Database-based Open Resource Service (DB-ORS) Framework, Usecases and Requirements

Fenlin Zhou; Dongyu Yuan; Sheng Wang; Xiaocong Qian
ZTE Corporation, China
Documents

• Requirements and Use Cases of DB-ORS (Database-based Open Resource Service)

• Database-based Open Resource Service Framework
Challenges in Current Network

Challenge in fine-granularity service provisioning:

Capabilities of the network remain invisible, thus **differentiated services** are not achieved. Applications with various requirements can **not be distinguished and served customarily**.
Challenges in network resources utilization:

Due to the **bandwidth restrictions** in the Shanxi-Hebei and Hebei-Liaoning sections, the bandwidth provided by the network for applications can not achieve 100 G, and the **bandwidth resources from GanSu to ShanXi is wasted**.
Requirements of DB-ORS

• Fine-granularity services provisioning
  • Conventional networks only provide clients with coarse-grained connection services.
  • Services which propose diverse network requirements such as ultra-low latency and high reliability emerge increasingly, and therefore differentiated service treatment is desired.

• Network resources utilization enhancement
  • The network resources are not orchestrated appropriately and the resource utilization proves to be relatively low for about 30%-50%.
  • The complexity of the network results in low bandwidth utilization of the real data load.

Besides bandwidth for instance, the network has also been endowed with various other capabilities including deterministic quality, network slicing, endogenous security, etc.

Referring to the concept of Software-as-a-Service (SAAS), capabilities of the network are abstracted into services, namely Network-as-a-Service (NAAS).

Applications (cloud, terminal, and CPE) subscribe corresponding customized network services.
Framework of DB-ORS

Service subscription
- Programmability of DB-Agent
- Efficiency of a ‘watch’ mechanism
- Simplicity of API and Extensibility

Service abstraction
- VDlink
- VTLlink
- Node
- EPE Link

Service publication
Distribute database, e.g. ETCD:
- Key-value model
- Standard schema template

Service re-orchestration
- Application identification
- Network path calculation
- Service orchestration & binding

Usecase of DB-ORS: abstract network capabilities
Usecase of DB-ORS: publish and subscribe services

### Perspective of Cloud A:

<table>
<thead>
<tr>
<th>Link</th>
<th>Link type</th>
<th>Logic Link ID</th>
<th>cost</th>
<th>delay</th>
<th>max bw</th>
<th>END/X/INT</th>
<th>BSID (segmentList)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A—B</td>
<td>vdlink</td>
<td>536870913</td>
<td>10</td>
<td>10us</td>
<td>3g</td>
<td>3c:1/1/1nt1</td>
<td>--</td>
</tr>
<tr>
<td>B—D</td>
<td>vdlink</td>
<td>536870914</td>
<td>10</td>
<td>10us</td>
<td>3g</td>
<td>3c:2/2/2nt3</td>
<td>--</td>
</tr>
<tr>
<td>A—D</td>
<td>vTlink</td>
<td>805306369</td>
<td>20</td>
<td>20us</td>
<td>5g</td>
<td>--</td>
<td>4BA:1 (d1A—Dd d2A—C—D)</td>
</tr>
</tbody>
</table>

### Perspective of Cloud B:

<table>
<thead>
<tr>
<th>Link</th>
<th>Link type</th>
<th>Logic Link ID</th>
<th>cost</th>
<th>delay</th>
<th>max bw</th>
<th>END/X/INT</th>
<th>BSID (segmentList)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A—B</td>
<td>vdlink</td>
<td>1073741825</td>
<td>10</td>
<td>10us</td>
<td>5g</td>
<td>3c:2/2/2nt2</td>
<td>--</td>
</tr>
<tr>
<td>B—D</td>
<td>vdlink</td>
<td>1073741826</td>
<td>10</td>
<td>10us</td>
<td>5g</td>
<td>3c:2/2/4nt4</td>
<td>--</td>
</tr>
<tr>
<td>A—D</td>
<td>vTlink</td>
<td>1342177281</td>
<td>20</td>
<td>20us</td>
<td>5g</td>
<td>--</td>
<td>4FE:1 (d1A—D)</td>
</tr>
</tbody>
</table>

### Key-Vaule in DataBase

- Cloud_A/VTLink/Logic_id 536870913
- Cloud_A/VTLink/Logic_id/linktype vdlink
- Cloud_A/VTLink/Logic_id/delay 10

### Standard Schema Template

```json
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "title": "Abstract definition of CN2-DCI network resources.",
  "description": "version v1.0.0",
  "properties": {
    "vTLink": {
      "description": "Logic top",
      "type": "array",
      "item": {
        "type": "object",
        "properties": {
          "id": {
            "description": "logic link id",
            "type": "string"
          },
          "delay": {
            "description": "link delay",
            "type": "string"
          }
        }
      }
    }
  }
}
```
<table>
<thead>
<tr>
<th>Scenario</th>
<th>ALTO</th>
<th>DB-ORS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Universal model of network capability exposure for various scenarios</td>
<td>Focus on abstraction of network capabilities, especially for the integration of the cloud and the network</td>
</tr>
<tr>
<td>Abstraction</td>
<td>Coarse-granularity, simple descriptions</td>
<td>Fine-granularity, explicit and diversified abstractions</td>
</tr>
<tr>
<td>Application</td>
<td>Endpoints selection in the <strong>Application layer</strong></td>
<td>Traffic steering and redirection in the <strong>Network layer</strong></td>
</tr>
<tr>
<td>Framework</td>
<td>C/S framework with a unique ALTO server</td>
<td>C/S framework with a distributed database as the server</td>
</tr>
<tr>
<td>API</td>
<td>Unified REST+JSON</td>
<td>Unified REST+JSON</td>
</tr>
<tr>
<td>Expandability</td>
<td>Easy to expand</td>
<td>Easy to expand</td>
</tr>
</tbody>
</table>
Conclusions and Considerations

What have we done & what do we expect?

• Solve the mentioned problems.
• More perceptions and drafts are expected in the future.
• Promote standardization, cooperate with working groups who have resonated with this issue.

Considerations in the future?

• More abstraction of network capabilities.
• Safety considerations.
• Service affinity.