Table of Contents

[1. MO: 2](#_Toc113900308)

[2. BO: 3](#_Toc113900309)

[Business service: 3](#_Toc113900310)

[3. Zone 4](#_Toc113900311)

[1.query zone: 4](#_Toc113900312)

[2.info zone: 4](#_Toc113900313)

[3.map zone: 4](#_Toc113900314)

[4. Scripts: 6](#_Toc113900315)

[1 .Service script: 6](#_Toc113900316)

[2 .Plug-in script: 6](#_Toc113900317)

[3 .BPA script: 6](#_Toc113900318)

[5. UI map: 7](#_Toc113900319)

[6. Change Handler: 8](#_Toc113900320)

[7. Algorithm 9](#_Toc113900321)

ORMB Components Overview

# MO:

1.custom entity.

2.you have to write a logic then you can create first MO.

3. if MO is created then automatic pojo classes are created.

4. Mo have at least 1 Fk ref. it display info for Mo

5. physical representation of tables.ref the gr of tables that holds objects of physical data.

6. Why MO’s: we don’t like directly perform to logical operation so in that case we can use mo.mo can determine how to satisfy the request.

service program: mapping the XMl representation .

info string: we can display description of particular entity. Using information algo.

referential integrity:

Clob: subset of table have a special column that can hold xml document.

xml document: implementation to hold the additional attribute.

metadata: it is used for many other framework process.

# BO:

1.perform crud operation. logging grouping(logical representation)

based on req you can invoke the add,delete,change opration.

eg: file request type.

2.maintenan a flow for perticular process.

information: -->Entity info display.

pre-processing:-->perform logic before starting intial state.dependent opration.modify the data before validation.

validation:-->certain condn for validatate.you can not update validation state .perticular entity validate.

post-processing: once you complete the entire process after you want to trigger it then you can use post-processing.

Audit: capture the event. capture the changes and store the XMl. compare the after change and before change.

# Business service:

1.communicate with DB.

service program to execute BS

2. if we want execute business logic without BO that time we use BS

3.interlink with the Service program (SP Schema map with the BS)

4. Java based bs: when we want to execute program base on java (email send)

-->perform calculation then you can use java bad bs.

5. Zone based bs: need execute SQL query and fetch the result and show

-->Fixed program.

Hard parameter: abstract methods

soft parameter: user input access.

# Zone

1. Info 2. Query 3. Map

1.fetch data in database and display the result in ui.

2.provide to template to display the ui map.

display the result in tabular format

3:In zone you can only perform read operation.you can’t perform update or add operation.

multiple sql: we do have placeholder to have multiple sql statements

Zone visibility service script fun: 1. display whether we want to display zone.it returns output in Boolean parameter.

2. based on certain condition true or false in that condition zone is display otherwise its hidden.

Pre-processing service script: in that service script we can write a business logic before that executing our sql.

Record count display: restrict the records that i want to show. suppose i want to show only 5 records so i can give the no here.

Filter Ui map: access to inputs to users providing and we can provide some validation.

user filter: configure fields to allow the user to limit the search based on their inputs.

Hidden filter: this filter are not expose to user.

Enable pagination: how many records i want to display in page that i control to pagination.

initial display column: no of column you retrieve a zone you need to mention here because This parameter controls the columns.

Broadcast Information: you want to click some row then this parameter control to broadcasting.

### 1.query zone:

show the filter. then execute the SQL query

### 2.info zone:

1.execute the query then display and if you add some filter then you can add on it

2. execute the SQL query first display the filter.

### 3.map zone:

showing the result in ui map. A map zone displays information related to one or more objects. For objects that are maintained using portal-based pages, the standard display of the object is using a map zone. Map zones may be configured to display an explicit map or may dynamically display the map based on the configuration of the record being displayed.

Because the information displayed in a map zone is configured using HTML, virtually any format supported by HTML is possible. The following is an example of a typical map zone used to display information about an object.

-->1.explicit-populated by specific bs, bo

-->2.derived-conditional specify ui.

# Scripts:

## 1 .Service script:

1.commonly logic write.

2. you can invoke anywhere.

3.perfom basic validation. (required=true, datatype=lookup).

4.ss can be secured because ss can optionally application service and the user must have access rights to these services to execute the ss.

5. service script have schema because ss defines input & output fields.

global variables declare: prefix with $$.it is used for default value for a schema element.

## 2 .Plug-in script:

1.specific spot you can validate.

2.a plug-in script input & outputs defines by its plugin spot that’s why no schema to allowed in plug-in script.

## 3 .BPA script:

1.it is used to ui to guide to user through business process.

2.this script is asked to user to enter info and then invoke the server based script.

e.g: add a person for existing account.

BPA area: when a bpa script execute. and it can open an area. it can display object

object display area:

Data Area: each entry references an object that has a schema.

Invoke map: display ui map to capture inputs for user. and it display 3 ways: 1.bpa area 2. object display area, 3.pop-up window.

page area: first is show the white page and then display the ui map.

pop-up area: on new page display ui map.

Data step: move to data in your destination field name.

portal: show the zone.

cm: custom table

c1,ci:product table

f1: framework

# UI map:

1.html components and display the users.

2.ui map allows your implementation to create a look and feel that matches the customer business.

3.it directly connected with ormb components .

4.html is auto generator you don’t create it.

Map zone: display only information about a specific instance of a BO along with buttons that can initiate business process to change the BO.

Info zone: contains grid of information related to the object being displayed in portal.

* oraPageTitle: This class has logic that retrieves the label from field defined in oraMdLabel attribute and formats it using the standard title style.
* oraSelect: populates the option values.(populates a dropdown using several methods.)
* sourceField: Bo’s unique identifier
* Target path: location of source field in BO schema.
* Supress=true: base package algo ignore the element.
* Private =true: this private field are not accessible by any service.
* Label: is used to override the label used for display.
* Datatype: overrides how the element values is validated; it is also controls how to elements value displayed.
* Fk Ref: you want to fk’s context menu and add info string to be displayed then you can use fk ref.
* info string: we can display description of particular entity.
* Context menu: user interaction.
* External lookup: set of configuration
* Complete Html : normal html
* Html fragment: filtration purpose. inject particular html element into existing html element.
* Complete Xhtml: it is dynamic
* namespace: all the tags present in namespace.
* Group -in: When we invoke the two bs in one ui then we need to add group in ui script. And give null to it.

oraSelect lookup:

# Change Handler:

1.Validation.

2.add event-driven logic to entities.

3.base field is required but in business req their was others changes(some fields are mandatory) at that time

Methods:

1. Validation rule: A. To determine how to validate business entity at runtime.

B. you cant update ,add.

C. it must return an array of validation rule object.

# Algorithm

Algo: Reusable logic

Hard parameter: Abstract method present in your interface  
- Soft parameter: which value you want to take from the user

1.algo defines 2 places:

A. Database table -1algo type-defines entity, java class name(program name), program type(java, Cobol)

-2 algo-algo defines a n instance of algo type. algo code specified on the customized entity .the algo will be invoked based on the specification.

B. OUF:

Algorithm spot: 1.Destination point (which you want to call out the place in the system ).

2.An interface provides abstraction between base and the customization.

3.It also specify the schema defined for a plug-in-script.

4.algo spot reference an algo entity lookup value.

Algorithm components: The implementation of algo spot i.e. business components.

Factory class: create an instance of an algo component .

When to use factory class?

Note: \_Impl: implementation class that contains the logic.(the hand coded impl class)

\_gen: soft parameters generate by the artifacts generator.

Components interface method : it define required methods for the impl class as viewed for from the application.

Static factory class: it has new instance method to create an instance of algo components at runtime.

\*Algo parameter defines parameter value for the instance.

B. OUF-the framework require implementation class -the program that contains logic and various generate artifacts.

e.g:1) validating the phone no entered by user.

2)calculate payment .

batch: processing similar bulk data. perform bulk operation.

fix structure.

outer class: figure out the worker class.

thread pool:

e.g: bill generation.

DTO: simple performing operation then use dto.

hibernate is mapping the java classes.

map field: logical mapping where you can define datatypes based on behaviour. MAPPING WITH TABLE

md field: physical field include in mo.display

ui hints: generate the automated html code.

includemap:comman logic you can use.html,javascript logic reuse.in that case you can use includemap.

total 22 alogorithm create mo.