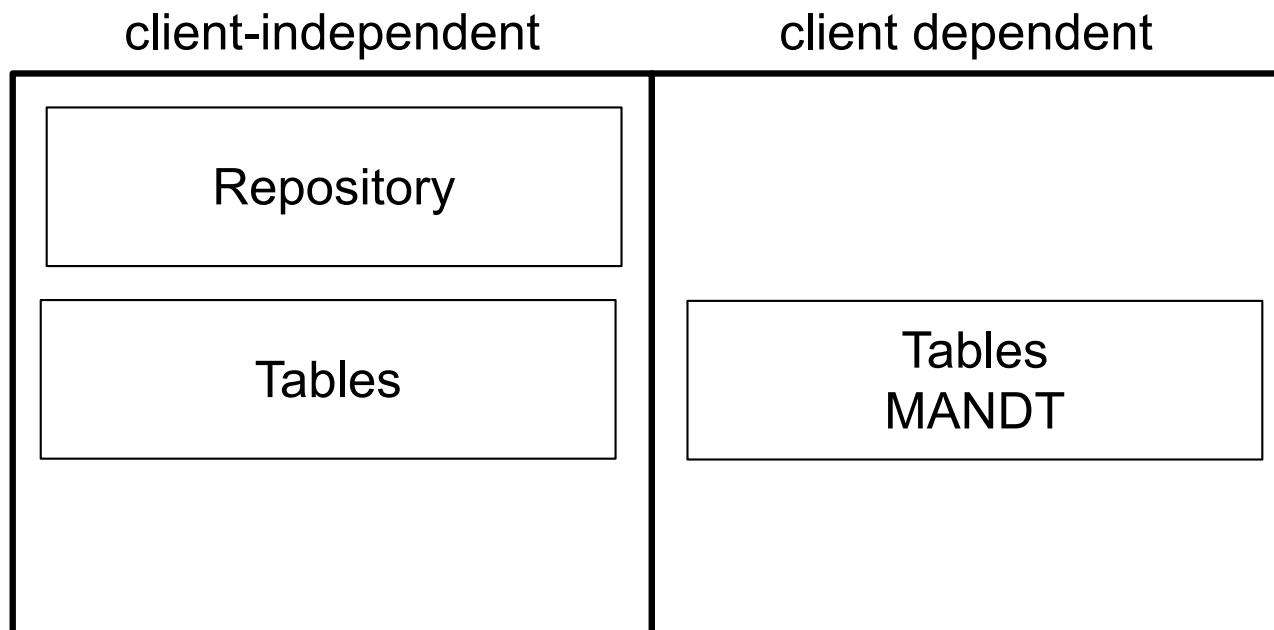


Terms

Repository Object

- **Repository Objects** (repository structure)

Database

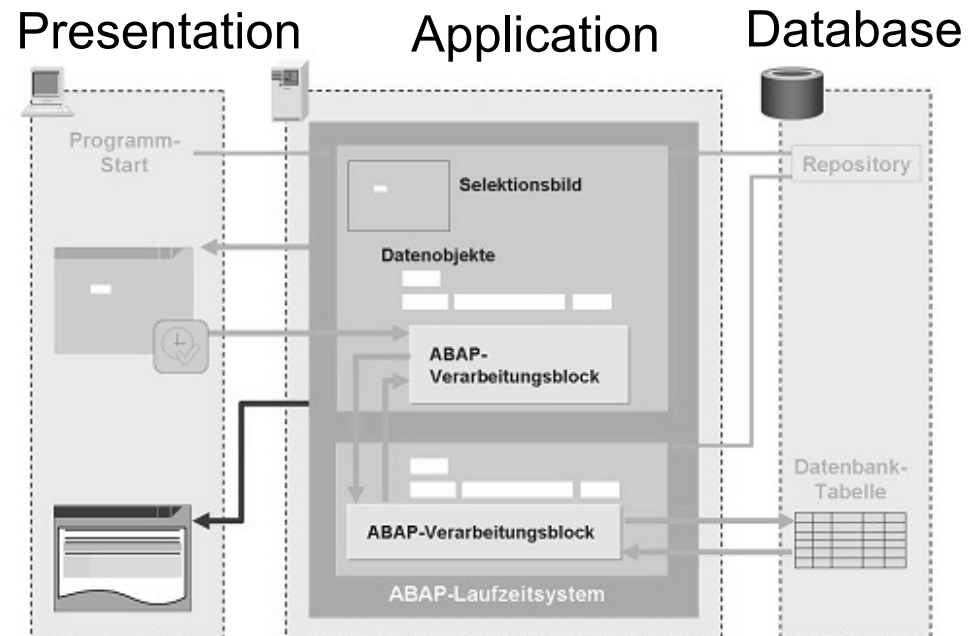
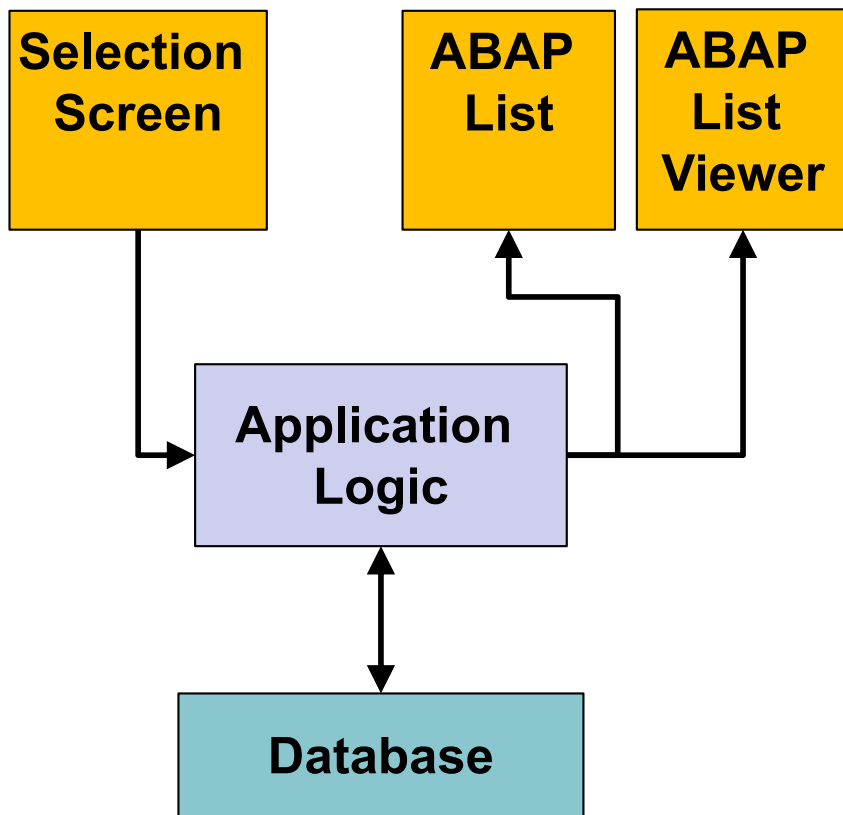


Terms

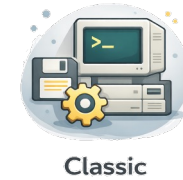
Processing Sequence



- Example program with exemplary sequence



Example



```
*&-----*  
*& report Z8820038_WS19_HELLO_WORLD  
*&  
*&-----*  
*&  
*&  
*&-----*
```

REPORT z8820038_ws19_hello_world.

* Airline input field
PARAMETERS: pa_car TYPE scarr-carrid .

* Target data object for FG name
DATA gd_carrname TYPE scarr-carrname .

INITIALIZATION.
* Initialization for input field
pa_car = 'LH'.

LOAD OF PROGRAM.

START OF SELECTION.

* Reading the FG from the DB table SCARR
SELECT SINGLE carrname FROM scarr INTO gd_carrname
WHERE carrid = pa_car .

END OF SELECTION.

* Output to standard list
WRITE ' Hello World!'.
* New line
new line .

* Output of the name of the FG
WRITE: / 'FG name: ', gd_carrname .

Terms: Sample program

MVC pattern



REPORT z8820038 ws19 hello world.

* Airline input field
PARAMETERS: pa_car TYPE scarr-carrid .

Selection screen = View

* Target data object for FG name
DATA gd_carrname TYPE scarr-carrname .

Business data = model

INITIALIZATION.

* Initialization for input field
pa_car = 'LH'.

LOAD OF PROGRAM.

START OF SELECTION.

* Reading the FG from the DB table SCARR
SELECT SINGLE carrname FROM scarr INTO gd_carrname
WHERE carrid = pa_car .

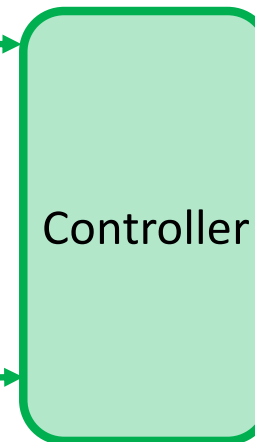
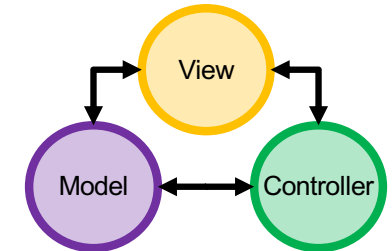
Business logic = model

END OF SELECTION.

* Output to standard list
WRITE ' Hello World!'.
* New line
new line .

Results list = View

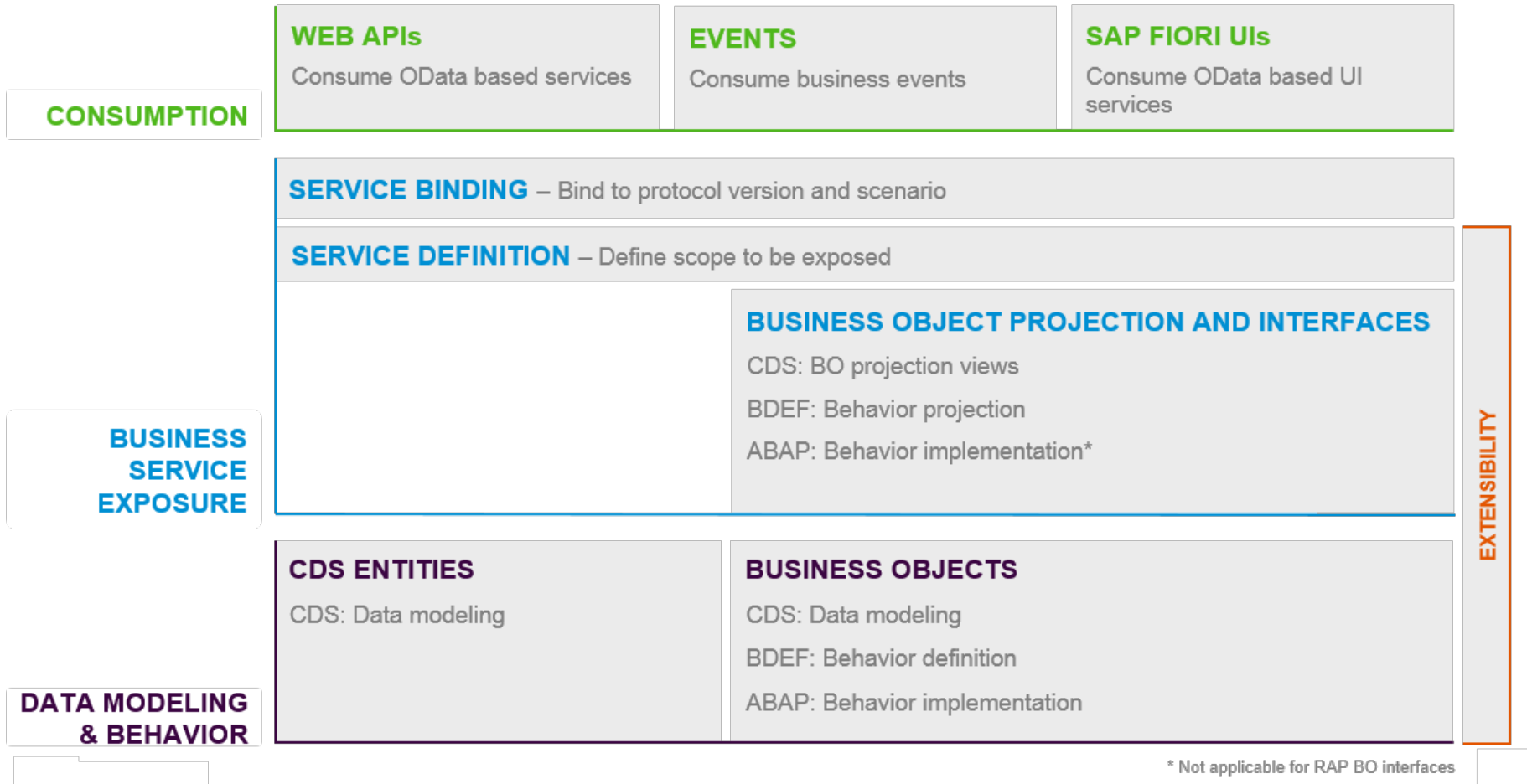
* Output of the name of the FG
WRITE: / 'FG name: ', gd_carrname .



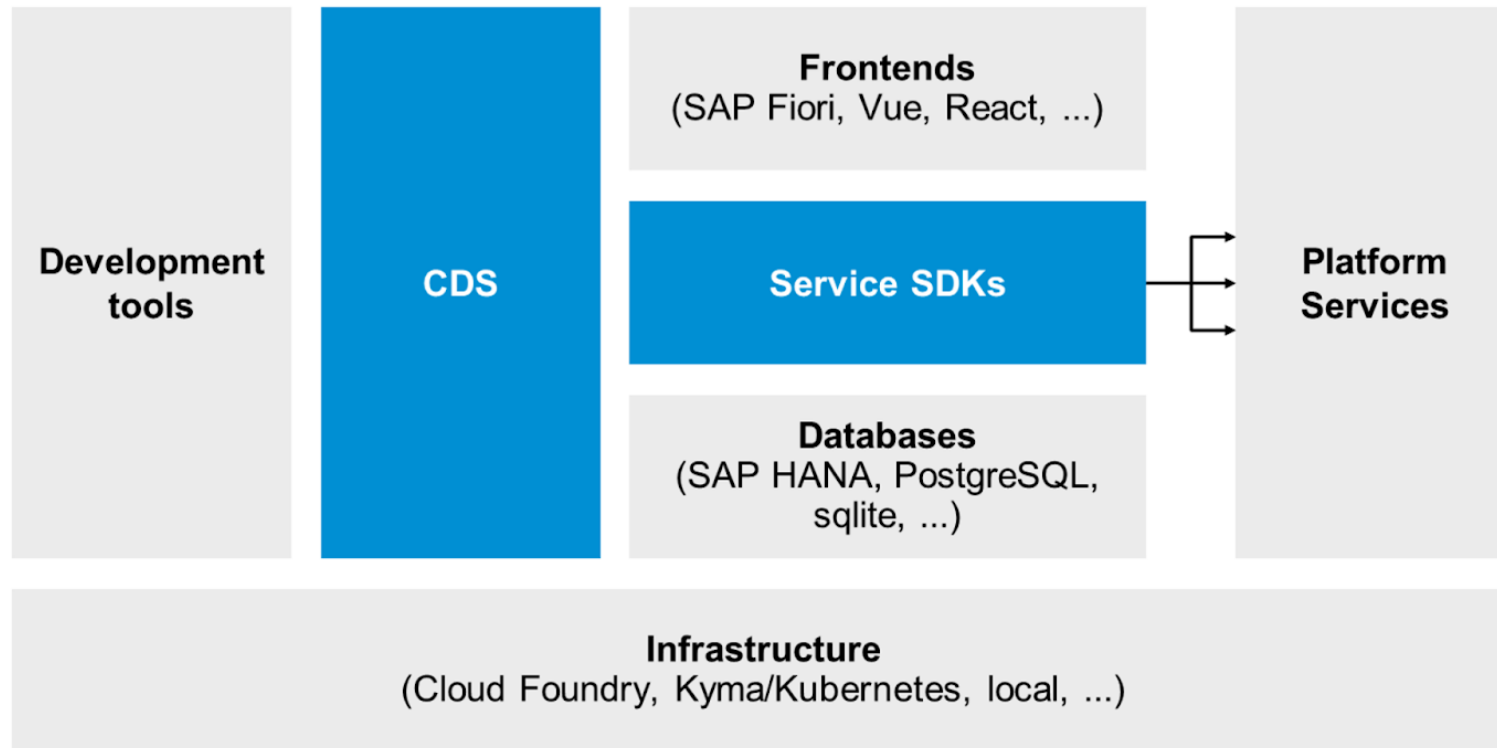
Development Model RAP



RAP - The big picture



Development Model CAP

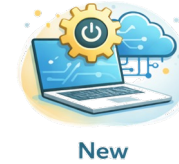


express



(c) SAP

Development Model CAP Example



Employee Management System | Welcome, R S (EMPLOYEE) | Logout

Dashboard | Employees | Departments | Leave Management | Time Sheets

Dashboard

TOTAL EMPLOYEES
8

ACTIVE EMPLOYEES
8

DEPARTMENTS
5

AVERAGE SALARY
\$70 000

Department Statistics

DEPARTMENT	EMPLOYEE COUNT	AVERAGE SALARY
Engineering	3	\$83 333,333
Human Resources	2	\$72 500
Finance	1	\$65 000
Marketing	1	\$45 000
Sales	1	\$55 000

Recent Hires (Last 6 months)

Employee Management System | Welcome, R S (EMPLOYEE) | Logout

Dashboard | Employees | Departments | Leave Management | Time Sheets

Employees

Search employees... | Add New Employee

NAME	EMAIL	DEPARTMENT	POSITION	HIRE DATE	SALARY	STATUS	ACTIONS
David Brown	david.brown@company.com	Human Resources	HR Manager	1.8.2022	\$85 000	ACTIVE	Edit Delete
Emily Davis	emily.davis@company.com	Finance	Financial Analyst	1.4.2023	\$65 000	ACTIVE	Edit Delete
John Doe	john.doe@company.com	Engineering	Senior Software Engineer	15.1.2023	\$75 000	ACTIVE	Edit Delete
Mike Johnson	mike.johnson@company.com	Engineering	Engineering Manager	1.6.2022	\$95 000	ACTIVE	Edit Delete
Alex Miller	alex.miller@company.com	Marketing	Marketing Coordinator	15.5.2023	\$45 000	ACTIVE	Edit Delete
Jane Smith	jane.smith@company.com	Engineering	Software Engineer	1.2.2023	\$80 000	ACTIVE	Edit Delete
Sarah Williams	sarah.williams@company.com	Human Resources	HR Specialist	15.3.2023	\$60 000	ACTIVE	Edit Delete
Lisa Wilson	lisa.wilson@company.com	Sales	Sales Representative	1.11.2022	\$55 000	ACTIVE	Edit Delete

Employee Management System | Welcome, R S (EMPLOYEE) | Logout

Dashboard | Employees | Departments | Leave Management | Time Sheets

Leave Management

[Request Leave](#)

EMPLOYEE	LEAVE TYPE	START DATE	END DATE	DAYS	REASON	STATUS	ACTIONS
John Doe	Annual	25.12.2024	31.12.2024	7	Christmas holidays	PENDING	Approve Reject
Jane Smith	Sick	15.11.2024	16.11.2024	2	Medical appointment	APPROVED	Approved by Manager

Employee Management System | Welcome, R S (EMPLOYEE) | Logout

Dashboard | Employees | Departments | Leave Management | Time Sheets

Time Sheet Management

[Add Time Entry](#)

EMPLOYEE	DATE	HOURS	PROJECT	DESCRIPTION	STATUS	ACTIONS
John Doe	16.12.2024	8h	Project Alpha	Development work on main features	SUBMITTED	Approve Reject
Jane Smith	15.12.2024	7.5h	Project Beta	Testing and bug fixes	APPROVED	Approved by Manager
John Doe	14.12.2024	8.5h	Project Alpha	Code review and documentation	DRAFT	Submit Delete

Weekly Summary

John Doe
This Week: 8h
Total Logged: 8h

Jane Smith
This Week: 7.5h
Total Logged: 7.5h

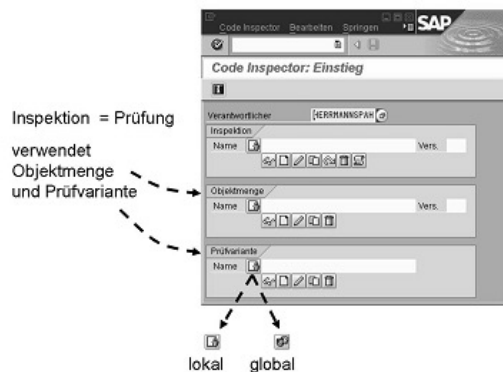
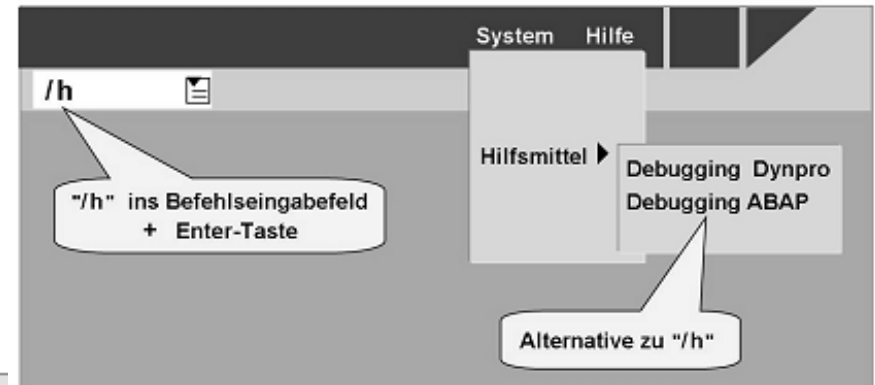
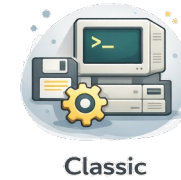
Mike Johnson
This Week: 8h
Total Logged: 8h

Terms

Code analysis

Dynamic vs. Static Analysis

- Debugger (/h)
- Code inspector (TA SCI)
 - Test Variant
 - Object Set
 - Inspection



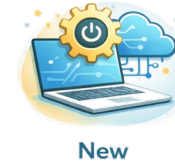
Terms

Code analysis



- **ABAP Test Cockpit (ATC) (Static Analysis)**: Identifies syntax errors, coding standard violations (e.g., naming conventions), and potential security vulnerabilities. It supports remote analysis for system conversions (e.g., S/4HANA readiness).
- **ABAP Profiler (Dynamic/Performance Analysis)**: Integrated into ADT to trace code execution, analyzing runtime, database accesses, and performance bottlenecks. It provides views like call trees and timelines to pinpoint slow code.
- **ABAP Debugger**: Allows interactive analysis of code at runtime, including breakpoints, stepping through code, and inspecting variable values, including specialized views for internal tables.
- **Code Inspector (SCI)**: Traditional static code analysis tool integrated into ADT for checking compliance with development standards.

Analysis Tools



Overview

General Information

Title: ZRAM_TEST_PROFILING

ID: `supersocd_psd_fm_AT010013.DAT`

Date: 15.06.2021, 00:10:55

Server: `supersocd`

User: `supersocd`

Aggregation: No Aggregation ([Call Sequence available](#))

Size: 40 KB

Analysis Tools

- Condensed Hit List Shows top consumers aggregated to called units (methods, fu
- Hit List Shows top consumers by call position
- Aggregated Call Tree Shows aggregated call stacks in a hierarchy
- Call Sequence Shows Sequence of trace events in a hierarchy
- Call Timeline Shows diagram of trace events and time consumed
- Database Accesses Shows database accesses
- SQL Trace Shows SQL trace in SAPGUI

Runtime Distribution

Overall: 0.02s

ΔBAP: 0.01s (37%)

Database: 0.01s (57%)

System: < 0.01s (6%)

Navigation: Overview | Condensed Hit List | Hit List | Aggregated Call Tree | Call Sequence | Call Timeline | Database Accesses

Stack →

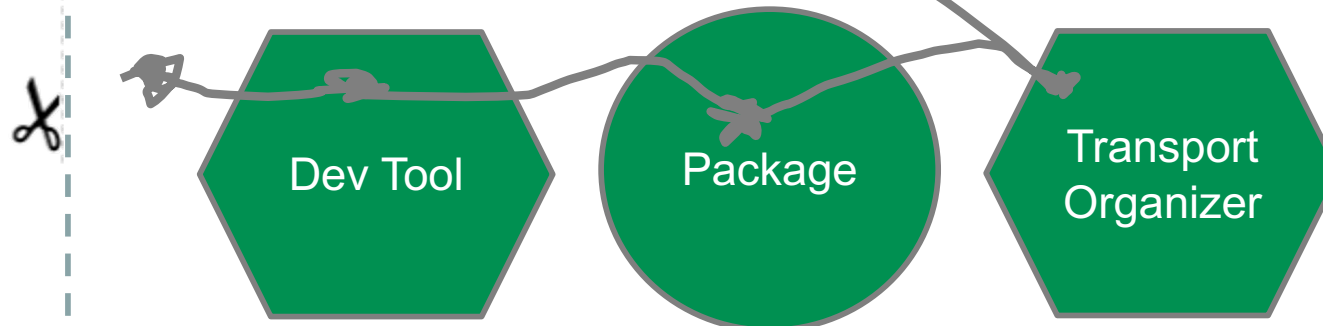
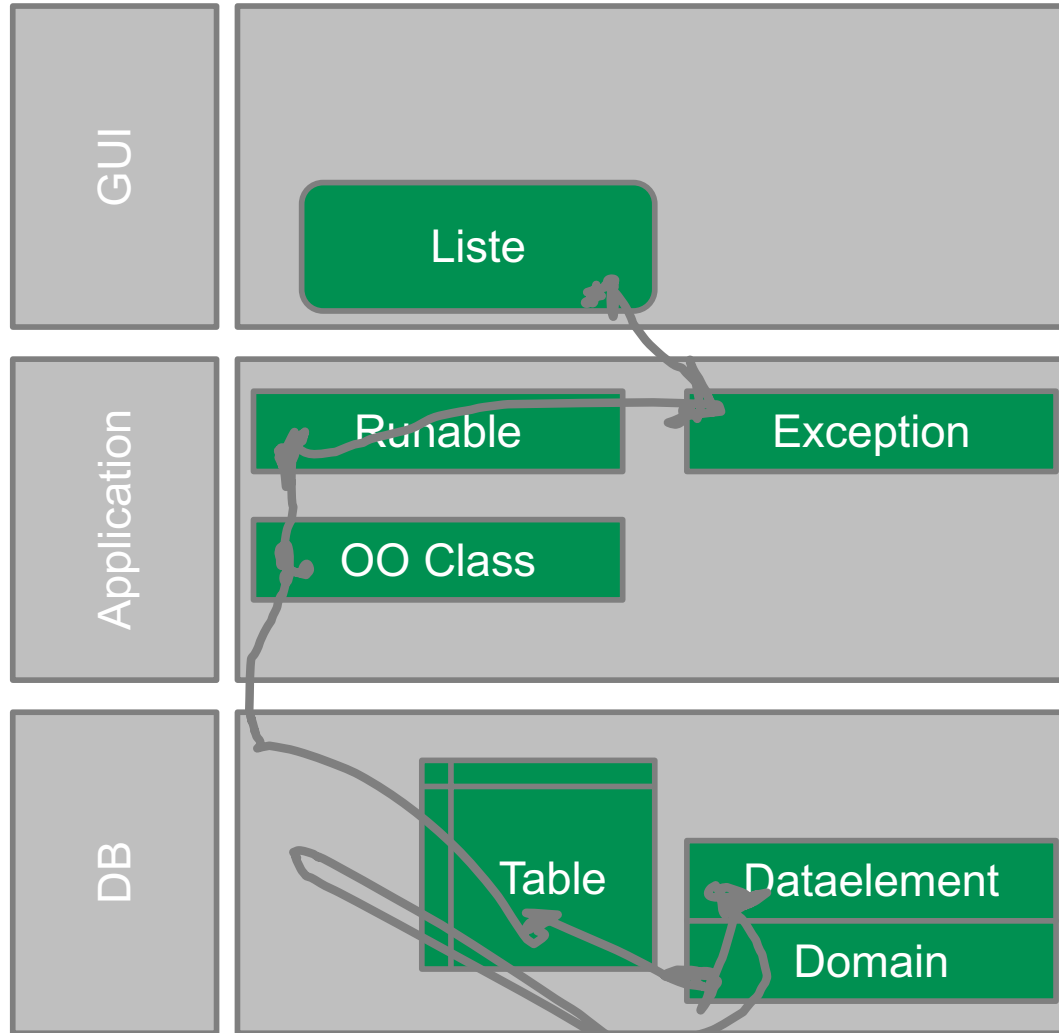
Debugger →

Variable →

Outline →

Plan

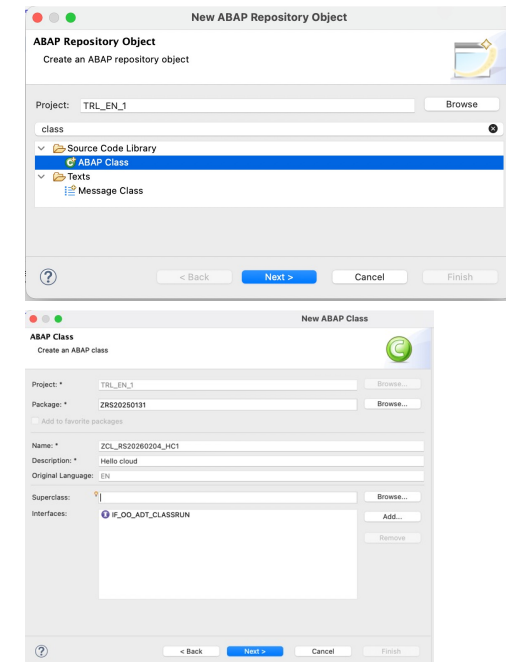
The Plan



Hello Cloud

Create a console application

- Create an ABAP class with interface
IF_OO_ADT_CLASSRUN
 - ZCL_<INITIAL>20260204_HC



Created class

Short introduction



```
CLASS zcl_rs20260204_hc1 DEFINITION PUBLIC FINAL CREATE PUBLIC .
```

```
PUBLIC SECTION.  
INTERFACES if_oo_adt_classrun .
```

```
PROTECTED SECTION.
```

```
PRIVATE SECTION.  
ENDCLASS.
```

```
CLASS zcl_rs20260204_hc1 IMPLEMENTATION.
```

```
METHOD if_oo_adt_classrun~main.  
* Here goes the code  
ENDMETHOD.
```

```
ENDCLASS.
```

Great code!

```
CLASS zcl_rs20260204_hc1 DEFINITION PUBLIC FINAL CREATE PUBLIC .
```

```
PUBLIC SECTION.
```

```
INTERFACES if_oo_adt_classrun .
```

```
PROTECTED SECTION.
```

```
PRIVATE SECTION.
```

```
ENDCLASS.
```

```
CLASS zcl_rs20260204_hc1 IMPLEMENTATION.
```

```
METHOD if_oo_adt_classrun~main.
```

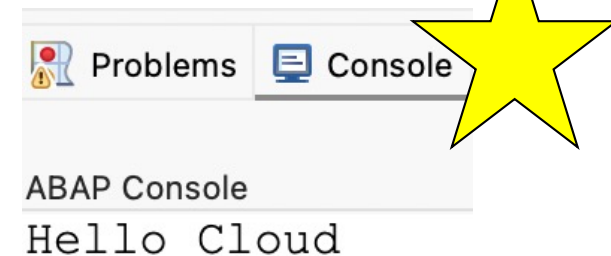
```
* Here goes the code
```

```
out->write( 'Hello Cloud' ).
```

```
ENDMETHOD.
```

```
ENDCLASS.
```

1. Activate and test your class.
 1. Activate the class with the keyboard shortcut **Ctrl + F3**.
 2. Run the class with the **F9** key.
2. Check the output in the *Console* view of Eclipse.
 1. Check the *Console* view that should have opened as a new tab below the editor view.
 2. If the *Console* view is not visible, open it by choosing *Window*→*Show view*→*Other*. Double-click *Console* in the hit list.



Dictionary

Dictionary

Structured Analysis and Structured Design (SA/SD) is a traditional software development methodology that was popular in the 1980s and 1990s. It involves a series of techniques for designing and developing software systems in a structured and systematic way. Here are some key concepts of SA/SD:

- **Functional Decomposition:** SA/SD uses functional decomposition to break down a complex system into smaller, more manageable subsystems. This technique involves identifying the main functions of the system and breaking them down into smaller functions that can be implemented independently.
- **Data Flow Diagrams (DFDs):** SA/SD uses DFDs to model the flow of data through the system. DFDs are graphical representations of the system that show how data moves between the system's various components.
- **Data Dictionary:** A data dictionary is a central repository that contains descriptions of all the data elements used in the system. It provides a clear and consistent definition of data elements, making it easier to understand how the system works.
- **Structured Design:** SA/SD uses structured design techniques to develop the system's architecture and components. It involves identifying the major components of the system, designing the interfaces between them, and specifying the data structures and algorithms that will be used to implement the system.
- **Modular Programming:** SA/SD uses modular programming techniques to break down the system's code into smaller, more manageable modules. This makes it easier to develop, test, and maintain the system.

Data Objects

Data Objects

Variables

- Value may change during runtime
- Identified by a name
- Well-defined starting value

Constants

- Value hard-coded in source code
- Identified by a name

Literals

- Value hard-coded in source code
- Anonymous (no name)

Declaration of variables

Statement DATA

```
DATA <variable_name> TYPE <type> [ VALUE <starting_value> ].
```

Name of the variable

Type of the variable

Starting value of the variable (optional)

Examples

```
DATA my_var1 TYPE i.  
DATA my_var2 TYPE string.  
DATA my_var3 TYPE string VALUE `Hello World`.
```

ABAP Dictionary (DDIC) Data Types

General Data Types

Scope of Types

- Built in types in ABAP
- Custom Types
- ABAP Dictionary Types

Kind of Types

- Elementary Types
- Structured Types
 - Structure
 - View
 - Transparent Table
- Table Types
- Referential Types

Try it out!

* Data Objects with Built-in Types

" comment/uncomment the following declarations and check the output

DATA variable TYPE string.

* DATA variable TYPE i.

* DATA variable TYPE d.

* DATA variable TYPE c LENGTH 10.

* DATA variable TYPE n LENGTH 10.

* DATA variable TYPE p LENGTH 8 DECIMALS 2.

* Output

```
out->write( 'Result with Initial Value' ).
```

```
out->write( variable ).
```

```
out->write( '-----' ).
```

```
variable = '19891109'.
```

```
out->write( 'Result with Value 19891109' ).
```

```
out->write( variable ).
```

```
out->write( '-----' ).
```

Constants

Statement CONSTANTS

```
CONSTANTS <constant_name> TYPE <type> VALUE <value>.
```

Name of the constant

Type of the constant

Value of the constant
(mandatory)

Examples

```
CONSTANTS my_const1    TYPE i VALUE 12345.
```

```
CONSTANTS my_const2    TYPE string VALUE `Hello World`.
```

```
CONSTANTS my_const3    TYPE C LENGTH 3 VALUE IS INITIAL.
```

Literals

Number Literals

123, -123

- integer numbers
- With or without sign
- Data type: I or P (with 0 decimals)

Text Literals

'LH', '123.45'

- Marked with simple quote (')
- TYPE C, length without trailing blanks

String Literals

`Hello`, `What's up?`

- Marked with backquote (`)
- TYPE STRING

Try it out!

* Example 1: Local Types

.....

* Comment/Uncomment the following lines to change the type of my_var
TYPES my_type TYPE p LENGTH 3 DECIMALS 2.
* TYPES my_type TYPE i.
* TYPES my_type TYPE string.
* TYPES my_type TYPE n length 10.

* Variable based on local type
DATA my_variable TYPE my_type.

```
out->write( `my_variable (TYPE MY_TYPE)` ).  
out->write( my_variable ).
```

* Example 2: Global Types

.....

* Variable based on global type .
" Place cursor on variable and press F2 or F3
DATA airport TYPE /dmo/airport_id VALUE 'FRA'.

```
out->write( `airport (TYPE /DMO/AIRPORT_ID)` ).  
out->write( airport ).
```

* Example 3: Constants

.....

```
CONSTANTS c_text TYPE string VALUE 'Hello World'.  
CONSTANTS c_number TYPE i VALUE 12345.
```

"Uncomment this line to see syntax error (VALUE is mandatory)

* constants c_text2 type string.

```
out->write( `c_text (TYPE STRING)` ).  
out->write( c_text ).  
out->write( '-----' ).
```

```
out->write( `c_number (TYPE I)` ).  
out->write( c_number ).  
out->write( '-----' ).
```

* Example 4: Literals

.....

```
out->write( `12345` ). "Text Literal (Type C)  
out->write( `12345` ). "String Literal (Type STRING)  
out->write( 12345 ). "Number Literal (Type I)
```

"uncomment this line to see syntax error (no number literal with digits)

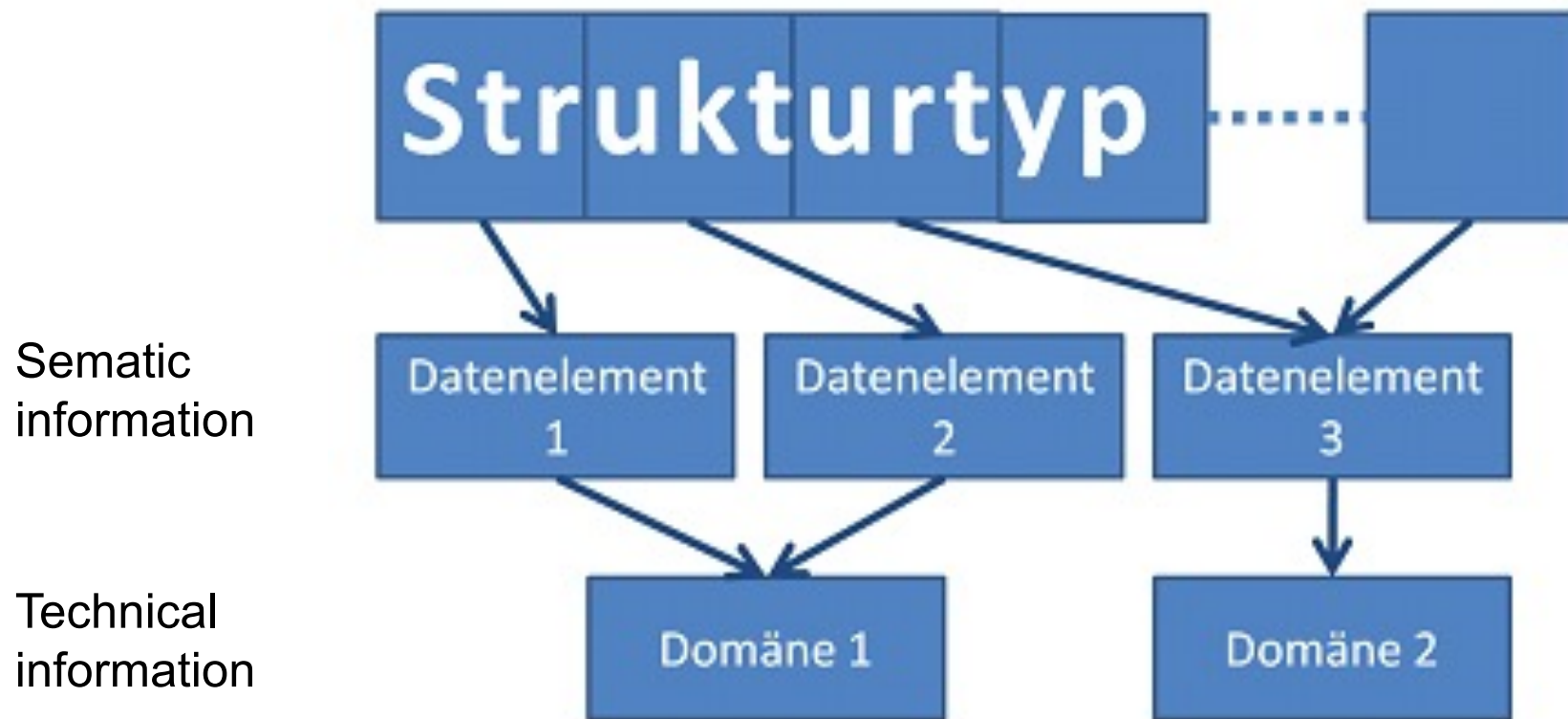
* out->write(12345.67).

DDIC Structured Type

```
Package  
/DMO/FLIGHT  
Table  
/DMO/AGENCY  
Structure  
/dmo/s_ext_incl_agency
```



Two-tier domain approach in the ABAP Dictionary



DDIC

Data Element

- Motivation: definition of global scalars Types in the ABAP Dictionary (business or also semantic Level)
- Create :
- Usage:

The screenshot shows the configuration for the Data Element `/DMO/AGENCY_ID`. It is categorized as a `Domain` with a `NUMC` data type and a length of `6`. The `Field Labels` section defines four labels: `Short` (Agency ID, length 10), `Medium` (Agency ID, length 20), `Long` (Agency Number, length 40), and `Heading` (Agency ID, length 55). The `Additional Properties` section includes search help fields for `Name` and `Parameter`, a `Parameter ID` field, a `Component Name` field, and checkboxes for `Change Document Logging` (unchecked) and `Input History` (checked). The `Bidirectional Options` section shows `Basic Direction` set to `Right to Left` and `BIDI Filtering` checked.

Typing of variables , interfaces , ...

- Motivation: Used to define a DB table
- Create:
- Indices
 - Primary
 - Secondary
- Technical settings

```
1 @EndUserText.label : !Flight Reference Scenario: Managing Agencies'  
2 @AbapCatalog.enhancement.category : #EXTENSIBLE_ANY  
3 @AbapCatalog.tableCategory : #TRANSPARENT  
4 @AbapCatalog.deliveryClass : #A  
5 @AbapCatalog.dataMaintenance : #RESTRICTED  
6 define table /dmo/agency {  
7  
8     key client           : abap.clnt not null;  
9     key agency_id       : /dmo/agency_id not null;  
10    name                 : /dmo/agency_name;  
11    street               : /dmo/street;  
12    postal_code         : /dmo/postal_code;  
13    city                 : /dmo/city;  
14    country_code        : land1;  
15    phone_number        : /dmo/phone_number;  
16    email_address       : /dmo/email_address;  
17    web_address         : /dmo/web_address;  
18    attachment          : /dmo/attachment;  
19    mime_type           : /dmo/mime_type;  
20    filename            : /dmo/filename;  
21    local_created_by    : abp_creation_user;  
22    local_created_at    : abp_creation_tstmpl;  
23    local_last_changed_by : abp_locinst_lastchange_user;  
24    local_last_changed_at : abp_locinst_lastchange_tstmpl;  
25    last_changed_at     : abp_lastchange_tstmpl;  
26    include /dmo/s_ext_incl_agency;  
27  
28 }
```

DDIC

Transparent Table

- Column definition :
 - Columns
 - Part of the primary key ?
 - Initialization
 - Typing

```
1 @EndUserText.label : !Flight Reference Scenario: Managing Agencies'  
2 @AbapCatalog.enhancement.category : #EXTENSIBLE_ANY  
3 @AbapCatalog.tableCategory : #TRANSPARENT  
4 @AbapCatalog.deliveryClass : #A  
5 @AbapCatalog.dataMaintenance : #RESTRICTED  
6 define table /dmo/agency {  
7  
8     key client           : abap.clnt not null;  
9     key agency_id       : /dmo/agency_id not null;  
10    name                 : /dmo/agency_name;  
11    street                : /dmo/street;  
12    postal_code          : /dmo/postal_code;  
13    city                  : /dmo/city;  
14    country_code         : land1;  
15    phone_number         : /dmo/phone_number;  
16    email_address        : /dmo/email_address;  
17    web_address          : /dmo/web_address;  
18    attachment           : /dmo/attachment;  
19    mime_type            : /dmo/mime_type;  
20    filename              : /dmo/filename;  
21    local_created_by     : abp_creation_user;  
22    local_created_at     : abp_creation_tstmpl;  
23    local_last_changed_by : abp_locinst_lastchange_user;  
24    local_last_changed_at : abp_locinst_lastchange_tstmpl;  
25    last_changed_at      : abp_lastchange_tstmpl;  
26    include /dmo/s_ext_incl_agency;  
27  
28 }
```

DDIC

Transparent Table



By maintaining a foreign key check **Non-referential integrity** is defined at the database level.

The definition of this check only has an effect if a Dynpro or Web Dynpro field is defined that creates a reference to the foreign key field.

Transparent Table

Technical settings & expandability

```
1 @EndUserText.label : '!Flight Reference Scenario: Managing Agencies'  
2 @AbapCatalog.enhancement.category : #EXTENSIBLE_ANY  
3 @AbapCatalog.tableCategory : #TRANSPARENT  
4 @AbapCatalog.deliveryClass : #A  
5 @AbapCatalog.dataMaintenance : #RESTRICTED  
6 define table /dmo/agency {  
7  
8     key client          : abap.clnt not null;  
9     key agency_id     : /dmo/agency_id not null;  
10    name               : /dmo/agency_name;  
11    street             : /dmo/street;  
12    postal_code       : /dmo/postal_code;  
13    city               : /dmo/city;  
14    country_code      : land1;  
15    phone_number      : /dmo/phone_number;  
16    email_address     : /dmo/email_address;  
17    web_address       : /dmo/web_address;  
18    attachment        : /dmo/attachment;  
19    mime_type         : /dmo/mime_type;  
20    filename          : /dmo/filename;  
21    local_created_by  : abp_creation_user;  
22    local_created_at  : abp_creation_tstmpl;  
23    local_last_changed_by : abp_locinst_lastchange_user;  
24    local_last_changed_at : abp_locinst_lastchange_tstmpl;  
25    last_changed_at   : abp_lastchange_tstmpl;  
26    include /dmo/s_ext_incl_agency;  
27  
28 }
```

ABAP

ABAP

Properties

"The language"

- is proprietary,
- is typed,
- enables multilingual applications,
- enables SQL access,
- has been extended in an object-oriented manner,
- is platform independent,
- is upwardly compatible.

ABAP Syntax

Allgemeiner Aufbau einer ABAP-Anweisung

<code>XXX</code>	<code>YYY</code>	<code>.</code>
ABAP- Schlüsselwort	Zusätze und Operanden (schlüsselwort-spezifisch)	Punkt als Abschluss der Anweisung

Programmbeispiel

```
PARAMETERS pa_num TYPE i.  
  
DATA gv_result TYPE i.  
  
MOVE pa_num TO gv_result.  
  
ADD 1 TO gv_result.  
  
WRITE 'Your input:'.  
WRITE pa_num.  
  
NEW-LINE.  
  
WRITE 'Result:'.  
WRITE gv_result.
```

ABAP

Instructions



Some important Instructions :

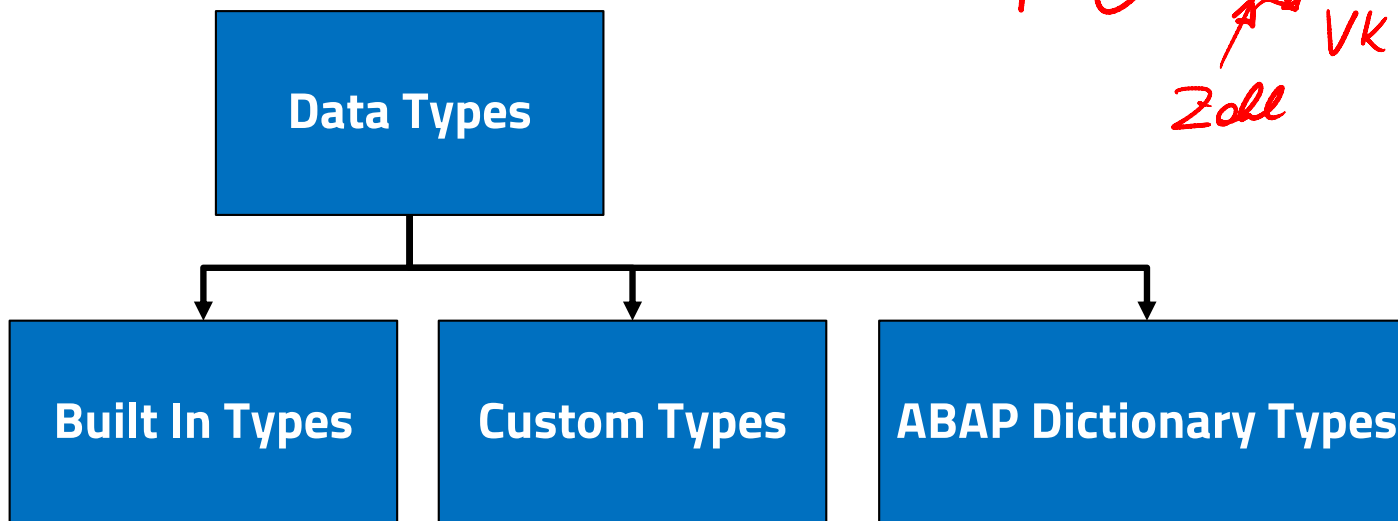
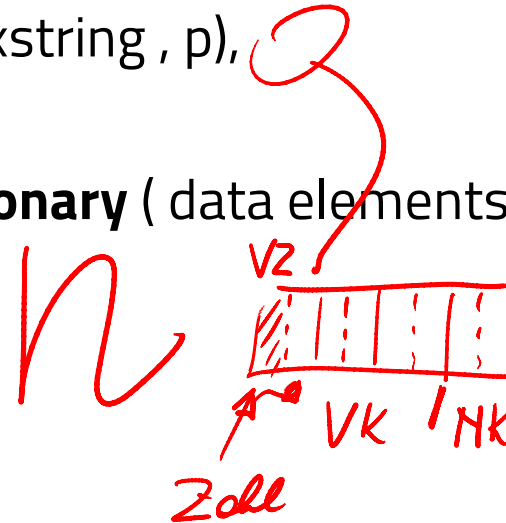
- DATA
- TYPES
- CONSTANTS

ABAP

Data Types

When typing variables we may use:

- **Built In** Types (d, t, i, c, n, f, string, xstring, p),
- **Custom** Types (TYPES) or
- **Global** Types from the **ABAP Dictionary** (data elements, structure types, table types)



Type Complexity

- Elementary Types
- Structured Types

* Line type for data buffering to FG and Eval

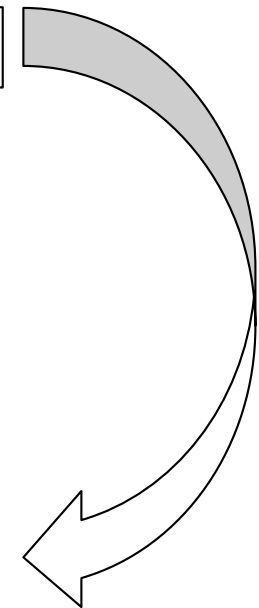
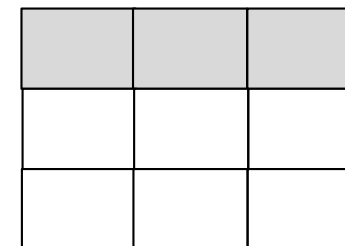
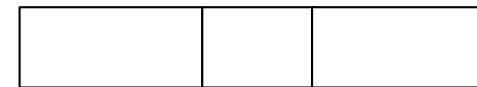
```
TYPES: BEGIN OF gst_fg_eval ,  
       carrid TYPE scarr-carrid ,  
       carrname TYPE scarr-carrname ,  
       eval TYPE zscarreval-eval ,  
END OF gst_fg_eval .
```

- Table-like Types

* Table type (standard table) with row type
gtt_fg_eval TYPE TABLE OF gst_fg_eval .



carrid carrname eval



ABAP Debugging

The screenshot shows the Eclipse IDE interface for debugging an ABAP program. The main editor displays the source code of class `zcl_rs20260204_hc1`. The method `if_oo_adt_classrun-main` is highlighted, showing the line `out->write('Hello Cloud');`. The right-hand side shows the 'Locals' view with a tree structure of variables and their values. The bottom console shows the output 'Hello Cloud'.

```
1 @CLASS zcl_rs20260204_hc1 DEFINITION
2   PUBLIC
3   FINAL
4   CREATE PUBLIC .
5
6   PUBLIC SECTION.
7
8   INTERFACES if_oo_adt_classrun .
9   PROTECTED SECTION.
10  PRIVATE SECTION.
11 ENDCLASS.
12
13
14
15 @CLASS zcl_rs20260204_hc1 IMPLEMENTATION.
16
17 @METHOD if_oo_adt_classrun-main.
18 * Here goes the code
19 out->write( 'Hello Cloud' );
20
21 @ENDMETHOD.
22
23 ENDCLASS.
```

Name	Value
<Enter variable>	
SY-SUBRC	0
ME	{O:111*(CLASS=ZCL_RS20260204_HC1)}
Locals	
OUT	{O:124*(CLASS=CL_OO_ADT_CLASSRUN_OUTPUT)}
LO_DEMO_OUTPUT	{O:126*(CLASS=CL_DEMO_OUTPUT)}
STREAM_HANDLE	{O:127*(CLASS=CL_DEMO_OUTPUT_STREAM)}
HEADING_LEVEL	0
MODE	TEXT
NONPROP	X
STATIC_STREAM_HAN	{O:125*(CLASS=CL_DEMO_OUTPUT_STREAM)}
STATIC_HEADING_LEV	0
STATIC_MODE	HTML
STATIC_HTML_STRING	
STATIC_TEXT_STRING	
STATIC_XML_STRING	
STATIC_JSON_STRING	
HTML_STRING	
TEXT_STRING	
XML_STRING	
JSON_STRING	
CLASSES	[8x1(80)]Sorted(Undue) Table
CLASSRUN	
CLASSRUN_OUTPUT	
STATIC_NONPROP	X

Global Class | Class-relevant Local Types | Local Types | Test Classes | Macros

Console | Problems | Debug Shell

ABAP Console
Hello Cloud

Structured Types

- Structure (Dictionary or program local)
- Transparent Table (Dictionary)
- View (Dictionary)
- CDS (Eclipse)

ABAP Literals

Fixe Datenobjekte ohne Bezeichner

Literale

Zahlenliterale	
positive ganze Zahl :	123
negative ganze Zahl :	-123

Textliterale	
Zeichenketten :	'Hallo'
Bezimazahlen :	'123.45'
Gleitpunktzahlen :	'123.45E01'

Fixe Datenobjekte mit Bezeichner

Konstanten

```
CONSTANTS gc_myconst TYPE type_name VALUE {literal | IS INITIAL}.
```

ABAP

Examples: Syntax

```
* comments ... }  
* comments ... }  
* comments ... }  
  
PARAMETERS pa_num TYPE i. " comments ...  
DATA gv_result TYPE i. " comments ...  
  
MOVE pa_num " comments ...  
  TO gv_result. " comments ...  
  
ADD 1 TO gv_result.  
  
WRITE: 'Your input: ' ,  
      pa_num. }  
  
NEW-LINE.  
  
WRITE: 'Result: ' , gv_result. }
```

Kommentare
(ganze Zeilen)

Kommentare
(Zeilenrest)

Kettensatz

Kettensatz

Return Codes (*sy-subrc*)

- Important System fields are in the ABAP Dict structure *SYST* defined .
- The most important is *sy-subrc* , which contains the statement about the exit value the previous operation returns .

IF *sy-subrc* = 0. " Everything is ok

...

ELSE. " There something went wrong

...

ENDIF.

```
* Reading the FG from the DB table SCARR
SELECT SINGLE carrname FROM scarr INTO gd_carrname
WHERE carrid = pa_car .
IF sy-subrc <> 0.
    message e002(zws19) with pa_car .
ENDIF.
```

ABAP

Modularization

- Level local – local reuse
 - Classic: Subprograms
 - Classic: SAP GUI modules
 - Classic: Event blocks
- Level global – global reuse
 - Classic: Function blocks (function groups)
 - **ABAP classes (ABAP interfaces)**
 - **New: CDS**
 - **New: AMDP = Stored Procedure with OO**

Idea Encapsulation

```
*&-----*
*& READ_EVAL
*&-----*
* Reading the review
*-----*
* -->ID_CARRID ID of the FG
* <--ED_EVAL rating
*-----*
METHOD read_eval          IMPORTING id_carrid TYPE scarr-carrid
                          EXPORTING ed_eval TYPE zscarreval-eval .

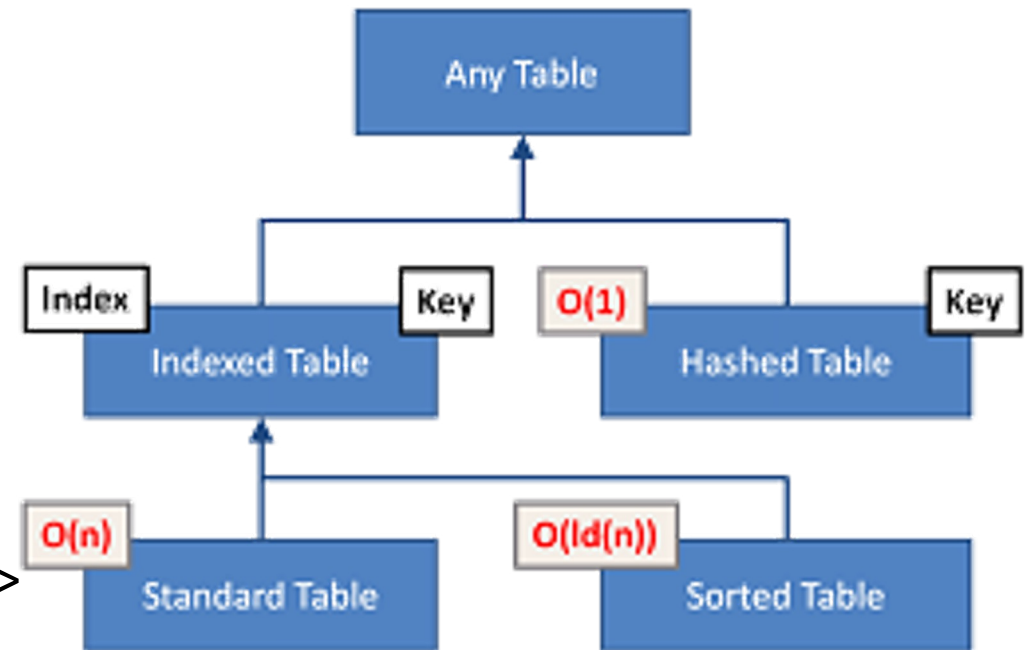
SELECT SINGLE eval FROM zscarreval INTO ed_eval
WHERE carrid = id_carrid .

ENDMETHOD. " READ_EVAL
```

DDIC: Structure

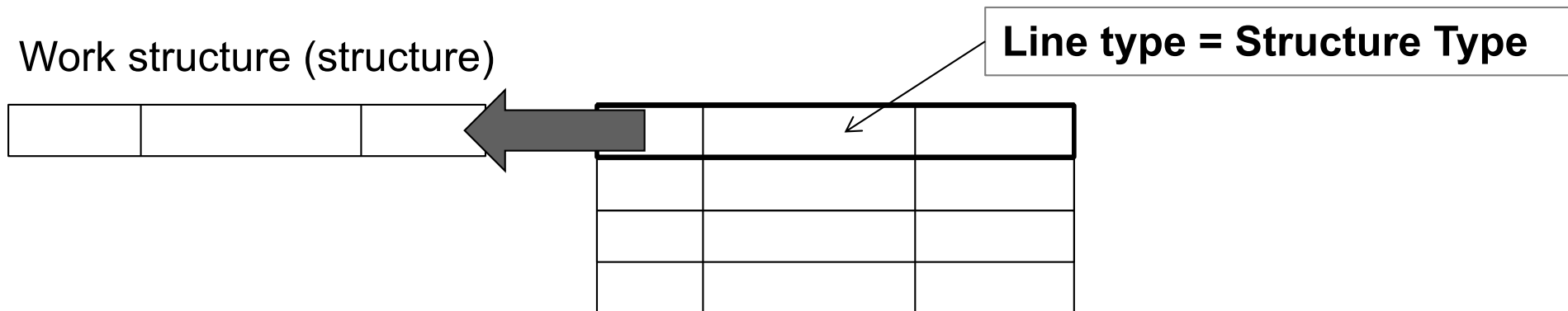
- Motivation: definition of global structured type (comparable : Record or one-dim array)
- Create :
- Usage: Typing of variables , interfaces , table types , ...
- Access to components : hyphen semantics
- Copying content : MOVE-CORRESPONDING

- Motivation: Defines the global type of an internal table
- Create : TA SE11
- Definition of one internal Table
- Important Instructions :
 - Sort < table > BY < column > [ascending|decending]
 - APPEND <row> TO <table> .
 - DELETE <row> FROM <table> .



ABAP Internal Table

Is in main memory!



Type of table

- Standard
- Sorted
- Hashed

Key

- Components
- Sequence
- Uniqueness

Output table

```
*&-----*
*& WRITE_FG_EVAL
*&-----*
* Output the table to the list
*-----*
* -->IT_FG_EVAL Table of FV and eval
*-----*
METHOD write_fg_eval IMPORTING it_fg_eval TYPE gtt_fg_eval .

DATA: ls_fg_eval LIKE LINE OF it_fg_eval .

LOOP AT it_fg_eval INTO ls_fg_eval .
WRITE: / ls_fg_eval-carrid ,
       ls_fg_eval-carrname ,
       ls_fg_eval-eval .
ENDLOOP.

ENDMETHOD. " WRITE_FG_EVAL
```

Read into internal table

```
*&-----*
*& READ_FG_EVAL
*&-----*
* Determine the FG for an eval
*-----*
* --> ID_EVAL Eval
* <--ET_FG_EVAL FG to eval
*-----*
METHOD read_fg_eval IMPORTING id_eval TYPE zscarreval-eval
EXPORTING et_fg_eval TYPE gtt_fg_eval .

DATA: ld_carrid TYPE scarr-carrid ,
      ls_fg_eval LIKE LINE OF et_fg_eval ,
      ld_carrname TYPE scarr-carrname .

* Bulk operation with SELECT
SELECT carrid FROM zscarreval INTO ld_carrid
WHERE eval = id_eval .
* Fill auxiliary structure
ls_fg_eval-eval = id_eval .
ls_fg_eval-carrid = ld_carrid .
* Reading the fg from the db tabel scarr
me->read_carrname
exporting
    ld_carrid
importing
    ld_carrname .

ls_fg_eval-carrname = ld_carrname .
* Create entry in the internal table
APPEND ls_fg_eval TO et_fg_eval .
ENDSELECT.

ENDMETHOD. " READ_FG_EVAL
```

ABAP console

Modularization

Motivation FuGr/FuBa

Function Group / Function Module



```
<<FuGr>>  
ZFG_<MATNR>_TRAVEL  
---  
TRAVELID:String  
---  
+ <<FuMo>>ZFM_<initials><date>_TRAVEL_READ
```

Function Group / Function Module

- Definition

- Interface

- IMPORTING
 - EXPORTING
 - CHANGING
 - EXCEPTIONS

```
FUNCTION Z  
IMPORTING  
  
    VALUE(IM_P1) TYPE type1 OPTIONAL  
    VALUE(IM_P2) TYPE type2 DEFAULT def_value  
EXPORTING  
    EX_P1        TYPE REF TO STRING  
CHANGING  
    CH_1         TYPE ANY  
TABLES  
    TAB_P1       LIKE structure_name  
    TAB_P2       TYPE tab_type  
RAISING  
    CX_SY_ZERODIVIDE  
    RESUMABLE(CX_SY_ASSIGN_CAST_ERROR).  
ENDFUNCTION.
```

- Use Auto completion

- CALL FUNCTION `<name>`

Function Module

Does it do anything useful?



```
FUNCTION zfm_rs20250202_travel_read
  IMPORTING
    VALUE(id_key) TYPE i
  EXPORTING
    ed_key TYPE i.

  ed_key = id_key.

ENDFUNCTION.
```

Function Module



```
FUNCTION z8820038_ws19_fg_r_eval.
```

```
*"-----  
**" Local Interface:  
*" IMPORTING  
*" REFERENCE(ID_CARRID) TYPE SCARR-CARRID  
*" EXPORTING  
*" REFERENCE(ED_EVAL) TYPE ZSCARREVAL-EVAL  
*"-----
```

```
SELECT SINGLE eval FROM zscarreval INTO ed_eval  
WHERE carrid = id_carrid .
```

```
END FUNCTION.
```

ABAP Object Oriented (OO)

Motivation 00

Classes

```
ZCL_<MATNR>_TRAVEL
---
- MS_TRAVEL:/DMO/TRAVEL
---
+ READ_TRAVEL(ID_TRAVELID):ES_TRAVEL
```

Motivation 00

Classes

```
ZCL_<MATNR>_CARRIER
---
- MS_SCARR:String
---
+ CONSTRUCTOR(ID_CARRID)
- READ_SCARR(ID_CARRID):MS_SCARR
+ GET_SCARR():ES_SCARR
```

```
MS_SCARR-CARRID = ,LH'
---
+ GET_SCARR():MS_SCARR
```

ABAP OO

Motivation

- For what? What for?
- Procedural vs. Object Orientation
- Properties of the OO
- Level of coverage in ABAP OO
 - Inheritance
 - Polymorphism
 - Encapsulation
- Separate definition and implementation

Modeling with the Unified Modeling Language (UML)

- Class diagram
 - Class
 - Attribute
 - Method
 - Association
 - Cardinality
- Sequence diagram

ABAP OO

Class and Object

- Dynamic vs. Static Type
- Visibilities
- Attribute
- Method
- Event
- Construction plan / blueprint

ABAP OO

Types of ABAP classes

- **Normal Class**
- **Exception class**
- ABAP Units Test class
- Persistent object

Properties of ABAP classes

- Abstract
- Final

ABAP OO



Attribute

- Static/Class
- Instance/Object
- Visibilities/Public, Protected, Private

ABAP OO

Definition **Method**

- Static/Class
- Instance/Object
- Visibilities/
Public,Protected,Private

ABAP OO



Testing Classes

- ABAP units

States of objects

- Create (Constructor)

```
IKT.  
* Objekt instanziiieren  
  CREATE OBJECT go_fg  
  EXPORTING  
    id_carrid = pa_car.  
* Fehlerbehandlung
```

- Terminate (Garbage Collector - GC)
 - Using CLEAR the object reference initialized become and thereby the GC collects the object and gives the resources free .
 - In principle, if a object is not reachable anymore , then the GC frees those affected resources

ABAP OO



Calling Methods:

- Static
- Instance
- Functional

```
cl_salv_table=>factory(  
* EXPORTING  
* list_display = IF_SALV_C_BOOL_SAP=>FALSE  
* r_container =  
* container_name =  
  IMPORTING  
    r_salv_table = go_alv  
  CHANGING  
    t_table = gt_fg_eval  
  ).
```

```
go_alv->display( ).
```

```
gd_carrname_single = go_fg->get_carrname( ).
```

Use of attributes

ABAP Exceptions

ABAP OO

Exceptions - History



1. Classic

IF sy-subrc = 0.

...

ENDIF.

2. Intermediate solution on the way to Object Orientation

CATCH SYSTEM EXCEPTION.

...

ENDCATCH. + sy-subrc Evaluation

3. Object Oriented exception concept

TRY.

...

CATCH.

...

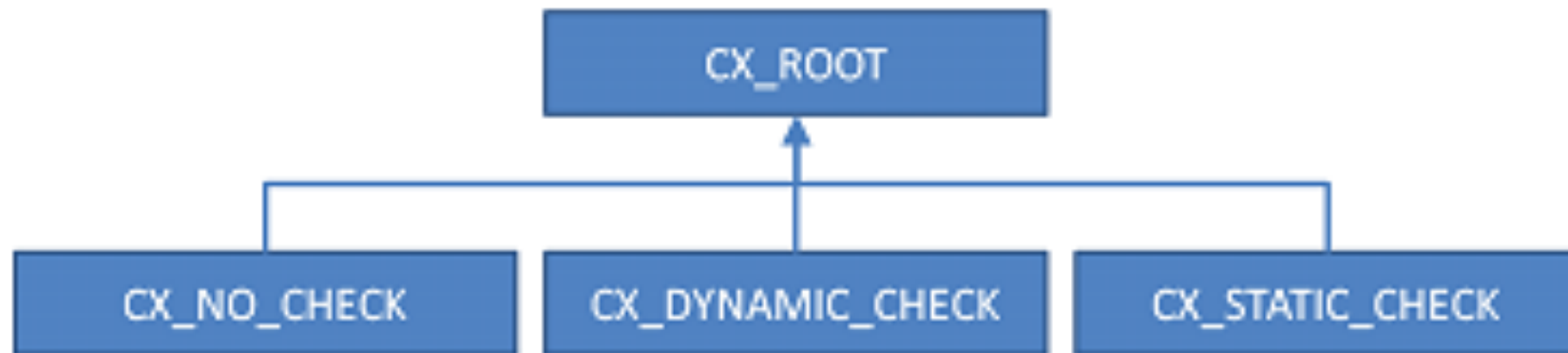
ENDTRY.

ABAP OO

OO Exceptions

Inheritance hierarchy for exceptions

CX_ROOT



ABAP OO

OO Exceptions

Create exception

Exception text



ABAP OO

OO Exceptions

Trigger exception

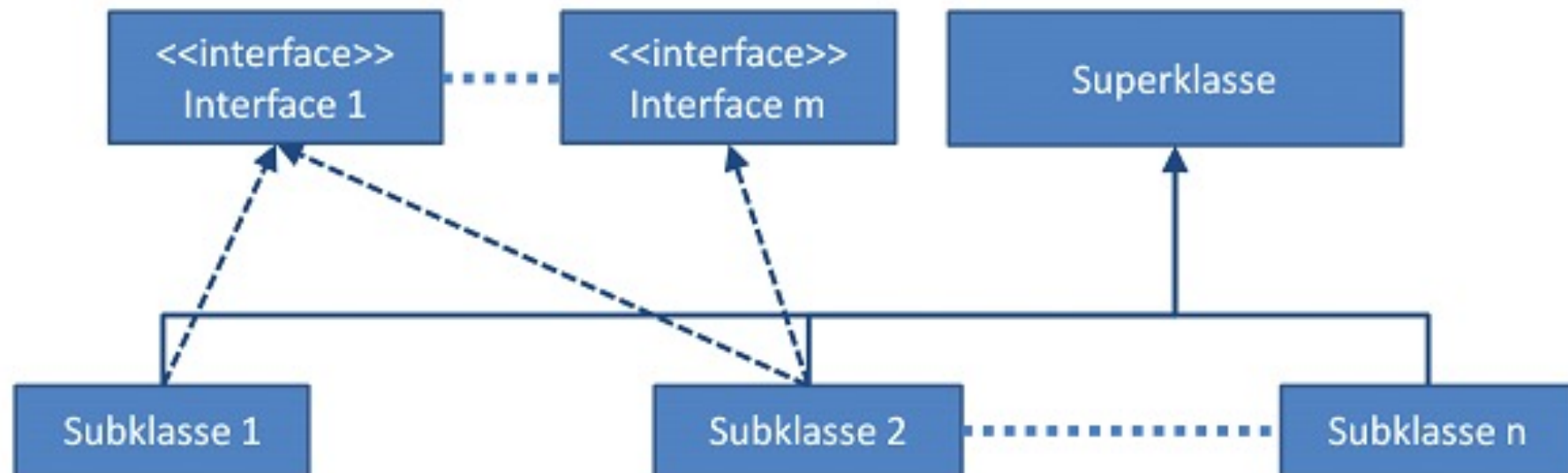
RAISE EXCEPTION



ABAP OO

Inheritance

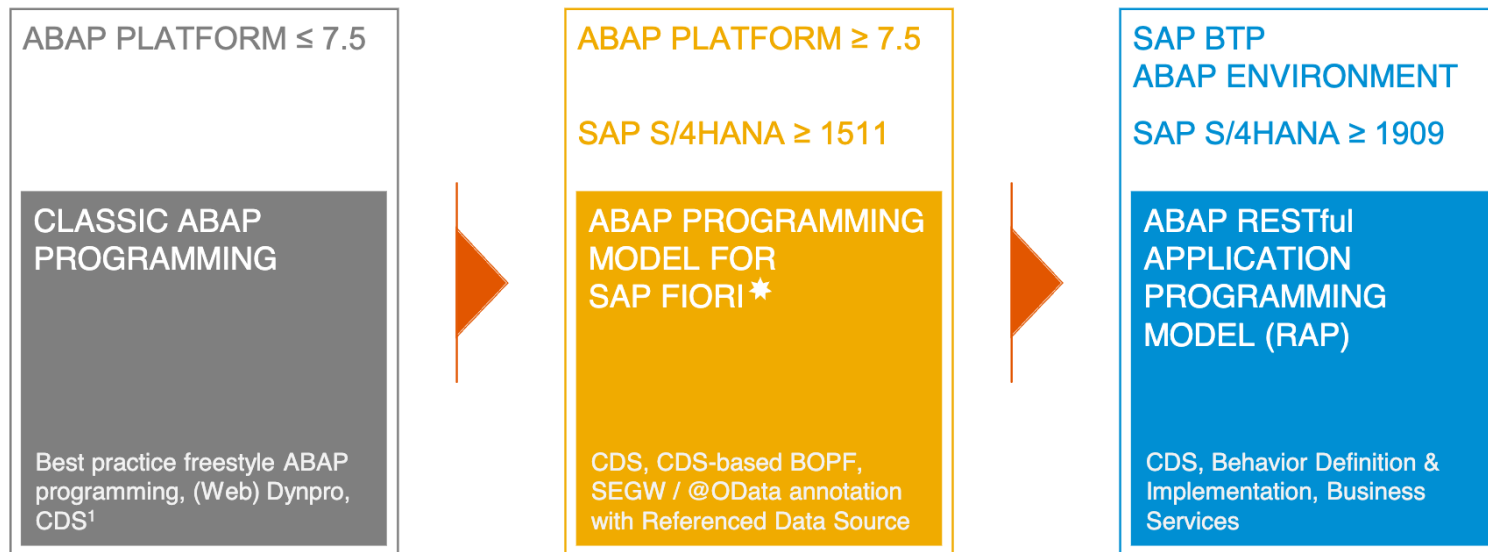
- Redefinition
- Casting (Up/Down)



ARCHITECTURE

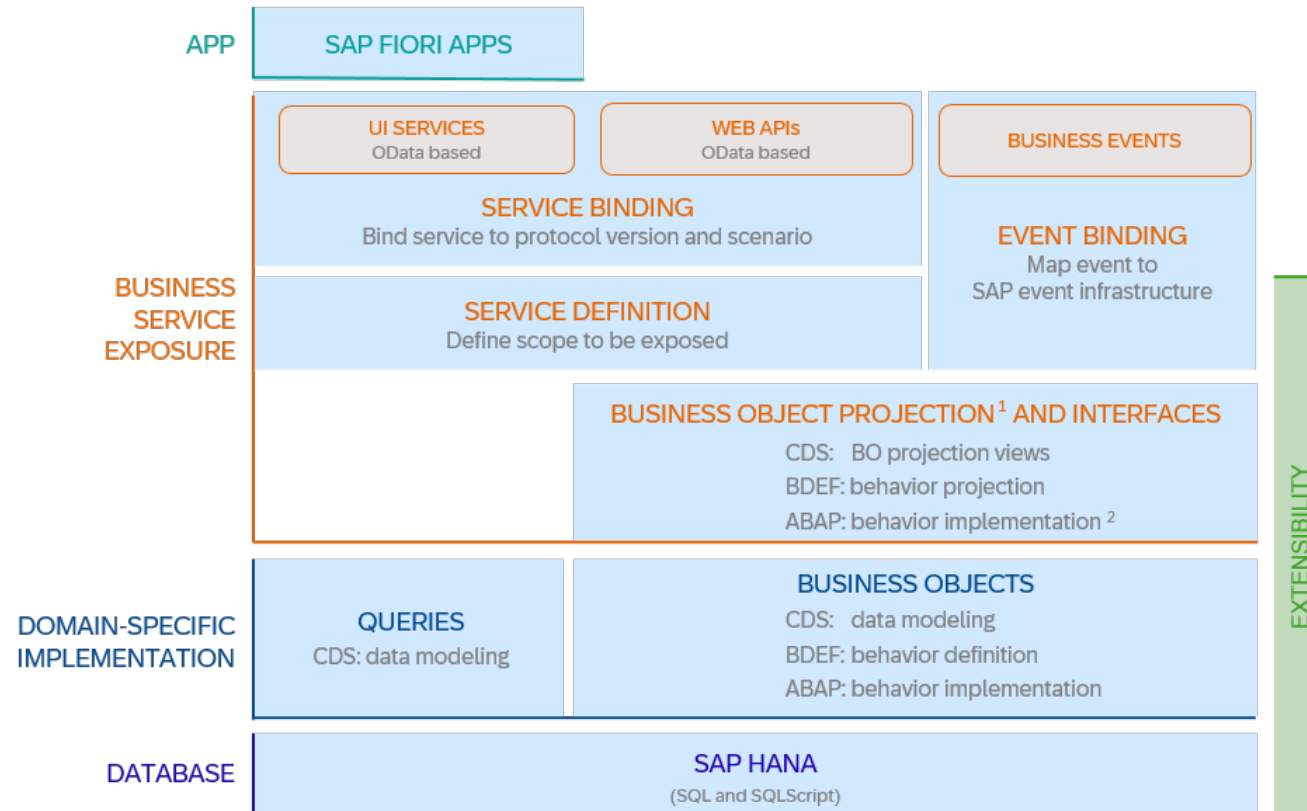
Evolution

Evolution of the ABAP programming model



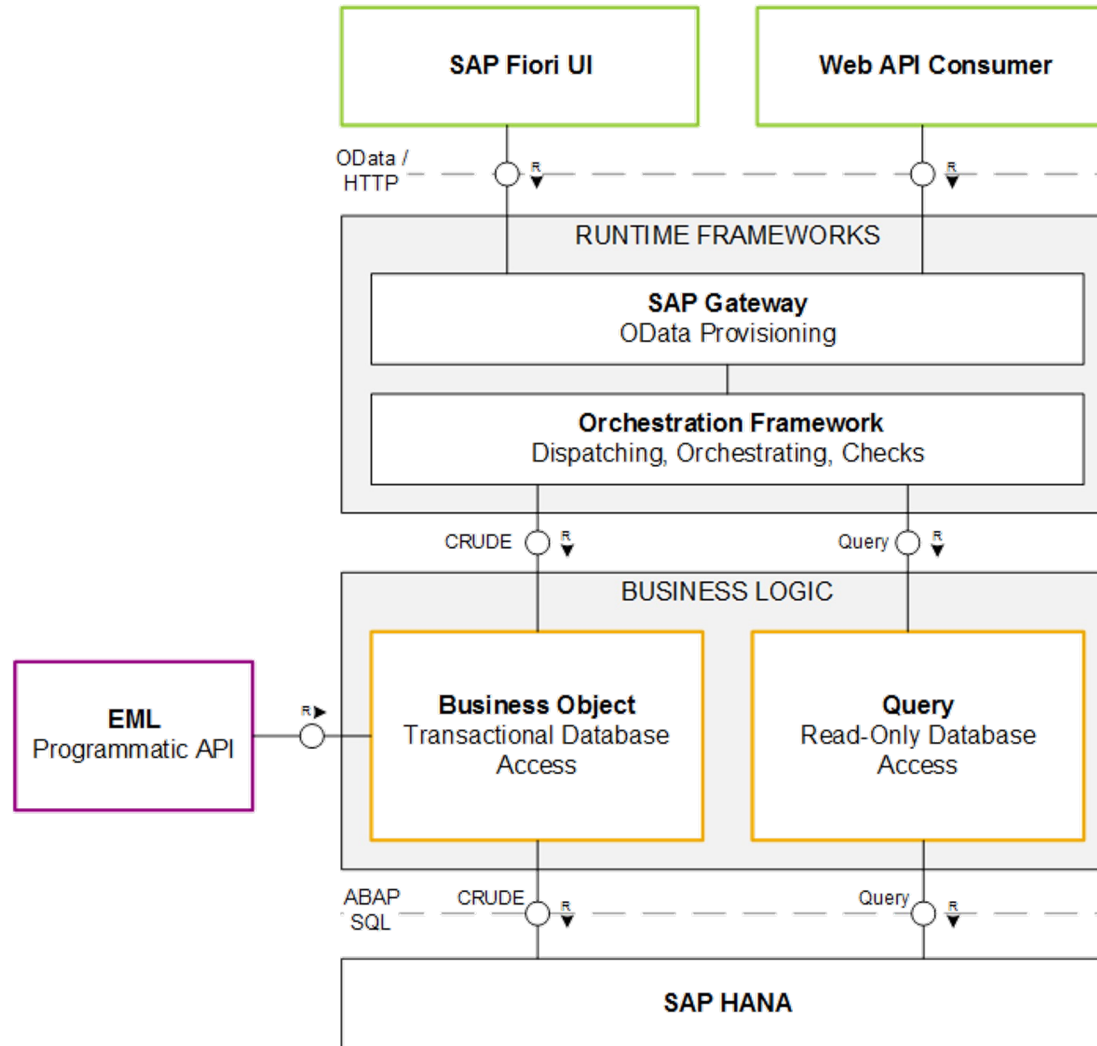
¹ starting with release 7.4 SPS05

RAP Big Picture



¹ Also called Service Projection ² Not applicable for RAP BO interfaces

RAP Runtime



Eclipse

MODERN DEVELOPMENT TOOLSET

Fully eclipse-based
Syntax check, Code completion
Syntax highlighting, Pretty printing
Navigation, Search, Quick Fixes
Available for ABAP releases ≥ 7.31

QUALITY ASSURANCE

Static code checks (ATC, CVA) with
remote and local scenarios
Unit testing incl. isolation frameworks
Test seams and injections

SUPPORTABILITY

Debugging, profiling, tracing
Static and dynamic logging
Runtime monitoring and analysis

OData

Technology Base



OData V2 (O2)

- Classic standard (around 2010), **built into SAP Gateway**.
- Typically implemented in **SEGW (Service Builder)** or via **CDS** → **OData Publish (@OData.publish: true)**.
- **Protocol:** OData V2 (Atom/XML, JSON).
- **Framework:** SAP Gateway Runtime.

OData V4 (O4)

- Newer standard (from 2014), broadly supported in SAP only **from ABAP Platform 7.55+**.
- Implemented only via **RAP (RESTful ABAP Programming Model)** → SEGW is no longer used.
- **Protocol:** OData V4 (JSON only, more compact, modern).
- **Framework:** RAP BO (Business Object) + OData V4 Framework.

OData

Data Model & Metadata



- **02:** EDMX metadata in V2 format. Supports EntitySets, Navigation, Associations.
- **04:** EDMX in V4 format. Supports *Singletons, Functions, Actions, Complex Types*, richer navigation.

OData

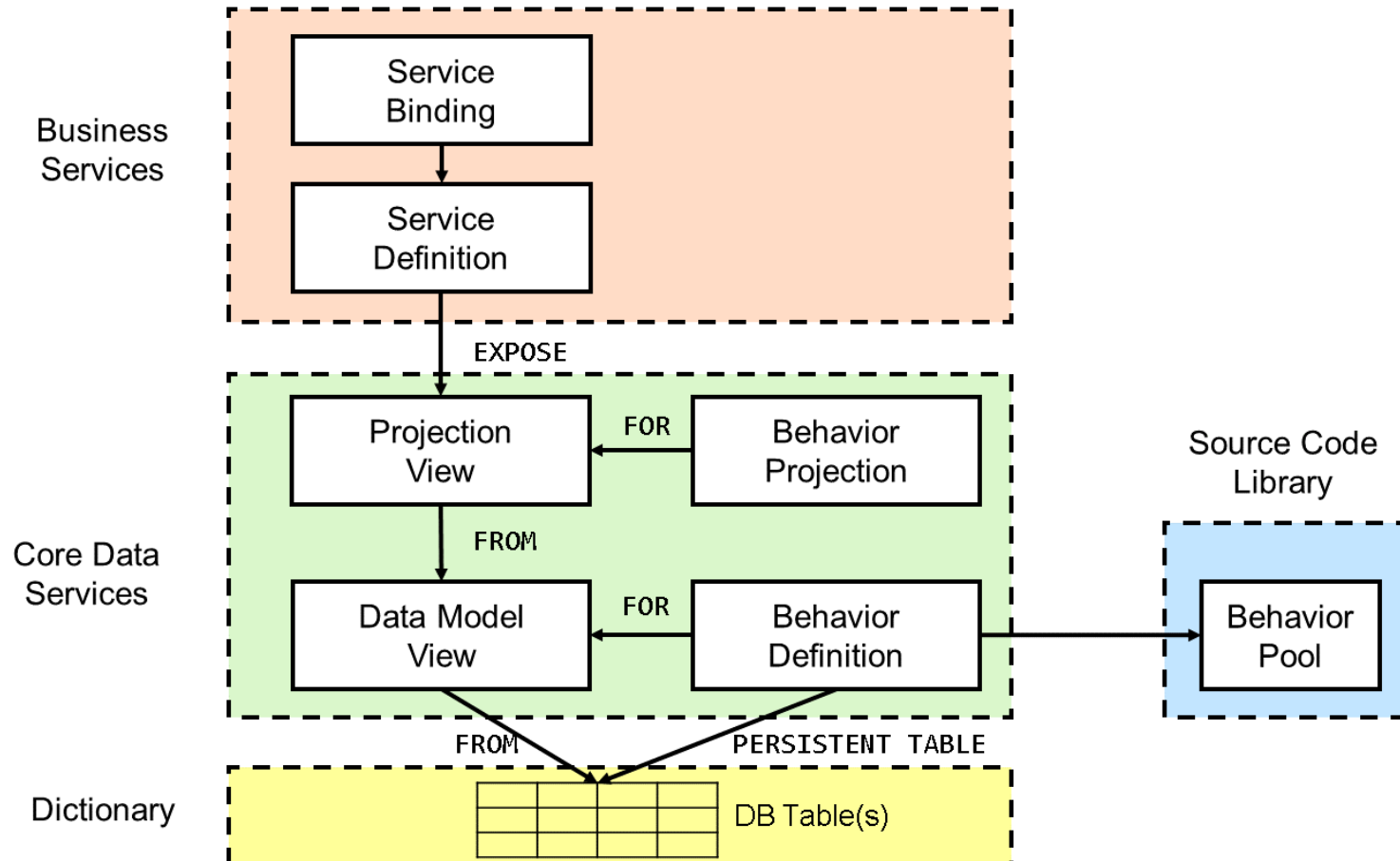
Feature Comparison



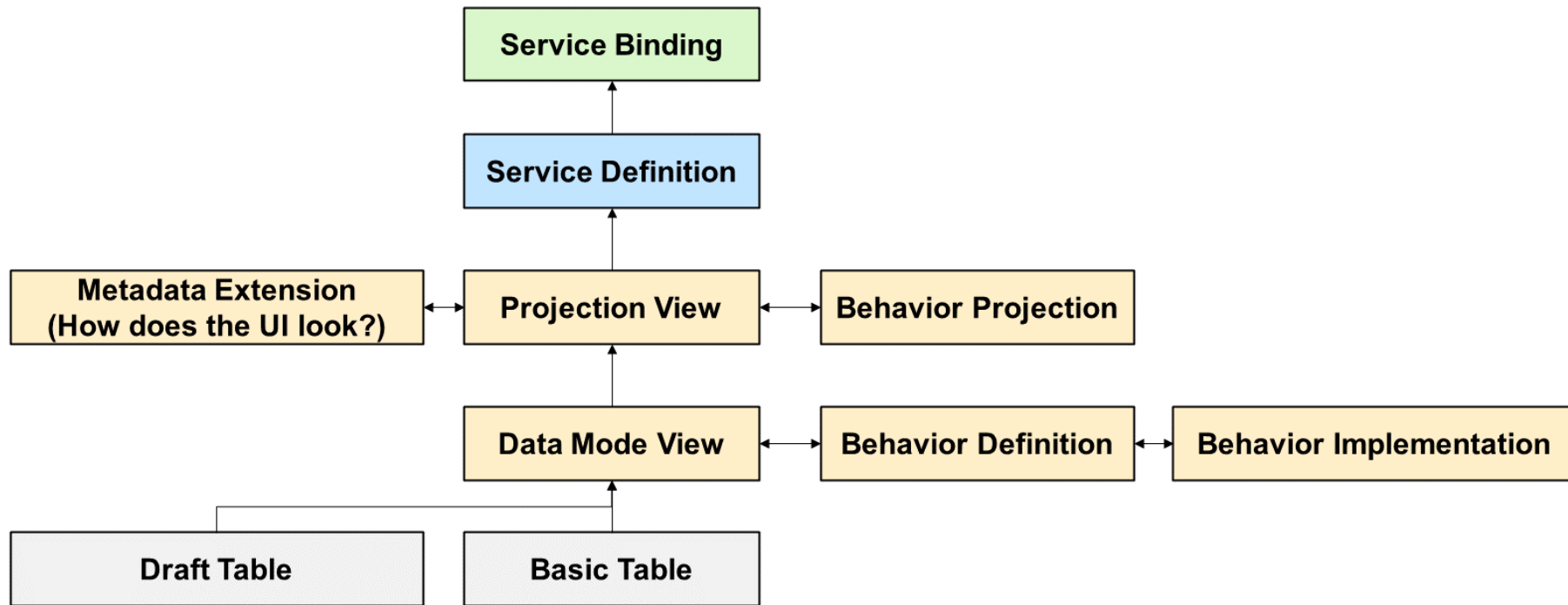
Area	OData V2 (O2)	OData V4 (O4)
Development	SEGW, MPC/DPC methods, CDS Publish	RAP BO Definition (Behavior, Service Definition, Service Binding)
Formats	XML + JSON	JSON only
Batch Support	Yes (\$batch requests)	Yes, cleaner and more efficient
Deep Insert/Update	Complex, partly custom	Fully supported in RAP
Actions/Functions	Limited	Fully standardized
Annotations	Basic	Rich (UI, semantic, CAP-like)
Draft Handling	Only via custom coding	Native in RAP
Value Helps (F4)	Manual implementation	Provided via CDS annotations
Query Options	\$filter, \$expand, \$select, \$orderby	Extended: \$count, \$search, \$compute, richer expressions
Delta Handling	Token-based but complex to implement	Native Delta Queries in RAP
Optimization	Limited paging and performance control	Server-driven paging, optimized query planning

APPLICATION

Developing a RAP application consists of the following main steps



Generate additional objects



Create a RAP Application



<https://developers.sap.com/group.abap-build-fiori-element-rap.html>

Package: \$<Initials><Date>_ANALYSIS

Table: Z<Initials><Date>_TRAV (max 16 char!)

RAP Application:

Build an SAP Fiori elements App Using the ABAP RESTful Application Programming Model (RAP) – Beginner [RAP100]

👤 Beginner ⏱ 1 hr, 25 min 📁 SAP BTP ABAP environment, Beginner, Tutorial, ABAP Development, SAP S/4HANA Cloud

Learn how to build an SAP Fiori App elements using the ABAP RESTful Application Programming Model. This is the beginner group for RAP100.

1.

TUTORIAL +

Create Database Table and Generate UI Service

🕒 20 min.

Start group

2.

TUTORIAL +

Enhance the Business Object Behavior With Unmanaged Internal Numbering

🕒 20 min.

RAP Creation Steps

Generate ABAP Repository Objects

Preview Generator Output

List of repository objects that are going to be generated

Repository Object Name	Type	Operation
ZI_RS20250717_TRAVELTP	Data Definition	
ZI_RS20250717_TRAVELTP	Behavior Definition	
ZCL_RS20250717_BP_TRAVELTP	Class	
ZRS2025717_DTRAV	Database Table	
ZC_RS20250717_TRAVELTP	Data Definition	
ZC_RS20250717_TRAVELTP	Behavior Definition	
ZC_RS20250717_TRAVELTP	Metadata Extension	
ZRS20250717_UI_TRAVELTP	Service Definition	
ZRS20250717_UI_TRAVELTP_04	Service Binding	

Navigation: < Back, Next >, Finish, Cancel

Code

EARLYNUMBERING_CREATE



METHOD earlynumbering_create.

DATA:

entity TYPE STRUCTURE FOR CREATE
ZRS00_R_TravelTP,

travel_id_max TYPE /dmo/travel_id,

" change to abap_false if you get the ABAP Runtime error
'BEHAVIOR_ILLEGAL_STATEMENT'

use_number_range TYPE abap_bool VALUE abap_false.

"raise EXCEPTION TYPE cx_abap_behv_runtime_error.

raise EXCEPTION type /iwbep/cx_v4_conversion.

"Ensure Travel ID is not set yet (idempotent)- must be
checked when BO is draft-enabled

LOOP AT entities INTO entity WHERE TravelID IS NOT
INITIAL.

APPEND CORRESPONDING #(entity) TO mapped-travel.

ENDLOOP.

```
DATA(entities_wo_travelid) = entities.  
  
"Remove the entries with an existing Travel ID  
  
DELETE entities_wo_travelid WHERE TravelID IS NOT INITIAL.  
  
"determine the first free travel ID without number range  
  
"Get max travel ID from active table  
  
SELECT SINGLE FROM zrs00_ atray FIELDS MAX( travel_id ) AS travelID INTO @travel_id_max.  
  
"Get max travel ID from draft table  
  
SELECT SINGLE FROM zrs00_ datrav FIELDS MAX( travelid ) INTO @DATA(max_travelid_draft).  
  
IF max_travelid_draft > travel_id_max.  
travel_id_max = max_travelid_draft.  
  
ENDIF.  
  
"Set Travel ID for new instances w/o ID  
  
LOOP AT entities_wo_travelid INTO entity.  
travel_id_max += 1.  
  
entity-TravelID = travel_id_max.  
  
APPEND VALUE #( %cid = entity-%cid  
  
%key = entity-%key  
  
%is_draft = entity-%is_draft  
  
) TO mapped-travel.  
  
ENDLOOP.  
  
ENDMETHOD.
```

TOOLS

Naming Conventions

l Local data object (DO)*

*g * Global DO*

i Importing parameters*

*e * Exporting parameters*

*r * Returning parameters*

c Changing parameters*

*+d_ * elementary DO*

*+s_ * structured DO*

*+t_ * table-like DO*

*+o_ * Object/ instance reference variable*

*+ r _ * data reference variable*

*< fs _ * > Field symbol*

*pa _ * Parameters*

*so_ * Select-Options*

Literature List

www.wikipedia.org

developers.sap.com

help.sap.com

nor.gmbh,rolandschwaiger.at



Recommendation



First Assignment


Register for BTP Trial

- Follow instructions on <https://www.sap.com/products/technology-platform/trial.html>
- Send screen shot of your sub account to my e-mail address
- Due to 19.12.2025
- Precondition for participating the course



Second Assignment

Instantiate ABAP Environment on SAP BTP, Install Eclipse with ADT and Create an ABAP Cloud Project

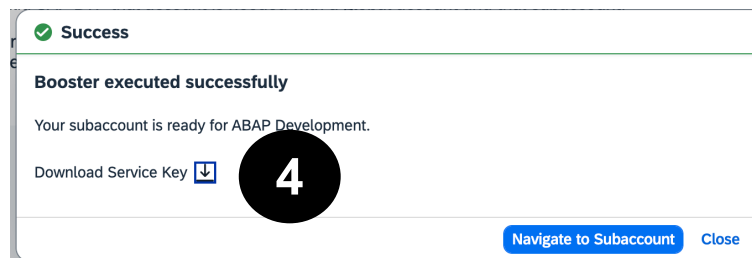
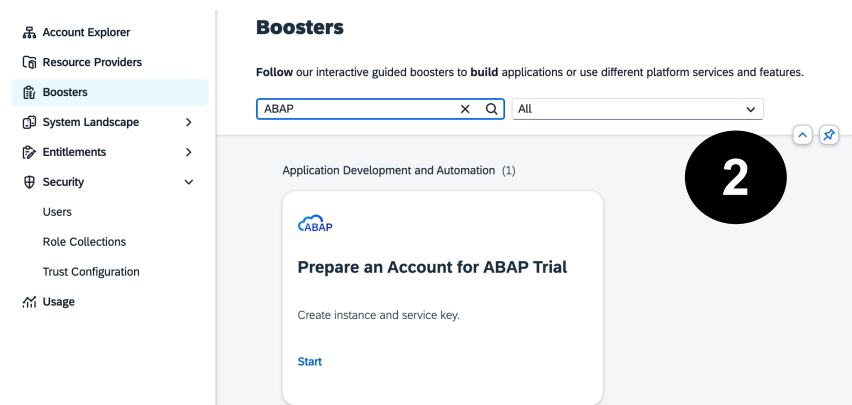
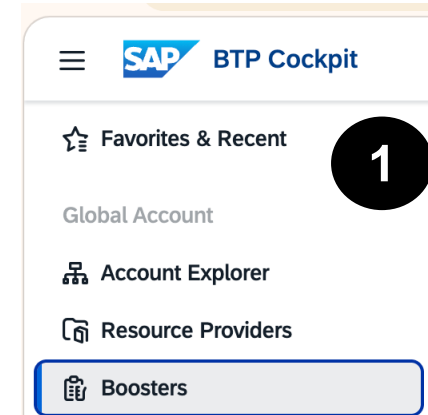
- Follow instructions on
 - Create ABAP Environment and save key:
 - See page 10
 - Install Eclipse:
<https://developers.sap.com/tutorials/abap-install-adt.html>
 - Create an ABAP cloud project in Eclipse:
 - <https://developers.sap.com/tutorials/abap-environment-create-abap-cloud-project.html>
 - See page 11
- Send a screen shot of your cloud project to my e-mail address, looks like > 
- Due to 31.01.2026
- Precondition for participating the course



Second Assignment

Instantiate ABAP Environment on SAP BTP

1. Select Booster
2. Search "Prepare an Account for ABAP Trial"
3. Select Tile, Press Start and wait for the Booster to finish
4. Download Service Key
5. Navigate to subaccount



Second Assignment

Create Cloud Project – Select URL

1. Select ABAP Cloud Project
2. Click the [Extract](#) link
3. Click the **Import** button on the “Extract Service Instance URL” screen. Select the key file you have downloaded before.
4. Click the **Copy to Clipboard** button and **Close**
5. Paste the URL in the “ABAP Service Instance URL” input field
6. Choose Next and Choose the button “Open Logon Page in Browser” (if this is not working then choose the second button and copy the url to the clipboard and paste it in your browser)

