


# CPS-461 Align DataNode for Get and Post/Put endpoints in CPS Core

 **CPS-461** - Spike for aligning JSON DataNode for Get and Post/Put endpoints in CPS Core CLOSED

- [Introduction](#)
  - [Yang module used for testing](#)
- [Solution](#)
- [Backwards Compatibility](#)
- [Backwards Incompatibility:](#)

## Introduction

### Yang module used for testing

## bookstore.yang

```
module stores {
  yang-version 1.1;
  namespace "org:onap:ccsdk:sample";

  prefix book-store;

  revision "2020-09-15" {
    description
      "Sample Model";
  }

  typedef year {
    type uint16 {
      range "1000..9999";
    }
  }

  container bookstore {

    leaf bookstore-name {
      type string;
    }

    list categories {

      key "code";

      leaf code {
        type string;
      }

      leaf name {
        type string;
      }

      list books {
        key title;

        leaf title {
          type string;
        }
        leaf lang {
          type string;
        }
        leaf-list authors {
          type string;
        }
        leaf pub_year {
          type year;
        }
        leaf price {
          type uint64;
        }
      }
    }
  }
}
```

It has been noted that the json content for read and write requests in CPS-Core is inconsistent

Description	Response for getDataNode root node	Response for getDataNode non root node	body when writing (root) DataNode
-------------	------------------------------------	----------------------------------------	-----------------------------------

Operation	GET	GET	POST/PUT
xPath	/	/bookstore/categories[@code='01']	/
Response /Body	{ "bookstore-name": "Chapters", "categories": [ ... ] }	{ "code": "01", "name": "SciFi", ... }	{ <b>"bookstore":</b> { "bookstore-name": " Chapters", "categories": [ ... ] } }
Notes	The response includes the 'value' of the bookstore node ie. the leaves and children	<= Same	the body needs to include the whole container object.

## Solution

#	Solutions	Issues
1	Use the parent node xpath in order to wrap the queried leaves. eg query xpath = /bookstore/categories[@code='01'] => categories	This should be reasonably straightforward as the node is named after the container name given by the module yang file.

For the JSON output of Get DataNode we need to alter the DataMapUtils class. The function toDataMap translates a datanode object to a JSON output. We only want this to occur on the parent node and as such we can create a new method which is called before toDataMap:

```
public static Map<String, Object> toDataMapParentNode(final DataNode dataNode) {
    final String nodeName = dataNode.getXpath().substring(dataNode.getXpath().lastIndexOf('/') + 1)
        .replaceAll("\\[.*?\\]", "");
    return ImmutableMap.<String, Object>builder().put(nodeName,
        toDataMap(dataNode)
    ).build();
}
```

We only want the container name on the highest parent level of the JSON output and as such have to distinguish between a parent level node and non parent level node. We create an outer map to wrap the inner map which creates the appropriate levels in the JSON output.

This produces the following response in the Post Request output:

Description	Response for getDataNode root node	Response for getDataNode non root node	Response for queryDataNode root node	Response for queryDataNode non root node
Operation	GET	GET	GET	GET
xPath	/	/bookstore/categories[@code='01']	/bookstore	/bookstore/categories[@code='01']
Response /Body	{ <b>"bookstore":</b> { "bookstore-name": "Easons", "categories": [...] } }	{ <b>"categories":</b> { "code": "01", "name": "SciFi", "books": [...] } }	{ <b>"bookstore":</b> { "bookstore-name": "Easons", "categories": [...] } }	{ <b>"categories":</b> { "code": "01", "name": "SciFi", "books": [...] } }
Notes	Response includes the container bookstore	The response includes the list categories	Response includes the container bookstore	The response includes the list categories

Tests would need to be updated to accept the new JSON output which is returned. Examples would need to be updated in openapi

## Backwards Compatibility

Queries which use toDataMap:

	Effected Components	Effected Clients	Query /Service	Class	Notes	Response now	Response After	Decision
1	CPS Core	CPS TBDMT	Get Data Node (as above)	DataRest Controller.java	This change is given in the example above. The JSON output for Get Node will be updated.	{ "bookstore-name": " Chapters", "categories": [ ... ] }	{ <b>"bookstore":</b> { "bookstor e-name": " Chapters", "categories": [ ... ] } }	Don't update version, check with tbdmt about impact of new JSON output for getNode/queryNode.

2	CPS Core	CPS Temporal	Notification Service	CpsData Updated EventFactory.java	Notification Service response should be the same as CPS Core for consistency. CPS Temporal	cpsDataUpdatedEvent Object content. data. additionalProperties.toString();  [[bookstore-name: Chapters]]	cpsDataUpdatedEvent Object content. data. additionalProperties.toString();  [bookstore:[bookstore-name: Chapters]]	CPS Temporal only recording and not processing this data. Therefore change (response after) is OK
3	CPS Core	CPS TBDMT	Query data nodes	QueryRestController.java	Query DataNodes will produce a datanode within an array with the container ID  and then the relevant data:  <a href="http://localhost:8883/cps/api/v1/dataspaces/test/anchors/bookstore-anchor-test2/nodes/query?cps-path=/bookstore/categories[@code='01']&amp;include-descendants=true">http://localhost:8883/cps/api/v1/dataspaces/test/anchors/bookstore-anchor-test2/nodes/query?cps-path=/bookstore/categories[@code='01']&amp;include-descendants=true</a>	[  {  "code": "01", "name": "SciFi", "books": [ ... ] }]	[  "categories": { "code": "01", "name": "SciFi", "books": [ ... ] } ]	Don't update version, check with tbdmt about impact of new JSON output for getNode/queryNode.
4	CPS Core	None	getNode by CMHandle and Xpath (NCMP)	NetworkCmProxyController.java	This method is deprecated and is no longer used as part of NCMP.			

## Backwards Incompatibility:

	Keep version	New version	New endpoints
Action	Keep the current version of the interface and alert effected client TBDMT of the changes.	Release version 2 of the interface with changes while retaining v1 endpoints and output.	Create new endpoints and deprecate the old endpoints sometime in the future. Keep the endpoints for get and query which we currently have and release new endpoints that have container context.
Preference/Impact	Current preferred option as long as TBDMT agree.  Impacts TBDMT.	Would need its own spike to analyze the impact. CPS Temporal would need the context of if request is coming from v1 or v2 for notification service.	A lot of duplicated code which will need to be deprecated in the future. CPS Temporal would need the context of if request is coming from new or old endpoints.