International Civil Aviation Organization

THE THIRD WORKING SESSION OF ELEVENTH MEETING OF SWIM IMPLEMENTATION PIONEERING GROUP (SIPG WS/3)

Bangkok, Thailand, 01 – 05 June 2026

Agenda Item 3: SWIM Technical Infrastructure
- Routing Mechanism

DISCUSSION ON KEY IMPLEMENTATION ISSUES FOR R/R MEP IN THE APAC SWIM

(Presented by the Republic of Korea)

SUMMARY

In developing the R/R MEP related guidance materials, several issues were identified where different views remain among members. This paper presents these issues for consideration and further discussion by the meeting.

1. INTRODUCTION

1.1 At the SWIM TF/10, discussions on implementing the R/R MEP in the APAC region have gained momentum following the definition of the initial APAC common SWIM information services under the SWIM TF (WP11, SWIM TF/10). And the SIPG established Task 3 – Guidance for Synchronous and Asynchronous R/R MEP, to conduct documentation and implementation activities related to R/R MEP within the APAC SWIM environment.

1.2 Task 3 has been developing guidance materials related to R/R MEP, and the progress of each document is as follows:
- R/R MEP Guidance Material (Draft)
- REST API URI Structuring & Naming Guidance Material (Draft)
- API GW Requirement Specification Document (In progress)

1.3 The draft guidance materials presented at the SWIM TF/11 still have several issues which require further discussion. During the review conducted through SIPG monthly meetings and email exchanges, members raised various comments and views. Some of these issues are not merely editorial, but relate to differing views on implementation approaches, technical directions. In order to support further progress of Task 3 and implement R/R MEP in the APAC SWIM prototype, it is necessary to bring these issues to the attention and seek further consideration by SIPG members. Therefore, this paper presents these issues currently identified.

2. DISCUSSION

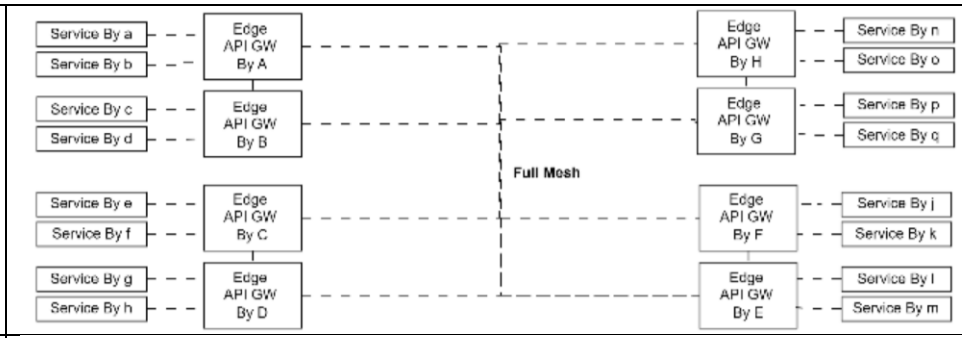
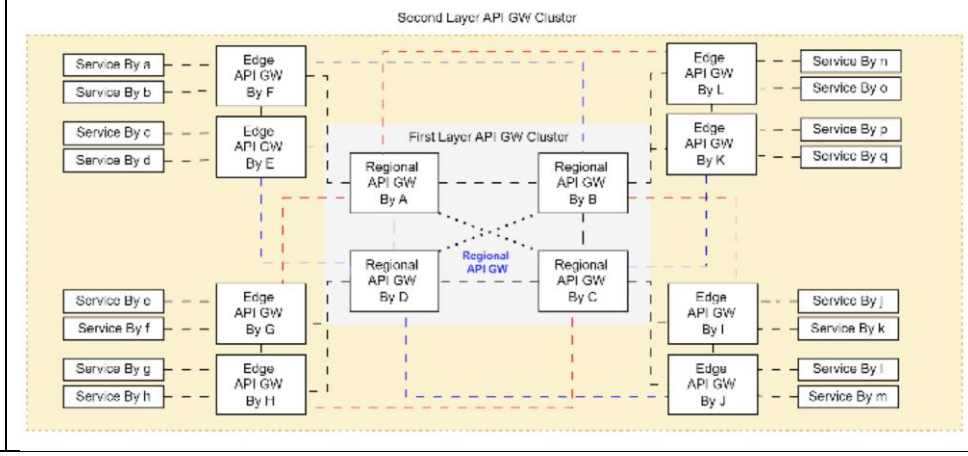
2.1 This section presents the identified issues and provides a brief explanation of each issue. If there are multiple aspects, the summary of key points for each view is described. In order to facilitate open discussion at the meeting and maintain neutrality, specific solutions or preferred options are not proposed.

Issue #1 Full Mesh or Hierarchical Architecture for Synchronous R/R MEP

To enable synchronous R/R MEP, API Gateways (or proxies) need to be deployed to deliver message (i.e., requests and replies), similar to the role of message brokers in the P/S MEP. Therefore, one of the key issues is how API Gateways (or proxies) should be deployed within the APAC SWIM environment.

Task 2 adopts hierarchical architecture for the P/S MEP, consisting of EEMS and GEMS, and its implementation is based on a message broker. If the synchronous R/R MEP also adopts a hierarchical architecture, the member states responsible for hosting EEMS and GEMS need to host an edge API GW and a global (or regional) API GW (or proxy). On the other hand, if the synchronous R/R MEP adopts a full-mesh architecture, it literally means that the synchronous R/R MEP will have an independent information backbone in parallel with the one for P/S MEP and asynchronous R/R MEP. In this case, there is no role for member states responsible for hosting GEMS, nor do all member states operating EEMS need to have their own API GWs. Also, a network routing policy must be configured by PCCW to support the full-mesh connectivity.

Furthermore, if the synchronous R/R MEP adopts a hierarchical architecture**, the definition and scope of EEMS/GEMS must be re-evaluated**. Under this framework, EEMS/GEMS will expand to encompass not only the edge and global message brokers for supporting P/S and asynchronous R/R MEPs, but also the edge and global (or regional) API Gateways (or proxies) required for synchronous R/R MEP.

Full Mesh Architecture	
Hierarchical Architecture	

R/R MEP guidance material describes pros and cons of full mesh and hierarchical architecture as follows:

Architecture	Pros	Cons
Full Mesh	- Low dependency between API GWs - Easy to implement - High sovereign	- Adding a new member requires updates across all existing nodes - Any change or update propagates to all members - A failure in one node can have a system-wide impact.

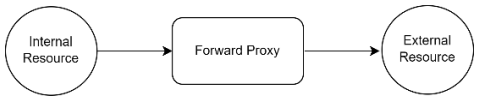
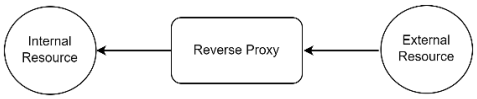
Hierarchical	- Higher fault tolerance - Better HA and scalability	- Hight cost for maintenance - Management complexity

Issue #2 Use of API GW or Proxy at the Global Level

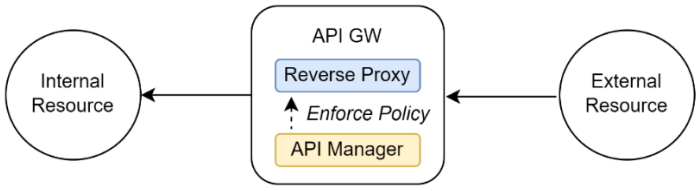
If a hierarchical architecture is adopted to enable synchronous R/R MEP in APAC SWIM, which is similar to that of the P/S MEP, another issue arises as to what type of solution should be adopted.

Some solutions could be used, and R/R MEP guidance material describes 1) proxy, 2) reverse-proxy, and 3) API GW for a R/R MEP solution in charge of middle-mile message delivery as follows :

At the edge level, there is consensus on the use of API GW. However, at the global level, different views remain as to whether API GW or proxy should be used.

<table>
<tr><td rowspan="3">Proxy</td><td rowspan="2">Diagram</td><td>Forward</td><td> 
</td></tr>
<tr><td>Reverse</td><td> 
</td></tr>
<tr><td>Explanation</td><td colspan="2">A forward proxy acts on behalf of an internal resource (e.g., client, server, or system), managing outbound requests to an external resource (e.g., client, server, or system) to provide capabilities such as access control, caching, and monitoring.

A reverse proxy acts on behalf of an external resource (e.g., client, server, or system), managing inbound requests to internal resource (e.g., client, server, or system) to security, load balancing, and routing, while hiding internal system details.

Forward and reverse proxy supports OSI 3rd, 4th, and 7th layer protocols such as HTTP, Web-socket, TCP, UDP, IP. Main focus of forward and reverse proxy is message forwarding.</td></tr>
<tr><td rowspan="2">API GW</td><td>Diagram</td><td colspan="2"> 
</td></tr>
<tr><td>Explanation</td><td colspan="2">An API GW is built on top of a reverse proxy, primarily supporting HTTP and providing advanced API management capabilities. The main difference between an API GW and a reverse proxy lies in how policies are applied and managed from an API management perspective.</td></tr>
</table>

Issue #3 Alignment with Topic and REST API URI Naming and Structuring Convention

The Section - 4.1. Rationale for URI Structuring and Naming in the REST API URI Structuring & Naming Guidance Material addresses the needs of common URI structuring and naming convention. However, considerations regarding such URI structuring and naming conventions are currently being discussed under two tasks in the SIPG: one is the topic naming convention being considered in Task 2, and the other is the REST API URI structuring and naming convention being considered in Task 3.

There is an opinion that alignment between the Topic naming and structuring convention and the REST API URI naming and structuring convention would be beneficial. As of April 2026, defined structuring conventions of Topic and REST API URI are as follows:

Topic	amqp://{message-broker-host}:{message-broker-port} /{type-of-address} /{country-code} /{organization} /{environment} /{system-name} /{version} /{system-specific-topic-hierarchy}
REST API	http://{api-gateway-host}:{api-gateway-port} /{region-code} /{country-or-jurisdiction-code} /{service-provider} /{service-name} /{service-version} /{resource} ?{query-parameters}

Explanations of each field defined in Topic and REST API URI are as follows:

Topic	{type-of-address}	Indicate that it is topic address. It is fixed value.
	{country-code}	The country code is based on ISO 3166-1 alpha3.
	{organization}	Company, corporation, agency, or an institution.
	{environment}	It is often necessary to have multiple environments to work on. For example, production, testing, offline and others
	{system-name}	System Name with optional service group name
	{version}	Indicate the version of the API or event.
	{system-specific topic-hierarchy}	Hierarchy and topic field is determined by system. Topic recommendation is {noun}/{verb}/{property}
REST API	{region-code}	ICAO regional identifier (e.g. APAC) used to support regional-level routing and governance
	{country-or-jurisdiction-code}	{country-code} : Country or territory identifier based on ISO 3166-1 alpha-3 {jurisdiction code} : Registered jurisdictional or regional routing namespace used when the service is provided under a regional, multinational, or centrally managed SWIM environment where a single State or territory code is not applicable. Topic Naming Convention > Country Code
	{service-provider}	Identifier of the organization or entity providing the service Topic Naming Convention > Organization

	{service-name}	Name of the exposed resource or service Topic Naming Convention > System Name
	{service-version}	Version identifier of the service (e.g. vX) Specific version of the service SHALL be explicitly indicated in the HTTP headers. GET /gufi-service/validate Host : example.aero Accept : application/json **API-Version : 1.1.1** Topic Naming Convention > Version
	{resource} <service-defined-path>	A relative URI (Uniform Resource Identifiers) to an individual endpoint. {resource} could have hierarchical path (i.e., service-defined-path) depending on service (e.g., /a/b/c/) Topic Naming Convention > System Specific Topic Hierarchy

While it is not necessary to fully align URI naming conventions between topics and REST APIs, if there are common fields across both, alignment might be needed to reuse consistent data definitions to ensure overall uniformity at the API governance level

Issue #4 Degree of Simplification of REST API URI Naming and Structuring Convention

There is an opinion that the REST API URI structure should be simplified by limiting URI path elements to mandatory runtime elements only. Under the current draft convention, URIs may include non-runtime descriptors such as country, service provider, and they may not always be necessary for runtime routing.

In addition, excessive inclusion of service parameters in the URI may increase coupling between URI structures and backend configurations. This could result in broken links and additional maintenance overhead when backend configurations change. Therefore, further discussion is required on the appropriate degree of simplification and the possible use of API proxies or abstraction layers.

According to this view, the option suggests using the metadata to be exposed through SWIM registry or SDS. By discovering service description from SWIM registry, detailed information to get access to the end-point of service to be available.

An example of a service description compliant with the SDM-J 2.0.0 schema is shown below. Some non-essential elements could be referenced for routing purposes.

```
curl -X GET "http://223.130.138.155:8000/smxs/v2/service-description?service-id=http://223.130.138.155:8000/smxs/v2/service-description/SMXS-2.0.0" \
  -H "Accept: application/json" \
  -H "Accept-Language: en"

…
"resource": [
        {
           "id": "http://223.130.138.155:8000/smxs/v2/discovery-service",
```

```
            "name": "SDS Instance Metadata",
            "description": "Access root instance metadata for the KAC SWIM SDS.",
            "path": "http://223.130.138.155:8000/smxs/v2/discovery-service",
            "method": "GET",
            "parameter": [
              {
                "name": "Accept",
                "description": "Required media type (application/json).",
                "parameter-type": "header",
                "permissible-value": [
                  "application/json"
                ]
              },
              {
                "name": "Accept-Language",
                "description": "Required language (en).",
                "parameter-type": "header",
                "permissible-value": [
                  "en"
                ]
              }
            ]
          },
        ]
    },
    "grounding": {
      "binding": {
        "name": "SMXS REST Binding",
        "interface": "SWIM Metadata Exchange Service (SMXS)",
        "description": "Standard RESTful binding over HTTP for metadata discovery.",
        "protocol": {
          "id": "http://www.ietf.org/rfc/rfc2616.txt",
          "title": "Hypertext Protocol -- HTTP/1.1",
          "publisher": "IETF",
          "date-issued": "1999-06-01",
          "version": "1.1",
          "location": "http://www.ietf.org/rfc/rfc2616.txt"
        }
      },
      "endpoint": {
        "name": "KAC SMXS v2.0 Instance",
        "description": "KAC SMXS v2.0 Instance (Test Only)",
        "network-address": "http://223.130.138.155:8000/smxs/v2/"
      }
    }
…
```


Issue #4 Alignment with Naming and Structuring Convention defined by other group (e.g., MET)

At the monthly meeting, it was reported that the MET Group had developed a separate naming convention for synchronous R/R MEP based MET information services. Members of MET group provided material to review as follows:

- SADIS User Guide
- OGC EDR API standard

Upon review of the materials provided, it was observed that the conventions defined by the MET group focus on domain-specific resource structures for MET information services, therefore, no significant conflict with the current draft version of REST API URI Structuring & Naming Guidance Material was found.

For example, a MET information service based on the OGC API EDR guidance material could be mapped as follows:

OGC API EDR Structure
/collections/{collectionId}/locations/{locationId}
?datetime={datetime}

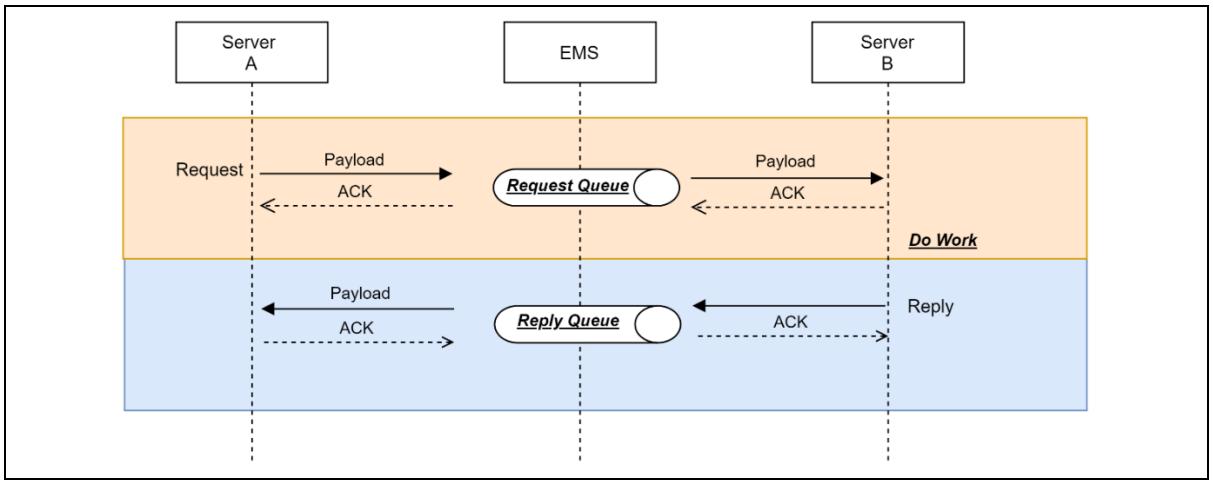
Mapping with APAC REST API URI Structure
/{region-code}
/{country-or-jurisdiction-code}
/{service-provider}
/{service-name}
/{service-version}
/{resource}
?{query-parameters}

Mapped Example
/{region-code}
/{country-or-jurisdiction-code}
/{service-provider}
/{service-name}
/{service-version}
/collections/{collectionId}/locations/{locationId}
?datetime={datetime}

Practical Example
/apac/kor/kamo/met-edr-service/v1**/collections/taf/locations/rkss**
?datetime=2026-05-08T00:00:00Z

Issue #5 Alignment between synchronous and asynchronous R/R MEP in the aspect of implementation

The Section - 5.3. Approach on Implementation of R/R MEP in Asia/Pacific Region defined an approach to implement asynchronous R/R MEP. First priority is AMQP-based asynchronous request and reply through GEMS-EEMS. Also, Task 2 is advancing the implementation of asynchronous R/R MEP in parallel with P/S MEP. At some point, discussion is required how to align with activities done by both Task2 and Task 3 for R/R MEP. (i.e., alignment between synchronous R/R MEP and asynchronous R/R MEP.



3. ACTION BY THE MEETING

3.1 The meeting is invited to:

a) note the information contained in this paper; and

b) discuss any relevant matter as appropriate

– – – – – – – – – – – – – – –