

---

MODULE *Topo*

---

The *Topo* module provides a formal specification of the *ONOS* topology service. The spec defines the client and server interfaces for *ONOS Topo* and provides helpers for managing and operating on connections.

CONSTANT *Nil*

VARIABLE *conns*

$gRPC \triangleq$  INSTANCE *gRPC* WITH  
      $OK \leftarrow \text{"OK"},$   
      $Error \leftarrow \text{"Error"}$

LOCAL INSTANCE *TLC*

$vars \triangleq \langle conns \rangle$

---

MODULE *Messages*

---

The *Messages* module defines predicates for receiving, sending, and verifying all the messages supported by *ONOS Topo*.

Message type constants

CONSTANT

$CreateRequest,$   
      $CreateResponse$

CONSTANTS

$UpdateRequest,$   
      $UpdateResponse$

CONSTANTS

$DeleteRequest,$   
      $DeleteResponse$

CONSTANT

$GetRequest,$   
      $GetResponse$

CONSTANT

$ListRequest,$   
      $ListResponse$

CONSTANT

$WatchRequest,$   
      $WatchResponse$

LOCAL  $messageTypes \triangleq$

$\{ CreateRequest,$   
          $CreateResponse,$   
          $UpdateRequest,$   
          $UpdateResponse,$   
          $DeleteRequest,$   
          $DeleteResponse,$

*GetRequest*,  
*GetResponse*,  
*ListRequest*,  
*ListResponse*,  
*WatchRequest*,  
*WatchResponse*

Message types should be defined as strings to simplify debugging  
 ASSUME  $\forall m \in \text{messageTypes} : m \in \text{STRING}$

---

This section defines predicates for identifying *ONOS Topo* message types on the network.

$\text{IsCreateRequest}(m) \triangleq m.type = \text{CreateRequest}$   
 $\text{IsCreateResponse}(m) \triangleq m.type = \text{CreateResponse}$   
 $\text{IsUpdateRequest}(m) \triangleq m.type = \text{UpdateRequest}$   
 $\text{IsUpdateResponse}(m) \triangleq m.type = \text{UpdateResponse}$   
 $\text{IsDeleteRequest}(m) \triangleq m.type = \text{DeleteRequest}$   
 $\text{IsDeleteResponse}(m) \triangleq m.type = \text{DeleteResponse}$   
 $\text{IsGetRequest}(m) \triangleq m.type = \text{GetRequest}$   
 $\text{IsGetResponse}(m) \triangleq m.type = \text{GetResponse}$   
 $\text{IsListRequest}(m) \triangleq m.type = \text{ListRequest}$   
 $\text{IsListResponse}(m) \triangleq m.type = \text{ListResponse}$   
 $\text{IsWatchRequest}(m) \triangleq m.type = \text{WatchRequest}$   
 $\text{IsWatchResponse}(m) \triangleq m.type = \text{WatchResponse}$

---

This section defines predicates for validating *ONOS Topo* message contents. The predicates provide precise documentation on the *E2AP* message format and are used within the spec to verify that steps adhere to the *E2AP* protocol specification.

LOCAL  $\text{ValidCreateRequest}(m) \triangleq \text{TRUE}$   
 LOCAL  $\text{ValidCreateResponse}(m) \triangleq \text{TRUE}$   
 LOCAL  $\text{ValidUpdateRequest}(m) \triangleq \text{TRUE}$   
 LOCAL  $\text{ValidUpdateResponse}(m) \triangleq \text{TRUE}$   
 LOCAL  $\text{ValidDeleteRequest}(m) \triangleq \text{TRUE}$

```

LOCAL ValidDeleteResponse(m)  $\triangleq$  TRUE
LOCAL ValidGetRequest(m)  $\triangleq$  TRUE
LOCAL ValidGetResponse(m)  $\triangleq$  TRUE
LOCAL ValidListRequest(m)  $\triangleq$  TRUE
LOCAL ValidListResponse(m)  $\triangleq$  TRUE
LOCAL ValidWatchRequest(m)  $\triangleq$  TRUE
LOCAL ValidWatchResponse(m)  $\triangleq$  TRUE

```

---

This section defines operators for constructing *ONOS Topo* messages.

```

LOCAL SetType(m, t)  $\triangleq$  [m EXCEPT !.type = t]

WithCreateRequest(m)  $\triangleq$ 
  IF Assert(ValidCreateRequest(m), "Invalid CreateRequest")
  THEN SetType(m, CreateRequest)
  ELSE Nil

WithCreateResponse(m)  $\triangleq$ 
  IF Assert(ValidCreateResponse(m), "Invalid CreateResponse")
  THEN SetType(m, CreateResponse)
  ELSE Nil

WithUpdateRequest(m)  $\triangleq$ 
  IF Assert(ValidUpdateRequest(m), "Invalid UpdateRequest")
  THEN SetType(m, UpdateRequest)
  ELSE Nil

WithUpdateResponse(m)  $\triangleq$ 
  IF Assert(ValidUpdateResponse(m), "Invalid UpdateResponse")
  THEN SetType(m, UpdateResponse)
  ELSE Nil

WithDeleteRequest(m)  $\triangleq$ 
  IF Assert(ValidDeleteRequest(m), "Invalid DeleteRequest")
  THEN SetType(m, DeleteRequest)
  ELSE Nil

WithDeleteResponse(m)  $\triangleq$ 
  IF Assert(ValidDeleteResponse(m), "Invalid DeleteResponse")
  THEN SetType(m, DeleteResponse)
  ELSE Nil

WithGetRequest(m)  $\triangleq$ 

```

```

    IF Assert(ValidGetRequest(m), "Invalid GetRequest")
    THEN SetType(m, GetRequest)
    ELSE Nil

```

```

WithGetResponse(m)  $\triangleq$ 
    IF Assert(ValidGetResponse(m), "Invalid GetResponse")
    THEN SetType(m, GetResponse)
    ELSE Nil

```

```

WithListRequest(m)  $\triangleq$ 
    IF Assert(ValidListRequest(m), "Invalid ListRequest")
    THEN SetType(m, ListRequest)
    ELSE Nil

```

```

WithListResponse(m)  $\triangleq$ 
    IF Assert(ValidListResponse(m), "Invalid ListResponse")
    THEN SetType(m, ListResponse)
    ELSE Nil

```

```

WithWatchRequest(m)  $\triangleq$ 
    IF Assert(ValidWatchRequest(m), "Invalid WatchRequest")
    THEN SetType(m, WatchRequest)
    ELSE Nil

```

```

WithWatchResponse(m)  $\triangleq$ 
    IF Assert(ValidWatchResponse(m), "Invalid WatchResponse")
    THEN SetType(m, WatchResponse)
    ELSE Nil

```

---

The *Messages* module is instantiated locally to avoid access from outside the module.

```

LOCAL Messages  $\triangleq$  INSTANCE Messages WITH
    CreateRequest  $\leftarrow$  "CreateRequest",
    CreateResponse  $\leftarrow$  "CreateResponse",
    UpdateRequest  $\leftarrow$  "UpdateRequest",
    UpdateResponse  $\leftarrow$  "UpdateResponse",
    DeleteRequest  $\leftarrow$  "DeleteRequest",
    DeleteResponse  $\leftarrow$  "DeleteResponse",
    GetRequest  $\leftarrow$  "GetRequest",
    GetResponse  $\leftarrow$  "GetResponse",
    ListRequest  $\leftarrow$  "ListRequest",
    ListResponse  $\leftarrow$  "ListResponse",
    WatchRequest  $\leftarrow$  "WatchRequest",
    WatchResponse  $\leftarrow$  "WatchResponse"

```

MODULE *Client*

The *Client* module provides operators for managing and operating on *Topo* client connections and specifies the message types supported for the client.

MODULE *Requests*

This module provides message type operators for the message types that can be send by the *Topo* client.

$$\begin{aligned}
& \text{CreateRequest}(c, m) \triangleq \\
& \quad \wedge \text{gRPC!Client!Send}(c, \text{Messages!WithCreateRequest}(m)) \\
& \text{UpdateRequest}(c, m) \triangleq \\
& \quad \wedge \text{gRPC!Client!Send}(c, \text{Messages!WithUpdateRequest}(m)) \\
& \text{DeleteRequest}(c, m) \triangleq \\
& \quad \wedge \text{gRPC!Client!Send}(c, \text{Messages!WithDeleteRequest}(m)) \\
& \text{GetRequest}(c, m) \triangleq \\
& \quad \wedge \text{gRPC!Client!Send}(c, \text{Messages!WithGetRequest}(m)) \\
& \text{ListRequest}(c, m) \triangleq \\
& \quad \wedge \text{gRPC!Client!Send}(c, \text{Messages!WithListRequest}(m)) \\
& \text{WatchRequest}(c, m) \triangleq \\
& \quad \wedge \text{gRPC!Client!Send}(c, \text{Messages!WithWatchRequest}(m))
\end{aligned}$$

Instantiate the *Topo!Client!Send* module  
 $\text{Send} \triangleq \text{INSTANCE } \text{Requests}$

MODULE *Responses*

This module provides predicates for the types of messages that can be received by an *Topo* client.

$$\begin{aligned}
& \text{CreateResponse}(c, h(-, -)) \triangleq \\
& \quad \text{gRPC!Client!Handle}(c, \text{LAMBDA } x, m : \\
& \quad \quad \wedge \text{Messages!IsCreateResponse}(m) \\
& \quad \quad \wedge \text{gRPC!Client!Receive}(c) \\
& \quad \quad \wedge h(c, m)) \\
& \text{UpdateResponse}(c, h(-, -)) \triangleq \\
& \quad \text{gRPC!Client!Handle}(c, \text{LAMBDA } x, m : \\
& \quad \quad \wedge \text{Messages!IsUpdateResponse}(m) \\
& \quad \quad \wedge \text{gRPC!Client!Receive}(c) \\
& \quad \quad \wedge h(c, m)) \\
& \text{DeleteResponse}(c, h(-, -)) \triangleq \\
& \quad \text{gRPC!Client!Handle}(c, \text{LAMBDA } x, m :
\end{aligned}$$

$$\begin{aligned} & \wedge \text{Messages!IsDeleteResponse}(m) \\ & \wedge \text{gRPC!Client!Receive}(c) \\ & \wedge h(c, m) \end{aligned}$$

$$\begin{aligned} \text{GetResponse}(c, h(-, -)) & \triangleq \\ & \text{gRPC!Client!Handle}(c, \text{LAMBDA } x, m : \\ & \quad \wedge \text{Messages!IsGetResponse}(m) \\ & \quad \wedge \text{gRPC!Client!Receive}(c) \\ & \quad \wedge h(c, m)) \end{aligned}$$

$$\begin{aligned} \text{ListResponse}(c, h(-, -)) & \triangleq \\ & \text{gRPC!Client!Handle}(c, \text{LAMBDA } x, m : \\ & \quad \wedge \text{Messages!IsListResponse}(m) \\ & \quad \wedge \text{gRPC!Client!Receive}(c) \\ & \quad \wedge h(c, m)) \end{aligned}$$

$$\begin{aligned} \text{WatchResponse}(c, h(-, -)) & \triangleq \\ & \text{gRPC!Client!Handle}(c, \text{LAMBDA } x, m : \\ & \quad \wedge \text{Messages!IsWatchResponse}(m) \\ & \quad \wedge \text{gRPC!Client!Receive}(c) \\ & \quad \wedge h(c, m)) \end{aligned}$$


---

Instantiate the *Topo!Client!Receive* module

$$\text{Receive} \triangleq \text{INSTANCE Responses}$$

$$\text{Connect}(s, d) \triangleq \text{gRPC!Client!Connect}(s, d)$$

$$\text{Disconnect}(c) \triangleq \text{gRPC!Client!Disconnect}(c)$$


---

Provides operators for the *Topo* client

$$\text{Client} \triangleq \text{INSTANCE Client}$$


---

MODULE *Server*

The *Server* module provides operators for managing and operating on *Topo* servers and specifies the message types supported for the server.

---

MODULE *Responses*

This module provides message type operators for the message types that can be send by the *Topo* server.

$$\begin{aligned} \text{CreateResponse}(c, m) & \triangleq \\ & \wedge \text{gRPC!Server!Reply}(c, \text{Messages!WithCreateResponse}(m)) \end{aligned}$$

$$\begin{aligned} \text{UpdateResponse}(c, m) & \triangleq \\ & \wedge \text{gRPC!Server!Reply}(c, \text{Messages!WithUpdateResponse}(m)) \end{aligned}$$

$$\begin{aligned}
DeleteResponse(c, m) &\triangleq \\
&\wedge gRPC!Server!Reply(c, Messages!WithDeleteResponse(m)) \\
GetResponse(c, m) &\triangleq \\
&\wedge gRPC!Server!Reply(c, Messages!WithGetResponse(m)) \\
ListResponse(c, m) &\triangleq \\
&\wedge gRPC!Server!Reply(c, Messages!WithListResponse(m)) \\
WatchResponse(c, m) &\triangleq \\
&\wedge gRPC!Server!Reply(c, Messages!WithWatchResponse(m))
\end{aligned}$$

Instantiate the *Topo!Server!Send* module  
 $Send \triangleq \text{INSTANCE } Responses$

MODULE *Requests*

This module provides predicates for the types of messages that can be received by an *Topo* server.

$$\begin{aligned}
CreateRequest(c, h(-, -)) &\triangleq \\
&gRPC!Server!Handle(c, \text{LAMBDA } x, m : \\
&\quad \wedge Messages!IsCreateRequest(m) \\
&\quad \wedge gRPC!Server!Receive(c) \\
&\quad \wedge h(c, m)) \\
UpdateRequest(c, h(-, -)) &\triangleq \\
&gRPC!Server!Handle(c, \text{LAMBDA } x, m : \\
&\quad \wedge Messages!IsUpdateRequest(m) \\
&\quad \wedge gRPC!Server!Receive(c) \\
&\quad \wedge h(c, m)) \\
DeleteRequest(c, h(-, -)) &\triangleq \\
&gRPC!Server!Handle(c, \text{LAMBDA } x, m : \\
&\quad \wedge Messages!IsDeleteRequest(m) \\
&\quad \wedge gRPC!Server!Receive(c) \\
&\quad \wedge h(c, m)) \\
GetRequest(c, h(-, -)) &\triangleq \\
&gRPC!Server!Handle(c, \text{LAMBDA } x, m : \\
&\quad \wedge Messages!IsGetRequest(m) \\
&\quad \wedge gRPC!Server!Receive(c) \\
&\quad \wedge h(c, m)) \\
ListRequest(c, h(-, -)) &\triangleq \\
&gRPC!Server!Handle(c, \text{LAMBDA } x, m : \\
&\quad \wedge Messages!IsListRequest(m)
\end{aligned}$$

$$\wedge gRPC!Server!Receive(c) \\ \wedge h(c, m))$$

$$WatchRequest(c, h(-, -)) \triangleq \\ gRPC!Server!Handle(c, \text{LAMBDA } x, m : \\ \wedge Messages!IsWatchRequest(m) \\ \wedge gRPC!Server!Receive(c) \\ \wedge h(c, m))$$

Instantiate the *Topo!Server!Receive* module  
 $Receive \triangleq \text{INSTANCE } Requests$

Provides operators for the *Topo* server  
 $Server \triangleq \text{INSTANCE } Server$

The set of all open *Topo* connections  
 $Connections \triangleq gRPC!Connections$

$$Init \triangleq \\ \wedge gRPC!Init$$

$$Next \triangleq \\ \wedge gRPC!Next$$

\ \* Modification History  
 \ \* Last modified *Mon Sep 13 15:38:22 PDT 2021* by *jordanhalterman*  
 \ \* Created *Mon Sep 13 15:07:05 PDT 2021* by *jordanhalterman*