

# OData V4 Model

## Improvements since UI5Con 2017

Thomas Chadzelek, Patric Ksinsik, Mathias Uhlmann, SAP SE  
June 28, 2019

# UI5con

learn.explore.connect.

PUBLIC



# Introduction

At UI5Con 2017 Patric and Thomas have introduced the OData V4 Model and demonstrated how it can be used.

The example used has been translated into the [OData V4 Tutorial](#) in the Demokit.

The focus of development since then has been on supporting the Draft use case.

- Draft Handling is described in <https://experience.sap.com/fiori-design-web/draft-handling/>.
- Editing of a business document is done on a draft version that is created when starting to edit. The draft version becomes the active version when explicitly saving the document.
- All changes are implicitly saved into the draft version immediately to prevent data loss.
- Warning and error messages regarding the current state of the document are created in the backend.

# Agenda

## Introduction

### Draft – Improvements for using the OData V4 Model in Draft applications

- earlyRequests (X-CSRF-Token, \$metadata)
- \$batch
- Side Effects (\$\$patchWithoutSideEffects, requestSideEffects)
- Operations (advertisement, bound, overloading, return value context, sync)

## Create

## Suspend / Resume

## Unit / Currency – Quantities / Amounts formatted according to backend customizing

## Server messages

## Conclusion / Outlook

# Draft: It's all about \$auto

```
"models" : {  
  "" : {  
    "dataSource" : "default",  
    "preload" : true,  
    "settings" : {  
      "autoExpandSelect" : true,  
      "earlyRequests" : true,  
      // "groupId" : "$auto",  
      // "updateGroupId" : "$auto",  
      "operationMode" : "Server",  
      "synchronizationMode" : "None"  
    }  
  }  
},
```

**earlyRequests**

# earlyRequests

Early requests for metadata and security token → [Performance Aspects](#) (since 1.54)

- Model parameter (default still **false**) which requests as early as possible:
  - \$metadata: service's root \$metadata document and annotation files
  - X-CSRF-Token: security token

```
"models" : {  
  "" : {  
    "dataSource" : "default",  
    "preload" : true,  
    "settings" : {  
      "autoExpandSelect" : true,  
      "earlyRequests": true,  
      // "groupId" : "$auto",  
      // "updateGroupId" : "$auto",  
      "operationMode" : "Server",  
      "synchronizationMode" : "None"  
    }  
  }  
},
```

**\$batch**

## Several \$auto groups → Batch Control to improve performance

### SubmitMode.Auto

(since 1.52)

Allows separate requests for fast and slow data

Controller or manifest.json

```
"models" : {
  "" : {
    "dataSource" : "default",
    "settings" : {
      "operationMode" : "Server",
      "synchronizationMode" : "None",
      "groupProperties" : {
        "fastGroup" : {"submit" : "Auto"},
        "slowGroup" : {"submit" : "Auto"}
      }
    }
  }
}
```

\$auto.\* (since 1.58): XML views

```
<Text text="{path : '/Me/Name', parameters : { '$$groupId' : '$auto.fast' }}" />
<Table rows="{path : '/BusinessPartners', parameters : { '$$groupId' : '$auto.slow' }}">
```



## Serialization of PATCHes (since 1.60)

No \$batch sent for group “foo” while request for the same group is still in flight

- Makes sure that PATCHes are sent **in sequence** and with up-to-date Etag
- User input wins over server response
- (even non-adjacent) PATCH requests for the same entity are merged into a single request
- [submitBatch](#) adds new change set → application still has control over isolation

# Automatic retry

of PATCH (update) and POST (create)

The OData V4 model automatically repeats failed updates (PATCH) and creates (POST)

- SubmitMode.API: repeat with next call of submitBatch
- SubmitMode.Auto (or Direct): repeat with next update for same entity or via [submitBatch](#) (since 1.67)

Support ETag header for PATCH with 204 No Content (since 1.66)

Events ([ODataContextBinding](#), [ODataListBinding](#))

- patchSent, patchCompleted (since 1.60):  
show DraftIndicator
- createSent, createCompleted (since 1.66):  
show BusyIndicator or otherwise lock UI



**Side Effects**

# Side Effects

[sap.ui.model.odata.v4.Context#requestSideEffects](#) (since 1.61ff.)

PATCHing one field might have a side effect on another field:

- changing a sales order line item's quantity has an effect on its gross amount
- application logic in backend computes these side effects, but client needs to request them
- V4 model sends efficient requests: only for (navigation) properties which...
  - may be affected: parameter to API call, typically based on annotations
  - are shown on the UI: as determined by auto-\$expand/\$select and table's visible range etc.

Use control event [validateFieldGroup](#) to request side effects as soon as field group has changed  
→ PATCH and GET inside same \$batch

```
oContext.requestSideEffects([{$PropertyPath : "GrossAmount"}]);
```

[\\$\\$patchWithoutSideEffects](#): binding parameter to ignore side effects in PATCH responses

**Actions, Functions**

## (Bound) Actions & Functions

Draft orchestration needs bound actions like Create, Edit, Prepare, Activate etc.

- Binding against operation advertisement (since 1.54)
  - `<Button enabled="{= !!%{# name.space.ConfirmAction} }"`  
`text="{path: '#name.space.ConfirmAction/title', type: 'sap.ui.model.odata.type.String'}"/>`
- Support bound actions and functions (1.54), even bound on collection (1.62; for example, Create):
  - `oListBinding = oModel.bindList("/Artists");`
  - `oHeaderContext = oListBinding.getHeaderContext();`
  - `oModel.bindContext("name.space.Create(...)", oHeaderContext).execute();`
- Bound action overloading (since 1.56; for example, same name „Create“ for many entity sets)
- Annotations for action overloads (since 1.66)

```
<Annotations Target="MySchema.Create(Collection(MySchema.ArtistsType))/Countryoforigin">  
  <Annotation Term="com.sap.vocabularies.Common.v1.Label" String="Country of Origin"/>  
</Annotations>
```

  - `oModel.getMetaModel().getObject(  
 "/Artists/name.space.Create/Countryoforigin@com.sap.vocabularies.Common.v1.Label")`

# Access Operation Results

“Return value context” (since 1.56): Edit action returns inactive version of its binding parameter

- [v4.0DataContextBinding#execute](#)’s promise resolves with a `v4.Context`
- Use binding parameter [\\$\\$inheritExpandSelect](#)
- Messages relate to this context only! [Read the fine print...](#)

Binding parameter is updated if bound action returns same entity (1.60; for example, Prepare)

**Create**



# Enhancements for create: Automatic refresh of created entity

An entity created via [ODataListBinding#create](#) is refreshed after create to also read navigation properties not sent in the POST response. (since 1.54)

- Refresh can be skipped via flag `bSkipRefresh` in [ODataListBinding#create](#). (since 1.60)
- Refresh is also supported for create on relative list bindings (since 1.67)

A single entity in a list can be refreshed separately via [Context#refresh](#) (since 1.54)

Requests for creation and refresh on creation of an entity:

Create: POST SalesOrderList

Refresh: GET SalesOrderList('0500001389')

?\$select=BuyerID,ChangedAt,CurrencyCode,GrossAmount,LifecycleStatus,LifecycleStatusDesc,Messages>Note,SalesOrderID  
&\$expand=SO\_2\_BP(\$select=BusinessPartnerID,CompanyName)

Sales Order ID	Buyer Name	Gross Amount	Currency	Note	Life Cycle Status	Changed At
0500001389	SAP	0,00	EUR	foo	Neu	13.06.2019, 16:28:04

# Enhancements for create: Defaults in initial data

Properties not supplied in the initial data for [ODataListBinding#create](#) are shown with defaults as defined in the service metadata on the UI; these properties are however not sent to the server (since 1.60)

```
// creation prefills initial data with defaults if not supplied
oContext = this.byId("SalesOrderList").getBinding("items").create({
  "BuyerID" : "0100000000",
  "LifecycleStatus" : "N"
});
```

```
// service $metadata: default "EUR" for currency
<EntityType Name="SalesOrder">
  ...
  <Property Name="CurrencyCode" Type="Edm.String"
    DefaultValue="EUR"/>
```

Sales Order ID	Buyer Name	Gross Amount	Currency	Note	Life Cycle Status	Changed At
			EUR			
0500000000	SAD	25.867.03	EUR	EDM DC: SOL	New	08.11.2015,

# Enhancements for create: Create multiple entities without list binding refresh

Multiple entities can be created without refresh in-between the create calls (since 1.65)

- Created entities persisted on the server are excluded from read-requests to avoid "duplicates" on the UI (since 1.65)

A newly created entity can be shown at the end of the list with parameter `bAtEnd` of [ODataListBinding#create](#) (since 1.66)

*[SalesOrders sample](#): Two new sales orders, one is persisted, the other is transient*

Sales Order ID	Buyer Name	Gross Amount	Currency	Note	Life Cycle Status	Changed At
			EUR	Second new sales order		
0500001389	SAP	0,00	EUR	First new sales order	Neu	13.06.2019, 16:28:04
0500000001	Talpa	24.841,25	EUR	EPM DG: SPO ID 05000000001 Deliv	Neu	12.06.2019, 15:56:01
0500000005	Telecomunicaciones Star	101.299,22	EUR	EPM DG: SO ID 05000000005 D...	Neu	29.05.2019, 00:00:00

**Suspend Resume**

# suspend and resume APIs

Pair of APIs on absolute bindings; "suspended" can also be declared for a binding (since 1.54).

- suspend lets the binding "sleep": binding sends no data service requests or events
- resume "wakes the binding up": binding sends read request which considers changes to the binding while it was suspended

See also [documentation on suspend and resume](#).

Use case: Only send data service request for a suggestion list when the dialog is shown

```
Main.view.xml (from UI5 demokit, Samples, OData V4, Sales Orders)
<Dialog title="Create New Sales Order">
  ...
  <Label text="Buyer ID"/>
  <Input id="BuyerID::new" showSuggestion="true"
    suggestionItems="{path : '/BusinessPartnerList', suspended : true}"
    value="{BuyerID}">
    <suggestionItems>
      <core:ListItem text="{BusinessPartnerID}"/>
    </suggestionItems>
  </Input>
  ...
</Dialog>
```

```
Main.controller.js
onCreateSalesOrder : function () {
  var oBPListBinding = this.byId("BuyerID::new")
    .getBinding("suggestionItems");

  if (oBPListBinding.isSuspended()) {
    oBPListBinding.resume();
  }
  oCreateSalesOrderDialog.open();
}
```

## suspend and resume: \$\$ownRequest binding parameter

Enforce an own request for a binding with binding parameter `$$ownRequest` (since 1.56)

Use case: Ensure separate requests for master-detail scenarios with `auto-$expand/$select`.

- Without `$$ownRequest` on a detail binding, resuming a suspended master binding may include a detail binding via `$expand` in the master list request.

Of course, `$$ownRequest` may also be used without suspend/resume to enforce a separate request.

```
<SimpleForm id="master" binding="{/TEAMS('TEAM_01')}">
  <Text text="{MEMBER_COUNT}" />
  <Table id="detail" items="{path : 'TEAM_2_EMPLOYEES', parameters
: { $$ownRequest : true }}">
    <columns><Column/><Column/></columns>
    <ColumnListItem>
      <Text text="{AGE}" />
      <Text text="{Name}" />
    </ColumnListItem>
  </Table>
</SimpleForm>
```

Separate data requests for master and detail:

Master: GET TEAMS('TEAM\_01')?\$select=MEMBER\_COUNT,Team\_Id

Detail: GET TEAMS('TEAM\_01')/TEAM\_2\_EMPLOYEES?\$select=AGE,ID,Name

# suspend and resume: Combine requests

APIs like sort, filter, changeParameters are allowed on a suspended binding (since 1.62)

Use case: Combine multiple request triggering APIs in one request, e.g. "sort" and "filter" on list binding

```
oBPListBinding.suspend();  
  
oBPListBinding.filter(new Filter("BusinessPartnerRole",  
FilterOperator.EQ, "01"));  
oBPListBinding.sort(new Sorter("CompanyName"));  
  
oBPListBinding.resume();
```

One data request containing \$filter and \$orderby:

```
GET BusinessPartnerList  
?$orderby=CompanyName  
&$filter=BusinessPartnerRole%20eq%20'01'  
&$select=BusinessPartnerID,CompanyName&$skip=0&$top=100
```

**Unit Currency**



# Unit and currency: Representation in UI5 and OData, Use cases

Out of the box, UI5 supports [units](#) and [currencies](#) from [Unicode CLDR](#).

An OData V4 Service may use specific unit and currency customizing via CodeList annotations. This customizing mainly contains the list of allowed units and currencies and the number of decimals to display a value.

See [Currencies and Units](#) in OData V4 Model documentation for details on required annotations.

Typical uses cases for service specific customizing

- Service specific set of units or currencies which are not identical to CLDR predefinition
- Special case: Multiple currencies for the same ISO currency with a different number of decimals  
Sample: "EUR" (standard), "EUR3" (for gas stations) and "EUR5" (for financial calculations)

Sample response for currency customizing request:

```
{
  "value" : [{
    "CurrencyCode" : "ADP",
    "ISOCurrencyCode" : "ADP",
    "Text" : "Peseta",
    "DecimalPlaces" : 0
  }, ...
}
```

# Unit and currency: Format and parse based on customizing (since 1.63)

Composite types `sap.ui.model.odata.types.Currency|Unit` format and parse composite bindings with parts for `value`, `unit or currency` and `customizing` (based on `sap.ui.core.format.NumberFormat`)

```
<Input value="{mode:'TwoWay', parts:['Price', 'CurrencyCode', {mode:'OneTime', path:'/##@@requestCurrencyCodes', targetType:'any'}]}, type:'sap.ui.model.odata.type.Currency'}/>
```

Formatting and parsing of value and currency or unit consider customizing.

- Parts `["1.2", "EUR", {"EUR" : {"ISOCODE" : "EUR", "DecimalPlaces" : 2}, ...}]` are formatted to `"1.20 EUR"` or `"1.20 €"`
- Input value `"1.23 €"` or `"€ 1.23"` is parsed into value and currency parts `[1.23, "EUR"]`

Possibility to specify a property binding for unit or currency customizing via paths

`' /##@@requestCurrencyCodes ' resp. ' /##@@requestUnitsOfMeasure ':`

- `##` path prefix branches to meta model
- `@@` represents a “computed annotation” executing `ODataMetaModel#requestCurrencyCodes` resp. `ODataMetaModel#requestUnitsOfMeasure` which load the customizing

## Unit and currency: Usability features










Parse keeps previous currency or unit if only a numeric value is entered (since 1.63)

Parse validates number of decimal places (since 1.64)

Parse empty string to value 0 and previous currency or unit (since 1.66)

**Server messages**

# Demo – part I

		Refresh all		 3		
		All	 1	 1	 1	
		 System Maintenance on 20190614 at 12h until 15h UTC				
	Changed At	 SAP UI5con learn.explore.connect				
	Jun 14, 2019, 2	 Enter a minimum quantity of 2				
	Jun 12, 2019, 3					
	May 29, 2019, 1					
	Jun 4, 2019, 12					
	May 29, 2019, 1					

# Server messages

The business logic is typically implemented in the backend.

- For example, when processing a Sales Order ATP (availability) check and pricing are done in the backend. Messages informing about the outcome may need to be shown to the user.

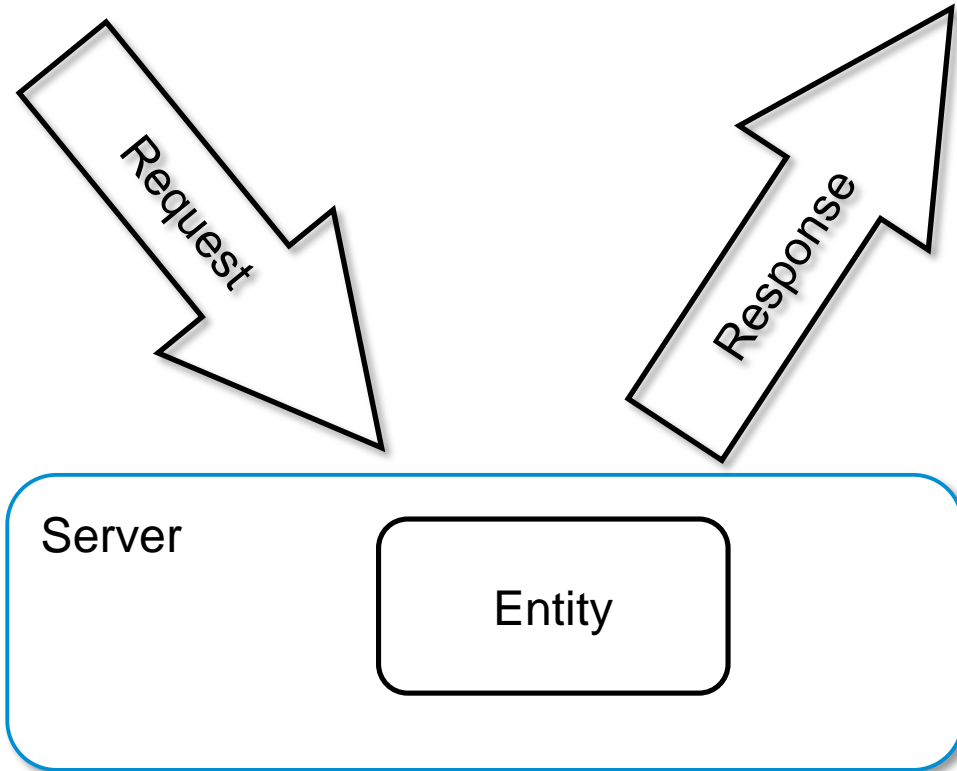
The handling of server messages in the OData V4 Model was implemented in and shipped with 1.58 and 1.60.

- The messages contained in server responses are parsed and reported to the MessageModel.
- Some messages are deleted by the OData V4 Model when they are no longer relevant.
- Some small additions are still missing.
- The feature is documented in [chapter "Server Messages in OData V4 Model,,](#) of the UI5 documentation.
- Messages are also demoed in [Sales Orders sample application](#).

To be done by the application:

- Displaying the server messages in the UI.
- Providing the messages in the server responses.

# The server side



The response may contain messages related to the

- state of the entity in the server (persistence) → **state messages**
- request → **transition messages**.

A state message can be retrieved with a GET request until a state change resolves the message.

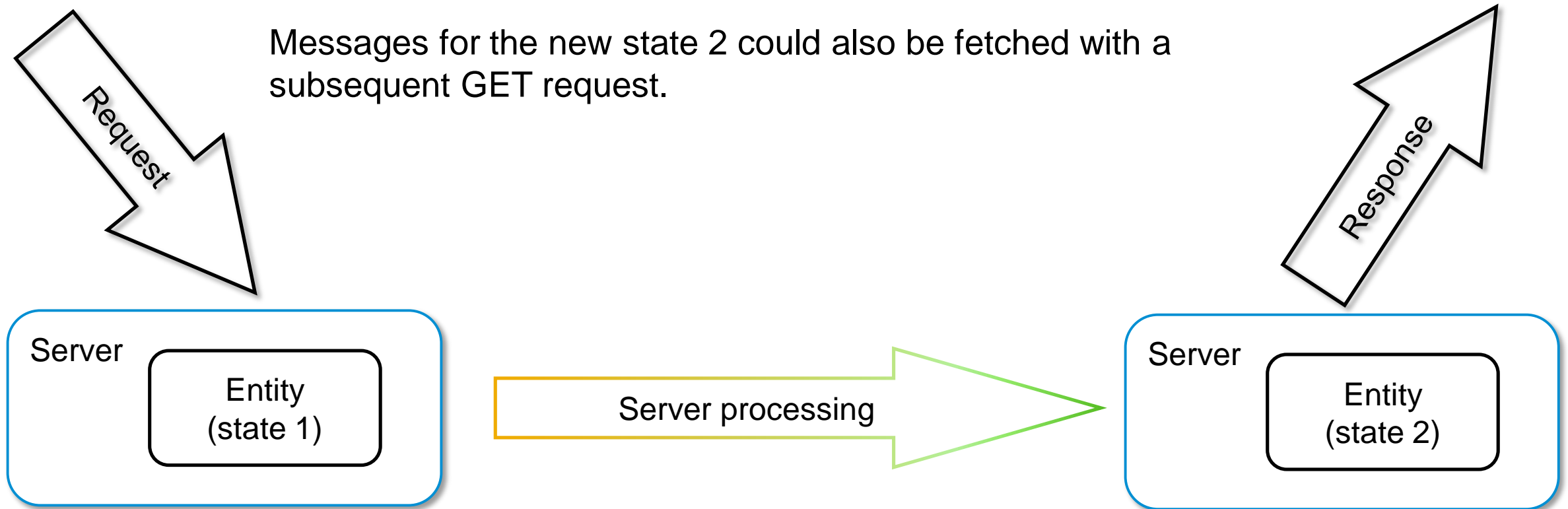
Transition messages are relevant only for the current request.

# The server side – successful PATCH

Response with

- Messages for the new state 2
- Messages related to the transition from state 1 to state 2

Messages for the new state 2 could also be fetched with a subsequent GET request.

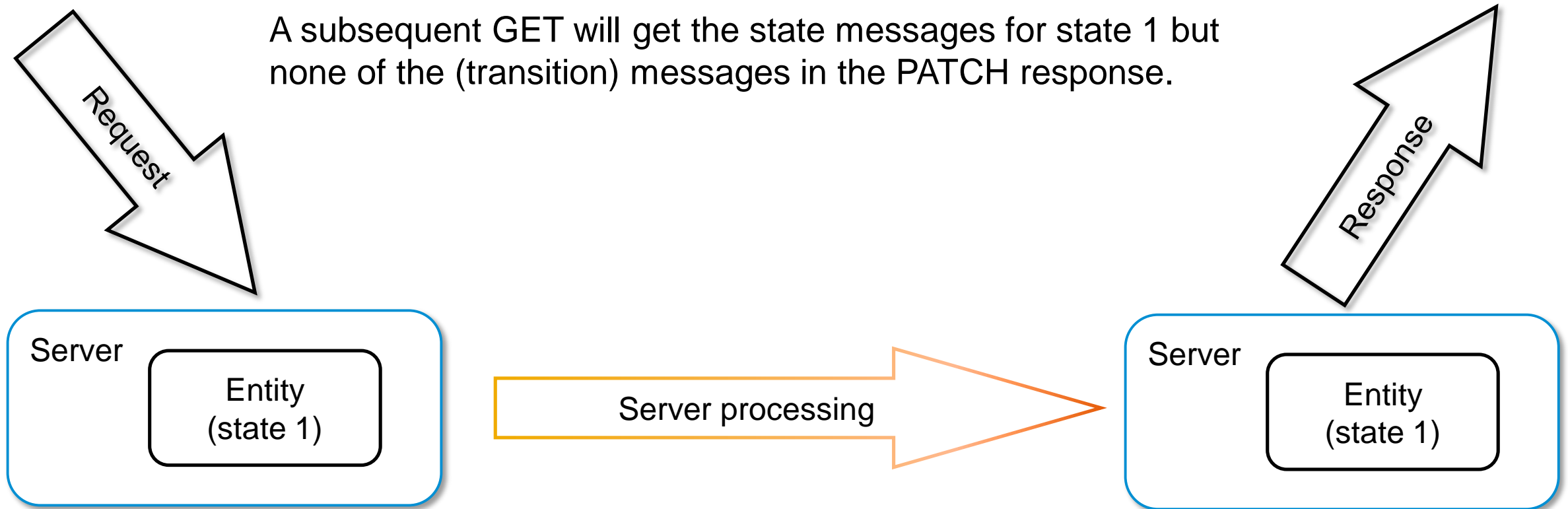




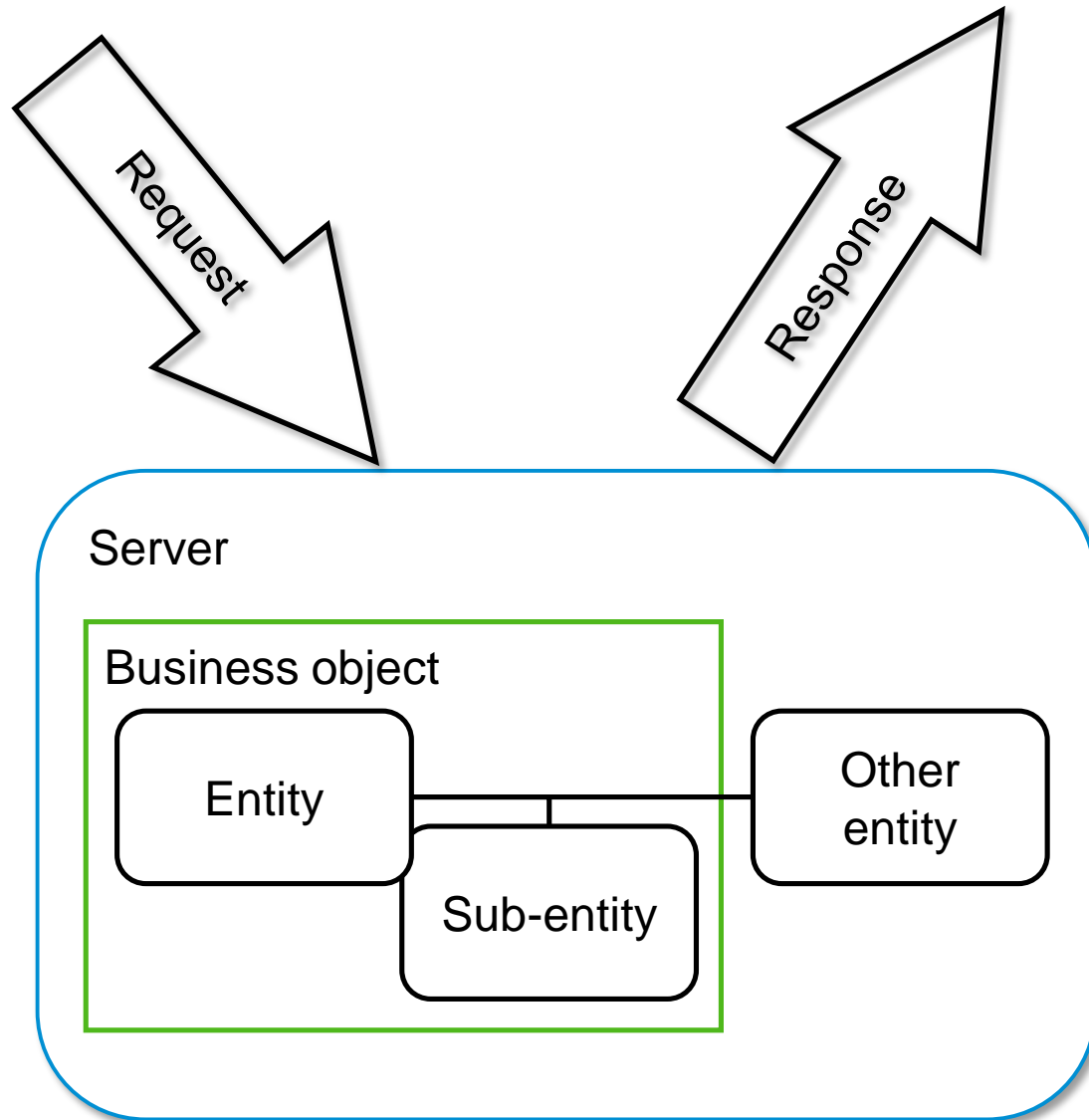
## The server side – failing PATCH

Error response with messages related to the failed transition from state 1 to state 2. These messages may refer to the entity (bound messages) or not (unbound messages).

A subsequent GET will get the state messages for state 1 but none of the (transition) messages in the PATCH response.



# The server side – successful GET



Request Entity

Response with

- state messages for Entity
- state messages with Sub-Entity

The response contains all state messages for the entity and its sub-entities within the same Business Object (if state messages are requested/returned).

# Transport mechanism

## Successful Requests (HTTP 2xx)

### Unbound Messages → HTTP Header

```
sap-messages:[
  {
    "code":"SYS/42",
    "message":"System will be down for maintenance next weekend.",
    "numericSeverity":2,
    "longtextUrl": "...",
  },
  ...
]
```

### Bound Messages → HTTP Body

```
{
  "OrderNo": 4711,
  ...,
  "myMessages": {
    {
      "code": "UF1",
      "message": "Delivery date has to be in the future",
      "target": "to_Lines(1)/DeliveryDate",
      "transition": false,
      "numericSeverity": 4,
      "longtextUrl": "...",
    },
    ...
  ]
}
```

property referenced via  
@com.sap.vocabularies.Common.v1.Messages

Bound messages may need to be explicitly  
requested via \$select

## Not Successful Requests (HTTP 4xx or 5xx)

### → HTTP Body using [OData V4 error response format](#).

```
{
  "error": {
    "code": "UF0",
    "message": "Unsupported functionality",
    "target": "",
    "details": [
      {
        "code": "UF1",
        "message": "$search query option not supported",
        "target": "",
        "@Common.numericSeverity": 4,
        "@Common.longtextUrl": "...",
      }
    ]
  }
}
```

Only transition messages in error responses as the  
state should not change with a failed request

# The client side

## The OData V4 Model

- reports the server messages to the client-side MessageModel and
- takes care to delete all outdated state messages.
- Transition messages are reported as persistent messages to the MessageModel. Messages with `persistent:true` are not deleted by the V4 model.

## The application has to take care of

- displaying the messages accordingly and
- removing transition messages when they are no longer relevant.

# Displaying messages

Bind the message model, e.g. in a message popover.

Attach to the change event to get called whenever messages are added or deleted. E.g.

- Determine additional information, e.g. whether to highlight, for the button which opens the message popover.
- For certain messages, display them instantly by opening the popover, a popup, ...

```
common/Controller.js (used in UI5 demokit, Samples, OData V4, Sales Orders)
initMessagePopover : function (sOpenButtonId) {
    this.messagePopover = new MessagePopover({
        items : {
            path : "messages>/",
            template : new MessageItem({
                description : "{messages>description}",
                longtextUrl : "{messages>descriptionUrl}",
                title : "{messages>message}",
                type : "{messages>type}"
            })
        }
    });
    ...
    this.messagePopover.setModel(sap.ui.getCore().getMessageManager().getMessageModel(), "messages");
    this.messagePopover.getBinding("items").attachChange(handleMessagesChange, this);
    ...
}
...
onToggleMessagePopover: function () {
    this.messagePopover.toggle(this.byId(this.messagePopoverButtonId));
}
```

# Lifecycle of transition messages

Transition messages have to be removed by the application when they are no longer required.

For bound (transition) messages, the respective field is highlighted as long as the message exists.

Proposal:

- Delete unbound transition messages after the user has seen them.
- Delete bound transition messages when the respective activity is repeated. E.g., messages of a failed Save (`oModel.submitBatch("updateGroup")`) are deleted with the next Save attempt.

```
common/Controller.js (used in UI5 demokit, Samples, OData V4, Sales Orders)
initMessagePopover : function (sOpenButtonId) {
    ...
    this.messagePopover.attachAfterClose(function (oEvent) {
        var oMessageManager = sap.ui.getCore().getMessageManager(),
            aMessages;

        // remove all bound messages which have to be handled by the application
        aMessages = oMessageManager.getMessageModel().getData().filter(function (oMessage) {
            return oMessage.persistent;
        });
        oMessageManager.removeMessages(aMessages);
    });
}
```

# Demo – part II

## OData Requests

[Clear](#)

[200] HEAD

http://localhost:8080/proxy/https/ldciqal.wdf.sap.corp:44326/sap/opu/odata4/sap/zui5\_testv4/default/sample/0002/?custom-option=value&sap-client=120

2019-06-14T14:44:07.087Z: 79.36299999892071 ms

[200] HEAD

http://localhost:8080/proxy/https/ldciqal.wdf.sap.corp:44326/sap/opu/odata4/sap/zui5\_testv4/default/sample/0002/?custom-option=value&sap-client=120

2019-06-14T14:44:12.396Z: 53.68899999848509 ms

[200] POST

http://localhost:8080/proxy/https/ldciqal.wdf.sap.corp:44326/sap/opu/odata4/sap/zui5\_testv4/default/sample/0002/\$batch?custom-option=value&sap-client=120

2019-06-14T14:44:12.472Z: 67.00799999998708 ms

[200] GET ProductList('HT-1000')/Name?custom-option=value&sap-client=120

[Open in new Window](#)

[200] POST

http://localhost:8080/proxy/https/ldciqal.wdf.sap.corp:44326/sap/opu/odata4/sap/zui5\_testv4/default/sample/0002/\$batch?custom-option=value&sap-client=120

2019-06-14T14:44:12.474Z: 326.23400000011316 ms

[200] GET SalesOrderList?custom-option=value&sap-client=120&\$count=true&\$filter=SalesOrderID%20ge%20'0500000000'%20and%20LifecycleStatus%20eq%20'N'%20and%20(CompanyName%20ge%20'M')&\$select=BuyerID,ChangedAt,CurrencyCode,GrossAmount,LifecycleStatus,LifecycleStatusDesc,Messages,Note,SalesOrderID,\$expand=SO\_2\_BP(\$select=BusinessPartnerID,CompanyName)&\$skip=0&\$top=5

[Open in new Window](#)

[200] POST

http://localhost:8080/proxy/https/ldciqal.wdf.sap.corp:44326/sap/opu/odata4/sap/zui5\_testv4/default/sample/0002/\$batch?custom-option=value&sap-client=120

2019-06-14T14:44:19.344Z: 85.36700000018347 ms

[200] GET BusinessPartnerList?custom-option=value&sap-client=120&\$orderby=CompanyName&\$filter=BusinessPartnerRole%20eq%20'01'&\$select=BusinessPartnerID,CompanyNa

```
1 {
2   "request": {
3     "headers": {
4       "Accept": "application/json;odata.metadata=minimal;IEEE754Compatible=true",
5       "Accept-Language": "en-US",
6       "X-CSRF-Token": "mi9lqnk3Ijcn3PvvBw4BBQ==",
7       "Content-Type": "application/json;charset=UTF-8;IEEE754Compatible=true"
8     },
9     "method": "GET",
10    "url": "SalesOrderList?custom-option=value&sap-client=120&$count=true&$filter=SalesOrderID%20ge%20'0500000000'%20and%20LifecycleStatus%20eq%20'N'%20and%20(SO_2_BP/CompanyName%20ge%20'M')&$select=BuyerID,ChangedAt,CurrencyCode,GrossAmount,LifecycleStatus,LifecycleStatusDesc,Messages,Note,SalesOrderID,$expand=SO_2_BP($select=BusinessPartnerID,CompanyName)&$skip=0&$top=5",
11    "httpVersion": "HTTP/1.1"
12  },
13  "response": {
14    "headers": {
15      "Content-Length": "2419",
16      "content-transfer-encoding": "binary",
17      "Content-Type": "application/json;ieee754compatible=true;odata.metadata=minimal",
18      "odata-version": "4.0",
19      "cache-control": "no-cache, no-store, must-revalidate",
20      "sap-messages": [
21        {
22          "code": "ZUI5_EPM_SAMPLE/001",
23          "message": "SAP UI5con learn.explore.connect",
24          "numericSeverity": 1
25        },
26        {
27          "code": "ZUI5_EPM_SAMPLE/002",
28          "message": "System Maintenance on 20190614 at 12h until 15h UTC",
29          "numericSeverity": 3
30        }
31      ]
32    },
33    "status": 200,
34    "statusText": "OK",
35    "body": {
36      "@odata.context": "$metadata#SalesOrderList(BuyerID,ChangedAt,CurrencyCode,GrossAmount,LifecycleStatus,LifecycleStatusDesc,Messages,Note,SalesOrderID,SO_2_BP(BusinessPartnerID,CompanyName))",
37      "@odata.metadataEtag": "W/\"20190611150141\"",

```

# Conclusion / Outlook



# Conclusion / Outlook

In the past two years a variety of new features has been added to the OData V4 Model.

- It is possible to develop Business Applications with the OData V4 Model.
- Some of the features are not available with OData V2 like the creation of multiple items without saving or the formatting of quantities and amounts according to the backend customizing.

Yet, there are still some topics waiting.

- Further improvements for CRUD in non-Draft applications, e.g. deferred delete
- Improvements to the data storage in the model
- OData V4 Analytics (“\$apply”)
- Hierarchies

## Meet the Experts

Meet us at Expert Table 2 from 13:45 to 15:00

# Thank you.

Contact information:

**Thomas Chadzelek**  
Development Architect

**Patric Ksinsik**  
Development Architect

**Mathias Uhlmann**  
Product Expert  
[Mathias.Uhlmann@sap.com](mailto:Mathias.Uhlmann@sap.com)

# UI5con

learn.explore.connect.