RAP

Behavior Definition

What is Behavior Definition?

Behavior Definition - Overview

- Helps to enable transactional capabilities of the BO entities
- New ABAP repository objects called Behavior Definition Language
- Looks like CDS Data Definition Language
- Can have one or more entity behaviour definition
- Root entities must be specified (Child entities are optional)
- Behavior implementation class can be defined for all BO Entity
- Formed by
    - Behaviour Characteristics
    - Behaviour Body

Behavior Characteristics - Overview

- Persistent table - dB table name should be specified
- Draft table - Draft dB table name should be specified
- Saving options - With additional save, With Unmanaged save
- Etag - Optimistic concurrency control
- Locking - Pessimistic concurrency control
- Authorization - Protects your data from unauthorized access
- Early numbering - Assign primary key fields values at Buffer update
- Late numbering - Assign primary key fields values before dB update

Behavior Body - Overview

- Field characteristics - Additional capability for fields
- Field numbering - Managed way of primary key assignment
- RAP BO operations - CRUD, CBA, RBA, Actions, Function etc.,
- Validations - checks consistency of BO instance data
- Determinations - Calculate or determine new values at runtime
- Type mapping - field mapping b/w dB table & CDS views
- Implementation grouping - Different implementation classes

Projection Behavior Definition

What is Projection Behavior Definition ?

Projection Behavior Definition

Behavior Characteristics:

- Persistent table
- Draft table
- Saving options
- Locking
- Etag
- Early numbering
- Late numbering

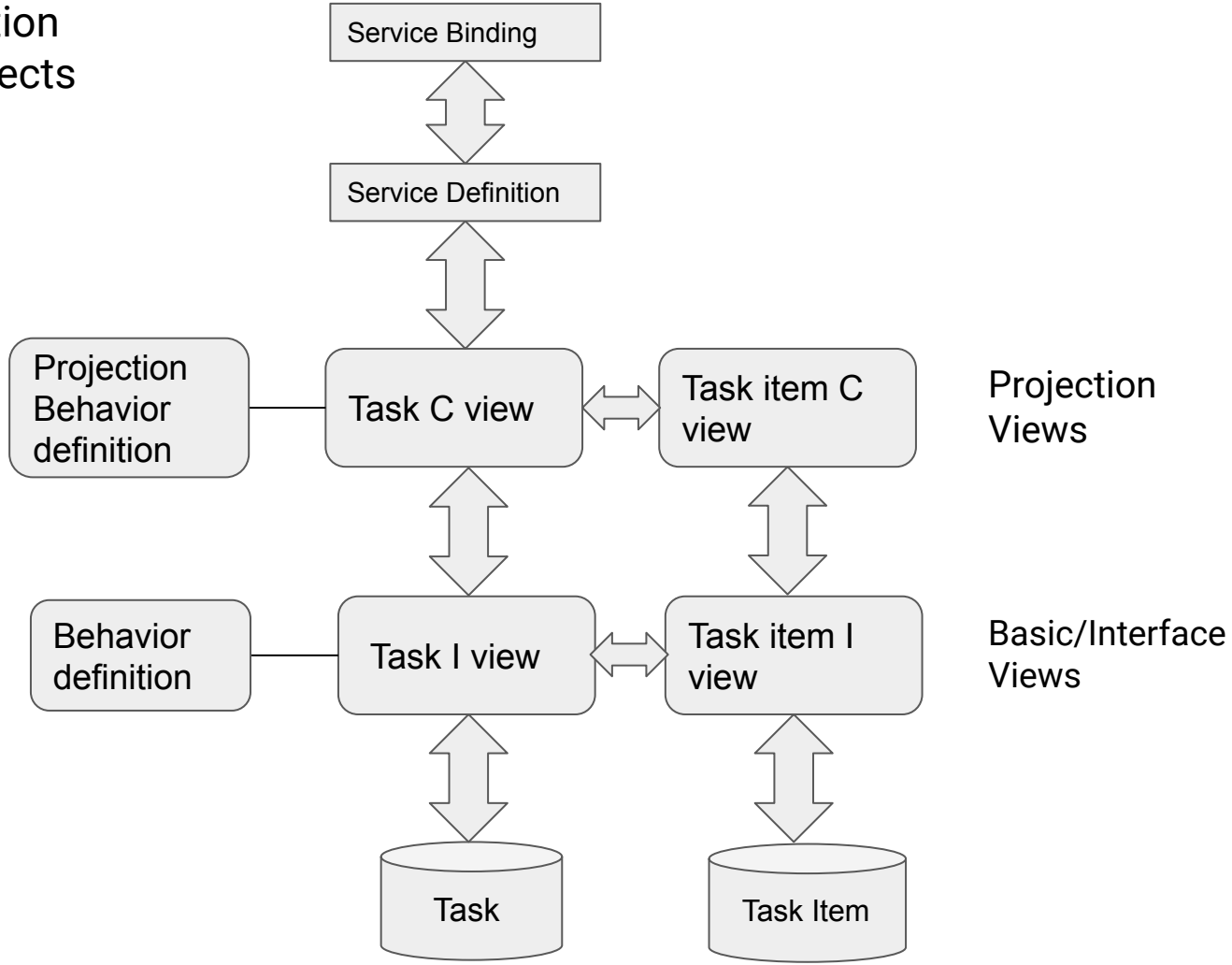
Behavior Body:

- Field characteristics
- Use
- Actions and functions
- Augment
- Type mapping

Service Definition & Binding

Business Services

Task Application
Technical objects



Managed BO

Managed Business Objects

BO Operations - Standard (CRUD)

- CRUD - Create, Read, Update, Delete called as Standard operations
- Read operation - Default operation for BDEF (not required to define)
- Managed - No implementation needed (Implementation needed for unmanaged)
- Projection BDEF - can reuse with the help of “use” keyword
- Create - new instance of BO entity
- Update - updates instance of BO entity
- Delete - deletes instance of BO entity
- Create by Association - creation of child entity instance should be via association

EML (Entity Manipulation Language)

What is EML?

EML - Entity Manipulation Language

- Standard API to access Business Objects Behaviours
- Direct way to consume BO (without Odata)
- READ Entities
  - READ
  - READ BY ASSOCIATION
- MODIFY Entities
  - CREATE
  - UPDATE
  - DELETE
  - CREATE BY ASSOCIATION
  - ACTION

Interaction Phase

- COMMIT Entities
- ROLLBACK Entities

Save phase

Read EML

What is Read EML ?

READ EML

```
READ ENTITIES OF BDEF
    ENTITY EntityName
     FIELDS (  field1, field2,.. ) | ALL FIELDS
     WITH CORRESPONDING #( keys )
    RESULT DATA(result)
    FAILED DATA(failed)
   REPORTED DATA(reported).
```

Returns requested instances data

Returns failed instances

Returns error details about failed instances

READ BY ASSOCIATION EML

```
READ ENTITIES OF BDEF
    ENTITY EntityName BY \_AssociationName
     FIELDS ( Field1, Field2..  ) | ALL FIELDS
     WITH CORRESPONDING #(  keys  )
RESULT DATA(travels)
LINK DATA(link)
FAILED DATA(failed)
REPORTED DATA(reported).
```


Returns Primary key values of source & target entity instance

Create EML

Create & Create By EML

CREATE & BY ASSOCIATION syntax

```
MODIFY ENTITIES OF BDEF
        ENTITY Header
          CREATE FROM itab_header
          CREATE BY \_Item FROM itab_item
MAPPED DATA(mapped)
FAILED DATA(failed)
REPORTED DATA(reported).
```


Returns Created Primary keys info mapped with Content id (CID)

CREATE & BY ASSOCIATION Preparation

Header:

Preparation of header content to be created

%CID - Content Id refers unique id before primary key creation

%DATA - Column values should be populated

%CONTROL - Set which column values are going to create

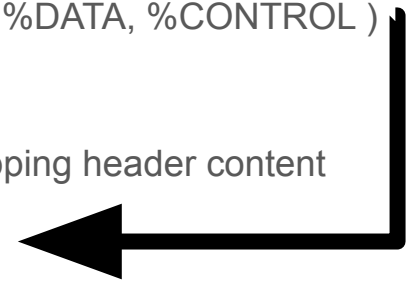
Item:

Preparation of item target to be created (%CID, %DATA, %CONTROL)

Preparation of item content to be created with

%CID_REF - Content Id reference helps for mapping header content

%TARGET - Associated item data to be created



Update EML

Update EML

UPDATE EML

```
MODIFY ENTITIES OF
BDEF
ENTITY Header
UPDATE FROM lt_header
FAILED DATA(failed)
REPORTED
DATA(reported).
```

```
MODIFY ENTITIES OF BDEF
ENTITY Item
UPDATE FROM lt_item
FAILED DATA(failed)
REPORTED
DATA(reported).
```

UPDATE EML - Combined

```
MODIFY ENTITIES OF BDEF
ENTITY Header
UPDATE FROM lt_header
ENTITY Item
UPDATE FROM lt_item
FAILED DATA(failed)
REPORTED DATA(reported).
```

Delete EML

Delete EML

DELETE EML

```
MODIFY ENTITIES OF BDEF
ENTITY Header
DELETE FROM VALUE
#( keys )
FAILED DATA(failed)
REPORTED
DATA(reported).
```

```
MODIFY ENTITIES OF BDEF
ENTITY Item
DELETE FROM VALUE
#( keys )
FAILED DATA(failed)
REPORTED
DATA(reported).
```

DELETE EML - Combined

```
MODIFY ENTITIES OF BDEF
ENTITY Header
DELETE FROM VALUE  #( keys )
ENTITY Item
DELETE FROM VALUE #( keys )
FAILED DATA(failed)
REPORTED
DATA(reported).
```

Validations

Validation Part1

Validation Part2

Validations

Syntax:

Validation ValName on save { create; update; delete; field field1, field2 etc., }

- Check consistency of the BO instance data based on trigger conditions
- Trigger conditions are
  - Modify operation (create, update, delete)
  - Modify fields (any BO columns)
- Validations are called at the *CheckBeforeSave* method by framework
- Validations order cannot be determined when more than 1
- EML modify statement should not be used in the implementation

Determinations

What is Determinations ?

Determination

Syntax:

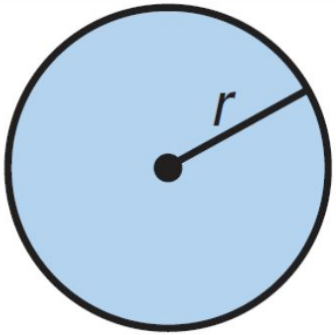
Determination DetName on save | modify { create; update; delete; field field1, field2 etc., }

- Modifies the BO instance data based on trigger conditions
- Trigger conditions are
  - Modify operation (create, update, delete)
  - Modify fields (any BO columns)
- Determination on Save - executes at the save sequence phase
- Determination on Modify - executes at the Interaction phase
- Determinations order cannot be determined when more than 1
- Determinations results should be always same if executed multiple times

radius r

$$C = 2\pi r$$

$$A = \pi r^2$$



From: May 15, 2016

To: Enter Date

May 2016

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
18	1	2	3	4	5	6	7
19	8	9	10	11	12	13	14
20	15	16	17	18	19	20	21
21	22	23	24	25	26	27	28
22	29	30	31	1	2	3	4

OK Cancel

Actions

What is Action in RAP ?

Action with Parameters

Actions

- Non Standard - Predefined set of operations - Class/Method
- Where we can consume
  - UI via services (OData)
  - From another actions via EML
- Internal - Consumed within BO
- Static - Related to the complete entity
- Repeatable - Can execute multiple time for same instance
- Factory - Used to create on entity by defaulting the values

Actions (Cont..)

Input Parameters

Output Parameters

Behaviour Characteristics

Numbering

Feature Control

Authorization

ETag

Draft handling

Feature Controls

Instance Feature Control

Static Feature Control

Global Feature Control

Numbering in RAP

Numbering

- External - What is External Early Numbering ?
- Internal
  - Early
    - Managed - What is Managed Internal Early Numbering ?
    - Unmanaged - What is Unmanaged Internal Early Numbering ?
  - Late
    - Unmanaged - What is Late Numbering ?
- Uniqueness Check for Primary keys

Concurrency Control

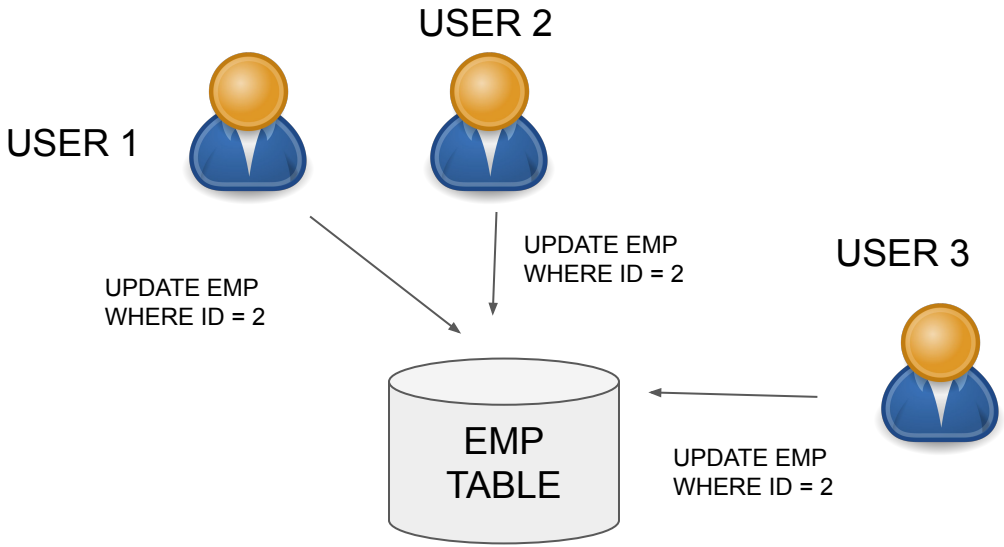
What is Concurrency Control in RAP ?

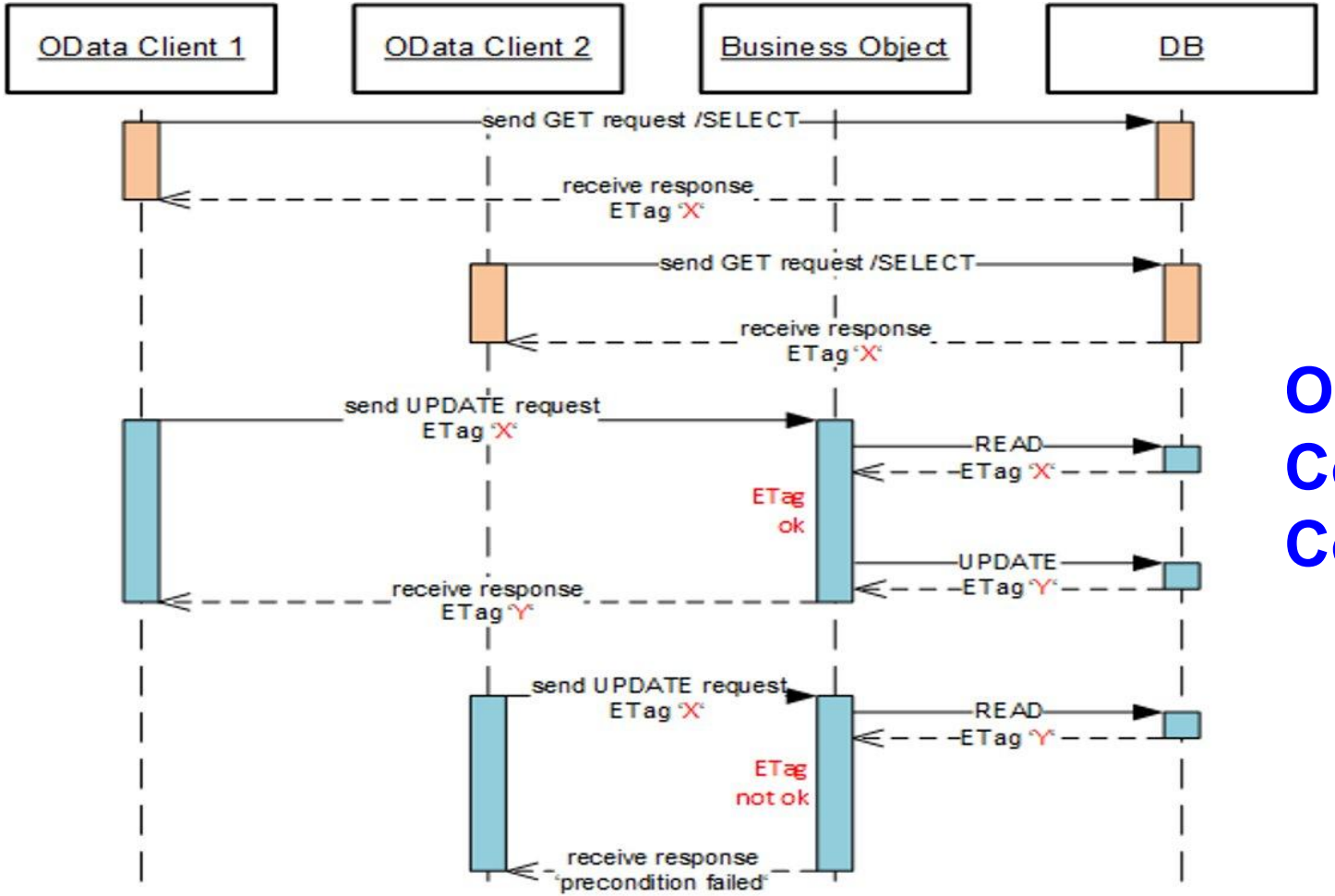
How to Define Etag in RAP ?

How to Define Locking ?

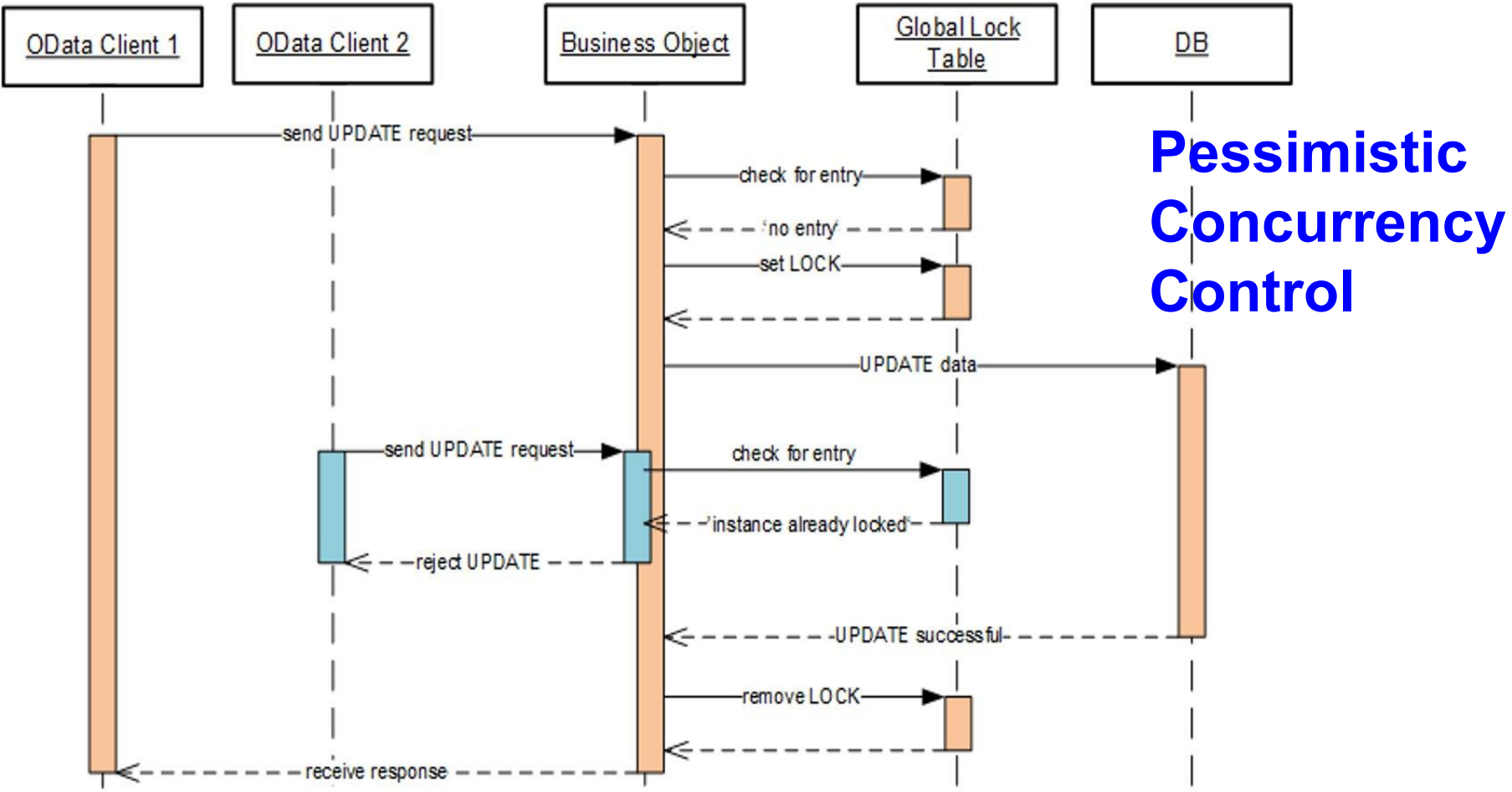
Concurrency Control

- Controls simultaneous access to the DB of different users on same resource
- If multiple simultaneous requests comes, then it should update only one and rejects other requests





Optimistic Concurrency Control



Pessimistic Concurrency Control

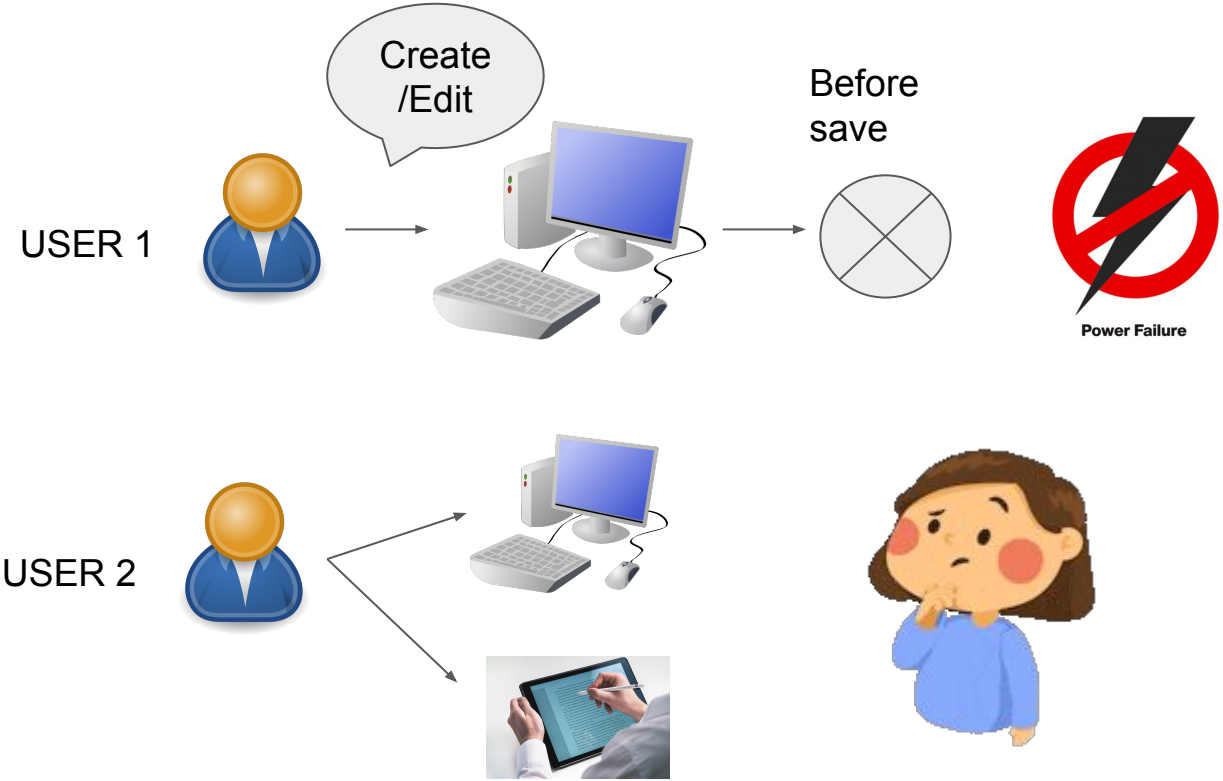
Draft Handling

Overview of Draft Handling

What is Draft Design time & Runtime ?

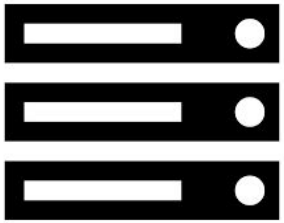
How to Implement Draft handling in RAP ?

Problem Statement



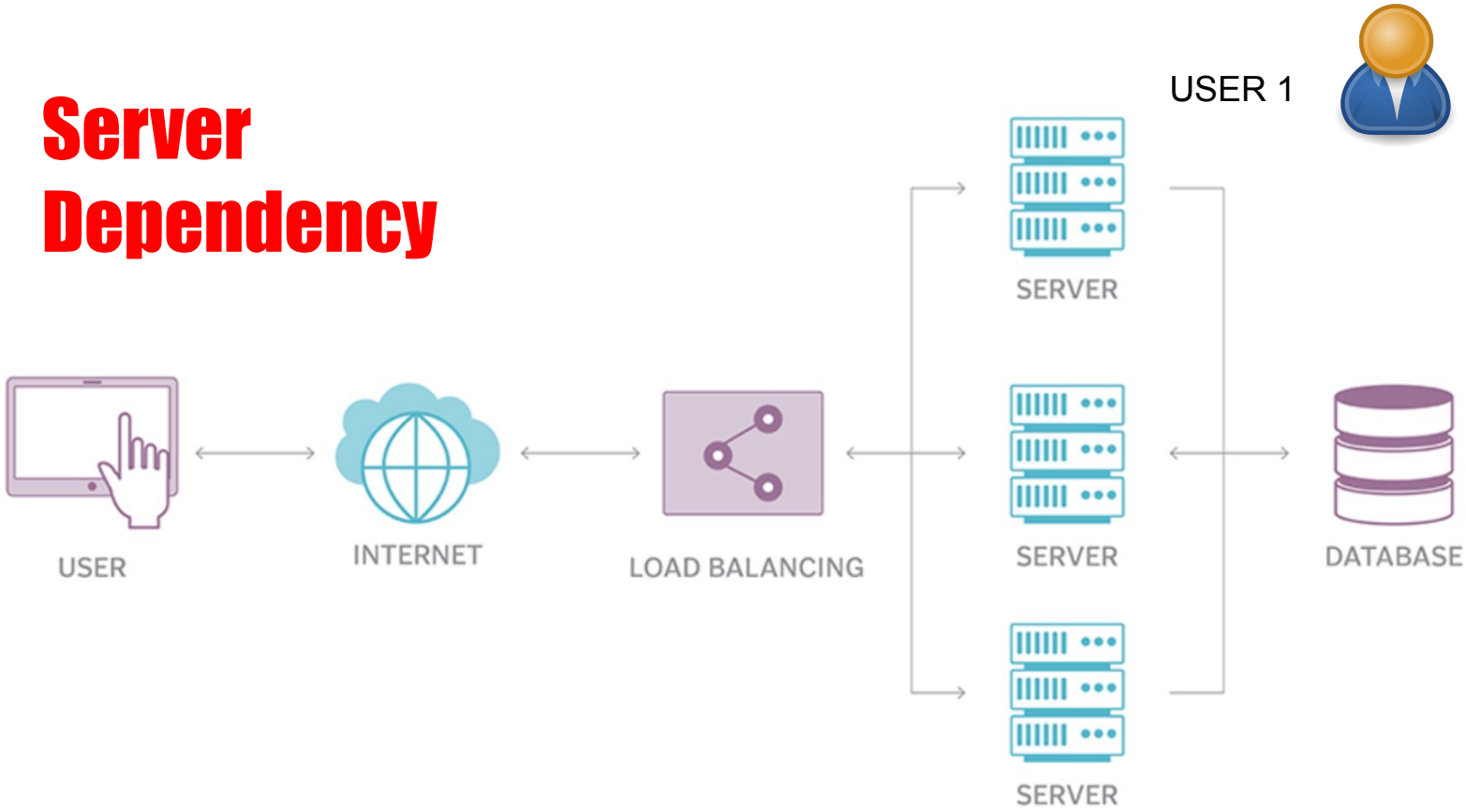
Thinking about Solution

Stateful communication

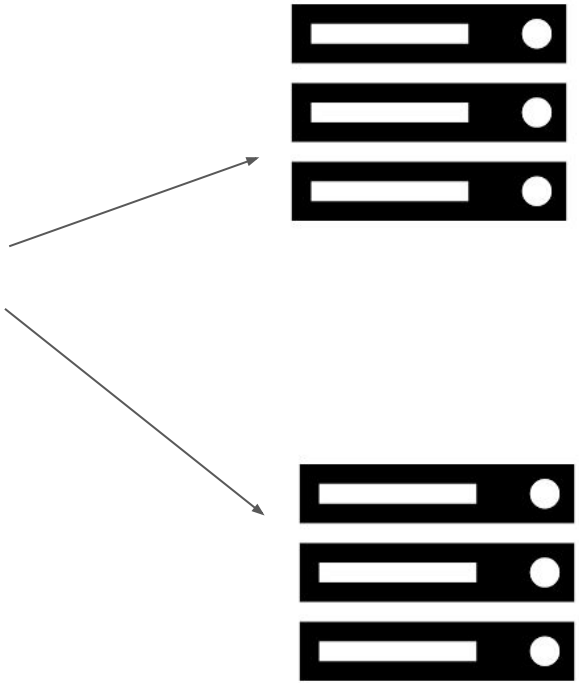


Server session store details of the each request

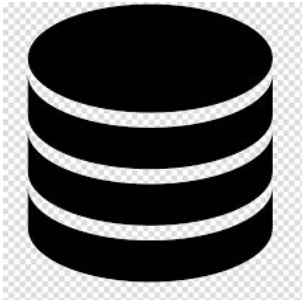
Server Dependency



Solution - Stateless communication with Stateful capability



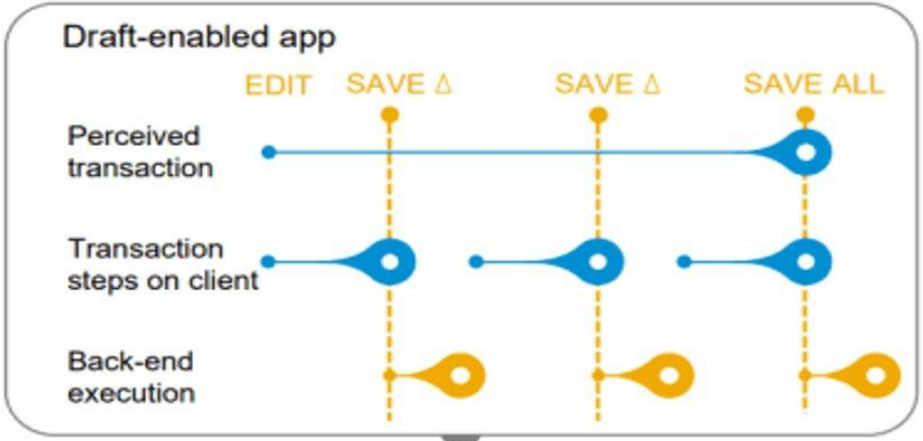
Stores data in the Draft DB table as temporary state



Draft – Stateful applications with stateless communication protocol

IMPACT for end user

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

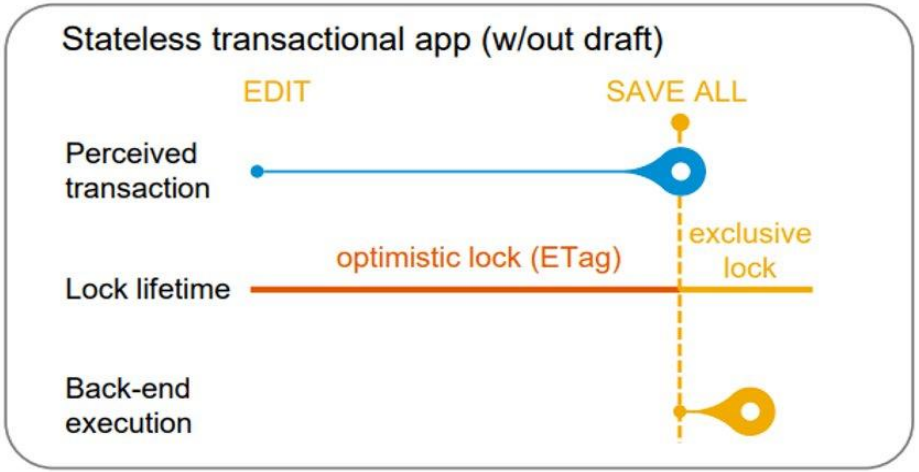


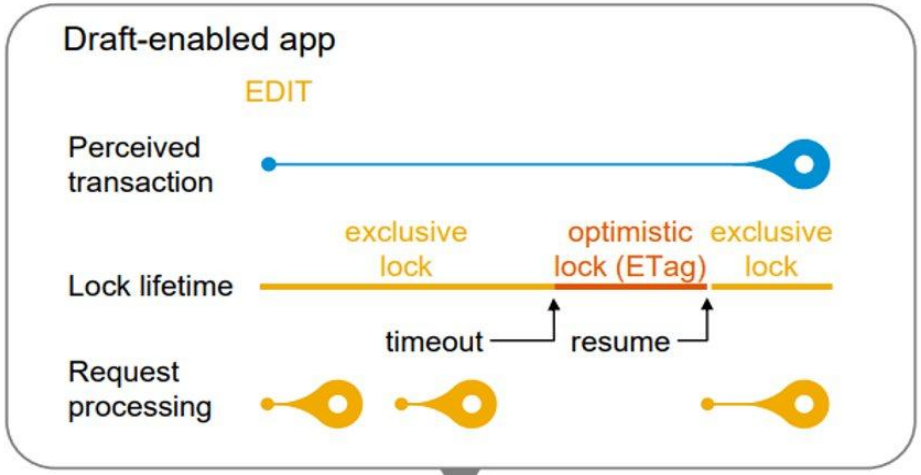
IMPACT for end user

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





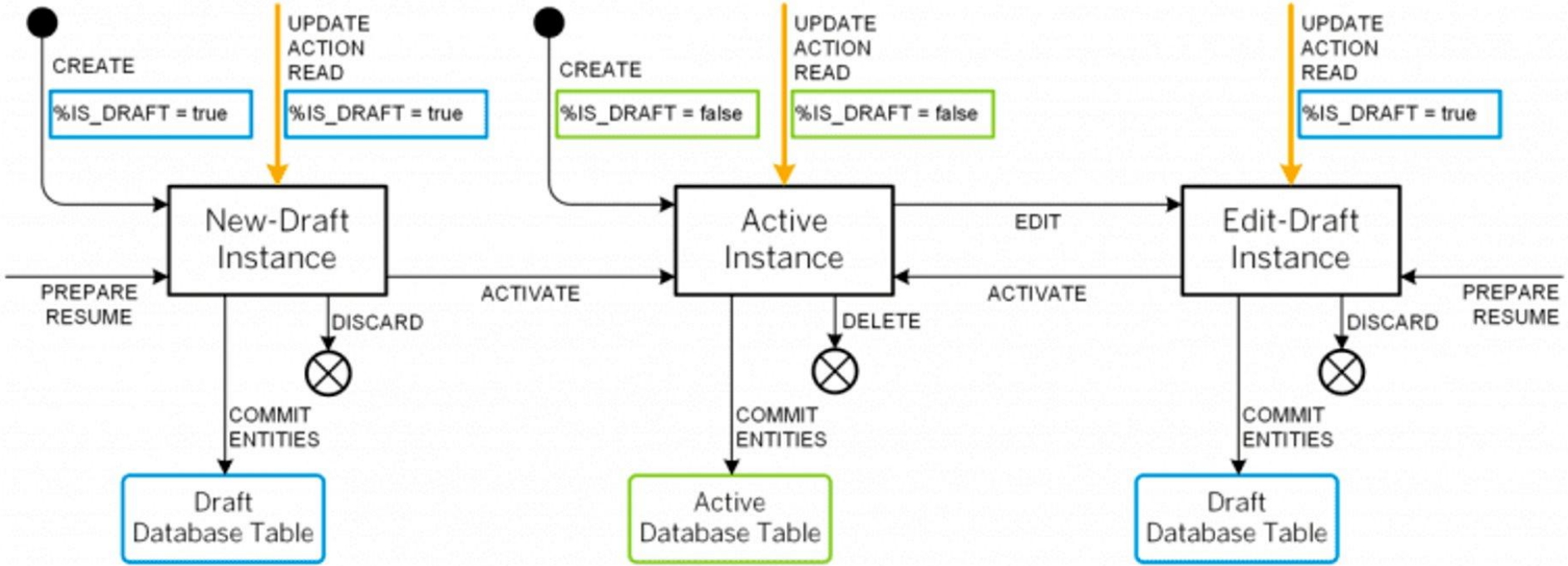
End user perceived transaction distributed over a long period, so locks can't be held all the time

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



Life Cycle of Draft and Active Data

How Draft works?

Stateless communication - No fixed backend session

Stateful communication - Fixed backend session

Due to scalability & elasticity requirements in the Cloud model, we have to use stateless communication

Side Effects in RAP

What is Side Effects ?

RAP Generator

What is RAP Generator ?

How to Deploy RAP Application ?

Step by Step guide