SDN Heading North: Towards a Declarative Intent-based Northbound Interface

Shiyam Alalmaei, Matthew Broadbent, Yehia Elkhatib, Mehdi Bezahaf, and Nicholas Race

School of Computing and Communications, Lancaster University, United Kingdom

Email: {s.alalmaei, m.broadbent, y.elkhatib, mehdi.bezahaf, n.race}@lancaster.ac.uk
Motivation

• Why is the NBI important?

• The success of SDN relies on application developers leveraging its capabilities.

• Hence, the Northbound Interface (NBI) is the key enabler for the realization of the ultimate SDN promise.

• Although Intent-based NBIs are gaining a lot of attention, their development remains in its infancy.
Outline

• Intent-Based NBI Challenges

• Related Work Limitations

• Proposed Intent-Based NBI framework and expressions

• A proof-of-concept cloud CDN use case of a caching intent

• Intent Refinement

• Conclusion & Future Work
### Intents vs. Policies

<table>
<thead>
<tr>
<th>Similarities</th>
<th>Differences</th>
</tr>
</thead>
</table>
| **Policies** | • *Prescriptive* rules  
               • Specify set of Event-Condition-Action (ECA) rules and determine precisely what to do under different circumstances and triggers (how to do?)  
               • Used by system experts who can articulate the set of rules  
               • System behavior defined proactively |
| **Intents**  | • *Declarative* expressions  
               • Express desired outcome (what to do?)  
               • Can be used by different users including service consumers and non-experts without enumerating rules  
               • Could be a learning reactive system |
Intent-Based NBIs Challenges

• Service consumers require **declarative** rather than **prescriptive** intent expressions.

• Translating service-oriented intents to system operations needs **intermediate interpretation** as policies.

• Intent-based NBI must be **platform-independent** and **extensible**
Related Work Limitations

- In general, current Intent-based NBIs are limited, ad-hoc and vendor-specific.

- They do not allow expressing declarative intents that handle other requirements beyond the network-level.

- Moreover, most current works don’t provide the tools to create new intents and map them to lower-level policies.
Intent-Based NBI Framework
Intent-Based NBI Syntax

Declarative Service-Oriented Intent Syntax

```
<SERVICE><RESOURCES><CONJUNCTION><TARGET>
```

Prescriptive Policies Syntax

```
<SERVICE><RESOURCES>
  <POLICY 1><OPERATOR><POLICY 2><OPERATOR>...
  {<POLICY k1><OPERATOR><POLICY k2><OPERATOR>...<POLICY km>}
  <OPERATOR>...<POLICY n>
</POLICY><CONDITIONS><ACTIONS><CONSTRAINTS>[[PRIORITY]]
```

Intent/Policy Mapping

<table>
<thead>
<tr>
<th>Tag</th>
<th>Basic Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;SERVICE&gt;</td>
<td>Caching</td>
</tr>
<tr>
<td>&lt;RESOURCES&gt;</td>
<td>contents to be cached</td>
</tr>
<tr>
<td>&lt;CONJUNCTION&gt;</td>
<td>that can handle, that can meet, etc.</td>
</tr>
<tr>
<td>&lt;TARGET&gt;</td>
<td>&lt;WORKLOAD&gt;</td>
</tr>
<tr>
<td>&lt;WORKLOAD&gt;</td>
<td>&lt;NUMBER&gt; &lt;UNIT&gt; or &lt;ADJECTIVE&gt; &lt;UNIT&gt; or &lt;ADJECTIVE&gt; “workload”</td>
</tr>
<tr>
<td>&lt;NUMBER&gt;</td>
<td>numeric values that can represent the workload</td>
</tr>
<tr>
<td>&lt;UNIT&gt;</td>
<td>GB/min, requests/sec, etc.</td>
</tr>
<tr>
<td>&lt;ADJECTIVE&gt;</td>
<td>max, min, dynamic, high, medium, etc.</td>
</tr>
<tr>
<td>&lt;CONDITIONS&gt;</td>
<td>new caching request, max threshold exceeded, ...</td>
</tr>
<tr>
<td>&lt;ACTIONS&gt;</td>
<td>allocate cache servers(), scale out(), ...</td>
</tr>
<tr>
<td>&lt;CONSTRAINTS&gt;</td>
<td>with max storage, with least latency, ...</td>
</tr>
<tr>
<td>&lt;PRIORITY&gt;</td>
<td>optional indicator of policy priority</td>
</tr>
<tr>
<td>&lt;OPERATOR&gt;</td>
<td>Policy will be executed in parallel / sequential way</td>
</tr>
</tbody>
</table>
Cloud CDN (CCDN) Use Case

In our solution (opposed to Cloud CDNs today), Content Providers can express their high-level declarative intents targets (e.g., expected caching workload) which leads to better caching and resource management decisions with respect to the intent’s target (well-, under-, and over-) estimation.

Content Provider (intent user)

CCDN Operator (intent developer)

I want caching for content X to handle 20 GB/min

```json
{
  "Resources-Allocation": {
    "Conditions": "New Caching Service Request",
    "Actions": "Allocate Cache Servers",
    "Constraints": "Average Number Of Servers"
  },
  "Cache-Service-Resizing": {
    "Scale-up": {
      "Conditions": "Max Threshold Exceeded",
      "Actions": "Add more caches",
      "Constraints": "Number Of Caches to Add"
    },
    "Scale-down": {
      "Conditions": "Underutilized Threshold",
      "Actions": "Remove some Caches",
      "Constraints": "Time"
    }
  }
}
```
Cache cluster size and handled data rate in a reactive traditional CCDN (without intents) and an Intent-based one.
Intent Refinement

To maximize the handled workload, the intent system would refine intent policies based on the previous demands and cluster size analysis. A way to achieve this is to scale out/in proactively. Two approaches can be taken:

• **conservative** method - aims to minimize the cost of deploying extra caches by sacrificing some unhandled requests;

• **greedy** method - handles more requests at the expense of additional deployment cost.
Intent Refinement Results

Cache cluster resizing (Conservative vs. Greedy policy)

<table>
<thead>
<tr>
<th></th>
<th>Traditional (No intent)</th>
<th>Intent-Based, conservative</th>
<th>Intent-Based, Greedy</th>
<th>Oracle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unhandled Requests</td>
<td>3.17% ~ 967 GB</td>
<td>0.96% ~ 295 GB</td>
<td>0% ~ 0 GB</td>
<td>0% ~ 0 GB</td>
</tr>
<tr>
<td>Cost (minutes)</td>
<td>0</td>
<td>82</td>
<td>286</td>
<td>20</td>
</tr>
</tbody>
</table>
Intent Refinement Results

Handled Requests (Conservative vs. Greedy policy)
Conclusions

• Discussed the limitations of the current Intent-based solutions

• Proposed Intent-Based NBI framework

• Proposed a Service Consumer Intent declarative expression along with its corresponding prescriptive policies

• Demonstrated a caching workload intent, and its corresponding policies and refinement in a CCDN use case
Future Work

• Extend intent-to-policy mapping to be dynamic based on several criteria and map them to existing Microservices.

• Investigate different intent targets.

• Implement the intent-based framework in a real cloud-based testbed along with the required APIs and translations.
Different Intent Types

Who
- Tech, or non-tech. users

How
- Abstract or tech. specific

Where
- Data center, cloud, etc
# Meta-Analysis

<table>
<thead>
<tr>
<th>Intent-Based Solution</th>
<th>Intent Expression</th>
<th>Domain</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>(NEMO) by Huawei [10]</td>
<td>Object + Operation or</td>
<td>Networking</td>
<td>Presc.</td>
</tr>
<tr>
<td></td>
<td>Object + Result (under test and not used yet)</td>
<td>Decl.</td>
<td></td>
</tr>
<tr>
<td>(DOVE) by IBM [13]</td>
<td>Not specified</td>
<td>Networking / NFV</td>
<td>–</td>
</tr>
<tr>
<td>Intent-based NBI service-oriented architecture [16]</td>
<td>application-specific language</td>
<td>Networking</td>
<td>–</td>
</tr>
<tr>
<td>(iNDIRA) [17]</td>
<td>Subject (Service or Condition), Relationship (has Arguments), Objects (multiple parameters)</td>
<td>Networking</td>
<td>Presc.</td>
</tr>
<tr>
<td>(SENSE) [18]</td>
<td>Service type, Service alias, Connections:  {name, terminals, bandwidth: { qos_class, capacity, unit}}, schedule: {start, end, duration}</td>
<td>Networking / NFV</td>
<td>Presc.</td>
</tr>
<tr>
<td>Northbound Interface [22]</td>
<td>Predicate, Commodity, Target (resources), Constraint, Condition</td>
<td>Networking</td>
<td>Presc.</td>
</tr>
<tr>
<td>Adaptive Service Deployment [23]</td>
<td>Verb, Object, Modifiers, Subject</td>
<td>General use cases, e.g. storage, caching, IDS</td>
<td>Presc.</td>
</tr>
</tbody>
</table>

TABLE I: Summary of the results of our meta-analysis of different intent-based solutions.
Cloud CDN Use Case: Caching Intent

When an intent developer wants to create a new intent, he has to decide how users can express their high-level targets using the Service-oriented declarative intent expression.

For example if `<TARGET>` is `<WORKLOAD>`,

The intent developer has to determine how this could be expressed by the CP either numerically such as “I want Caching for Content x to meet 10,000 requests/region”

or describe it with an adjective such as “I want Caching for Content x with the maximum requests/region”.
Caching Intent Expression Descriptors

**TABLE II. Basic expression of caching service-oriented intent**

<table>
<thead>
<tr>
<th>Tag</th>
<th>Basic Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;SERVICE&gt;</td>
<td>Caching</td>
</tr>
<tr>
<td>&lt;RESOURCES&gt;</td>
<td>contents to be cached</td>
</tr>
<tr>
<td>&lt;CONJUNCTION&gt;</td>
<td>that can handle / that can meet / etc.</td>
</tr>
</tbody>
</table>

**TABLE III. Basic expression syntax of caching service-oriented intent <TARGET> as <WORKLOAD>**

<table>
<thead>
<tr>
<th>Tag</th>
<th>Basic Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;WORKLOAD&gt;</td>
<td>&lt;NUMBER&gt; &lt;UNIT&gt; or &lt;ADJECTIVE&gt; &lt;UNIT&gt; or &lt;ADJECTIVE&gt; “workload”</td>
</tr>
</tbody>
</table>

**TABLE IV. Basic expression <WORKLOAD>**

<table>
<thead>
<tr>
<th>Tag</th>
<th>Basic Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;NUMBER&gt;</td>
<td>numeric values that can represent the workload</td>
</tr>
<tr>
<td>&lt;UNIT&gt;</td>
<td>requests/region / requests/sec / etc.</td>
</tr>
<tr>
<td>&lt;ADJECTIVE&gt;</td>
<td>max / min / dynamic / high / medium / etc.</td>
</tr>
</tbody>
</table>
Cloud CDN Use Case: **Caching** Intent

Then, the intent developer determines how to decompose the *declarative intent* expressed by the CP to a set of *abstract policies*

If a *new caching request* is received  
then *allocate cache servers* with *maximum storage capacity*  
*<sequential>*  
If *new user request* is received  
then *forward request to cache server* with *least latency*

<table>
<thead>
<tr>
<th>Tag</th>
<th>Basic Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>&lt;CONDITIONS&gt;</strong></td>
<td>new caching request / incoming user request / etc.</td>
</tr>
<tr>
<td><strong>&lt;ACTIONS&gt;</strong></td>
<td>allocate cache servers() / forward request() / etc.</td>
</tr>
<tr>
<td><strong>&lt;CONSTRAINTS&gt;</strong></td>
<td>with max storage / with least latency / etc.</td>
</tr>
<tr>
<td><strong>&lt;OPERATOR&gt;</strong></td>
<td>parallel / sequential</td>
</tr>
</tbody>
</table>
## Intent Refinement

<table>
<thead>
<tr>
<th></th>
<th><strong>Abstract Policies</strong></th>
<th><strong>Technical Policies</strong></th>
</tr>
</thead>
</table>
| **Initial Intent’s Policies** | - **Policy 1:**  
  *Condition:* if a new caching request has been received  
  *Action:* proactively allocate caches  
  *Constraints:* number of caches that can handle the average requested load  
  *Operator:* Sequential  
  
  - **Policy 2:**  
  *Condition:* if the caches’ thresholds have been exceeded  
  *Action:* scale out the cluster  
  *Constraints:* number of caches to be started  
  *Operator:* Parallel  
  
  - **Policy 3:**  
  *Condition:* if the caches are underutilized  
  *Action:* scale in the cluster  
  *Constraints:* number of caches to be stopped  |
| **Refined Intent’s Policies** | - **Policy 1:**  
  *Condition:* if the current day is a weekday AND the caches’ optimistic thresholds have been exceeded  
  *Action:* scale out the cluster  
  *Constraints:* at time $X_e$  
  *Operator:* Parallel  
  
  - **Policy 2:**  
  *Condition:* if the current day is a weekday AND the caches are underutilized  
  *Action:* scale in the cluster  
  *Constraints:* underutilization time > $X_e$  
  *Operator:* Parallel  
  
  - **Policy 3:**  
  *Condition:* if the current day is a weekend AND the caches’ pessimistic thresholds have been exceeded  
  *Action:* scale out the cluster  
  *Constraints:* at time $X_w$  
  *Operator:* Parallel  
  
  - **Policy 4:**  
  *Condition:* if the current day is a weekend AND the caches are underutilized  
  *Action:* scale in the cluster  
  *Constraints:* underutilization time > $X_w$  |
| **Initial Intent’s Policies** | - **Policy 1:**  
  *Condition:* if the current day is a weekday AND the caches’ CPU Utilization > 75% have been exceeded  
  *Action:* scale out the cluster  
  *Constraints:* at 14:45  
  *Operator:* Parallel  
  
  - **Policy 2:**  
  *Condition:* if the current day is a weekday AND the caches’ CPU Utilization < 20%  
  *Action:* scale in the cluster  
  *Constraints:* underutilization time > 20 minutes  
  *Operator:* Parallel  
  
  - **Policy 3:**  
  *Condition:* if the current day is a weekend AND the caches’ CPU Utilization > 70% have been exceeded  
  *Action:* scale out the cluster  
  *Constraints:* at 14:35  
  *Operator:* Parallel  
  
  - **Policy 4:**  
  *Condition:* if the current day is a weekend AND the caches’ CPU Utilization < 20%  
  *Action:* scale in the cluster  
  *Constraints:* underutilization time > 30 minutes |
Thank You