OPEN NETWORKING FOUNDATION

Mapping Cross Stratum Orchestration (CSO) to the SDN architecture

Issue 1
March 11, 2016

ONF TR-528

Abstract

This document describes how the cross stratum orchestration application can be mapped to the architecture of software defined networking (SDN).

OpenFlow™

ONF Document Type: TR (Technical Reference), non-normative, type 2
ONF Document Name: Mapping Cross Stratum Orchestration (CSO) to the SDN architecture

Disclaimer

Open Networking Foundation
2275 E. Bayshore Road, Suite 103, Palo Alto, CA 94303
www.opennetworking.org

Table of Contents

List of Figures

1 Introduction

1.1 Scope and Objectives

This document provides a mapping between the SDN architecture and the Cross Stratum Orchestration (CSO) application as described in the CSO session at the March 2016 Member Workdays meeting (minutes in onf2016.090).

1.2 Common Terms, Abbreviations and Definitions

None

2 Background

Version 1.1 of the SDN architecture in TR-521 [1] provides further definition and description of some concepts that are useful as we consider mapping the CSO application to the SDN architecture. These concepts are summarized below. Also of interest are the Intent based NBI describe in TR-523 [2] and the Transport API functional requirements described in TR-527 [3].

2.1 SDN controller relationships

Figure 2 of TR-521 [1] shows the core of the SDN architecture. The text in section 2 states:

> "SDN is modelled as a set of client-server relationships between SDN controllers and other entities that may themselves be SDN controllers. In its role as a server, an SDN controller may offer services to any number of clients, while an SDN controller acting as client may invoke services from any number of servers."

Note: The architecture is recursive and that the client of an SDN controller (i.e. the entity that requests a service) may be an application or another SDN controller. Because of the abstraction barrier provided by the interface the (server) SDN controller is not aware of the nature of its client. Similarly the client context presents an abstraction of the subset of the server resources that the client can use, i.e. the client cannot directly observe the resources.

2.1.1 Client and server contexts

TR-521 defines these contexts that exist within a SDN controller:

> **Client context**
>
> The conceptual component of a server that represents all information about a given client and is responsible for participation in active server-client management-control operations.
>
> **Server context**
>
> The conceptual component of a client that represents all information about a given server and is responsible for participation in active server-client management-control operations.

A SDN controller interacts with its client via a (client specific) client context. The client context presents the resources that are assigned to the client has and the shared context (e.g. name space).

Similarly the SDN controller interacts with its resources via a (server specific) server context which holds the resources that are provided by that server and the shared name space.

2.2 Essence of a Controller

Section 6.1 of TR-521 states:

> "The core function of an SDN controller is to continually adapt the state of its resources to serve its clients according to an optimization policy. The concept of state is interpreted broadly. The very existence of a particular resource is itself a state, as are the values of all of its attributes and its relationship to other resources, both within and beyond the controller's domain."

And:

> "Control is the process of establishing and maintaining a desired state. Open-loop control is of little value in the SDN context; feedback is the essence of control."

Note: In a simplistic view we could consider that a controller has two different "phases" of operation:

- Service change: During this phase the controller validates the service request and (if appropriate) configures the resources to support the requested service (or service change).
    - During this phase the controller may modify the configuration of the resources used by existing services to optimize resource utilization.
- Steady state: During this phase the controller monitors the service to ensure that the SLA is being met and makes any necessary adjustment to the resources.
    - During this phase the controller may modify the configuration of the resources to optimize when resources are added or removed.

2.3 Orchestration

Section 6.2 of TR-521 states:

> "In the sense of feedback control, orchestration is the defining characteristic of an SDN controller. Orchestration is the selection of resources to satisfy service demands in an optimal way, where the available resources, the service demands and the optimization criteria are all subject to change."

2.4 Service context

Note that the service context and the server context are different concepts.

Section 6.8 of TR-521 states:

> "A service context conceptually contains at least all of the attributes of a service as requested by the client, and server-specific information necessary to map service attributes into the realization of the service."

This information includes the information required to validate the client control interface and to identify the relevant data plane interfaces (type, location, name etc.).

2.5 Intent NBI

TR-523 "Intent NBI – Definition and Principles" defines the principles governing Intent Northbound Interfaces (Intent NBIs) between Application Plane systems and Controller Plane systems.
Section 2.1 "What is Intent" states

> "In order to simplify the interaction between network service-consuming applications or operators and the service delivery system (i.e., the network and its control apparatus) – while letting the controller provide the full scope of value-added functionality that it should provide – Application Plane-Controller Plane communications should be based on Intent. Intent expresses desired service outcomes solely in consuming application-relevant terms."

2.6 Transport API

TR-527 [3] "Functional Requirements for Transport API" proposes the usage of standard application program's interface (APIs) to the network control functions. It specifies the following transport network controller services:

- Topology Service
- Connectivity Service
- Path Computation Service
- Virtual Network Service
- Notification Service

Section 1.2 "Scope" states:

> "… it is assumed that access control and policy details are conveyed via a sideways/orthogonal interface. It is understood that all API requests would be subject to filtering and scoping based on the privileges assigned to the calling entity and these would be based on business contracts as well as security and organizational roles."

3 CSO

The essential functions of Cross Stratum Orchestration (CSO) are to:

- Accept a service request from a client[1], which can be satisfied by using resources from several different domains (each of which provide different types of resources). The request is normally expressed in terms understood by the client (which are typically not understood by the underlying resources).
- Split the client request into domain specific requests, expressed in terms understood by each domain.

[1] A client may be application, another orchestrator a SDN controller, etc.)

- Receive the responses from each domain, select the optimum set of response, coordinate service activation and notify the client.
- Monitor the ongoing service.

The CSO acts as the intelligent entity in a feedback loop that configures resources to satisfy a user request and continues to monitor the state of the resources to ensure that the service is delivered.

3.1 Simple CSO use case

A simple CSO use case is shown in Figure 3.1.

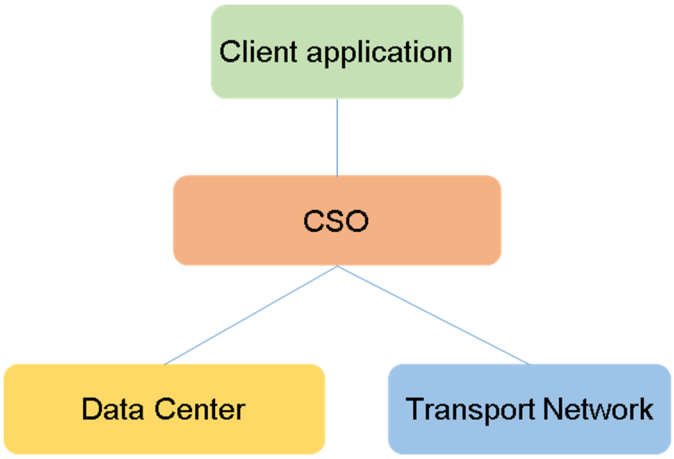

Figure 3.1: Simple CSO use case

In this use case the CSO has a control relationship to one data center and one transport network.

Figure 3.2 shows the mapping of the control interfaces in Figure 3.1 to the SDN architecture.

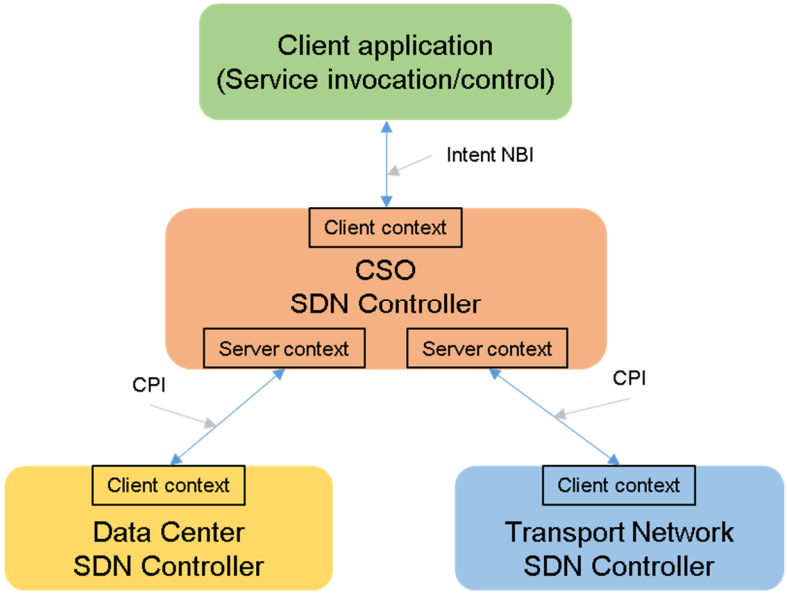

Figure 3.2: Mapping of the control interfaces to the SDN architecture

Note that the CPI between the CSO SDN controller and the Transport Network SDN controller should use the Transport API.

Figure 3.3 shows the relationship between the controllers and the data planes that they represent/manage.

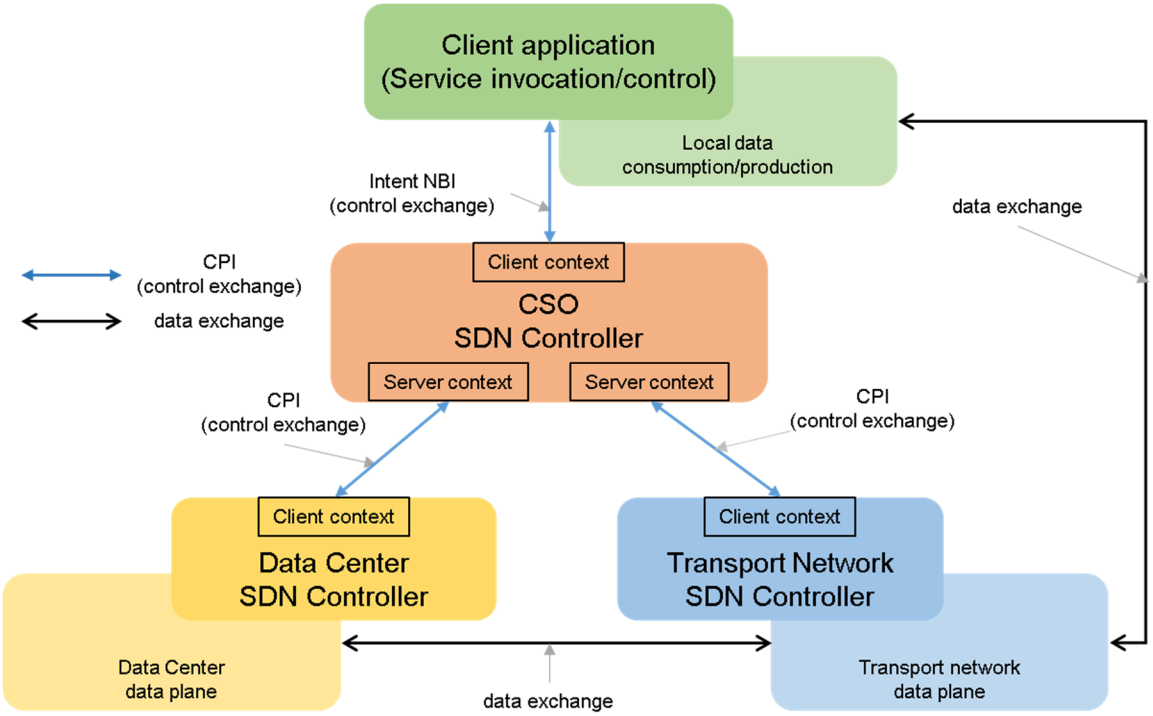

Figure 3.3: Relationship between the controllers and the data planes

The service context (described in section 2.4), for each control interface, identifies the resources that are provide and the data plane interfaces (including the protocol stack that is used for the data exchange). The data plane interfaces are named within the service context name space.

In this example the CSO SDN controller receives a request from the client application (in terms of the parameters understood by the application). This service request is mapped into a request for "compute service" to the Data Center SDN controller and a request for connectivity service to the Transport Network SDN Controller. These requests must identify the location of the data plane interfaces[2] that will be used to support the service. If both the data center and transport network controllers can provide the requested compute and transport services the CSO controller selects specific instances of the data plane interfaces (transport service end points), activates the service[3] and checks that the request was completed. The CSO controller then notifies the client (including the specific instance of the data plane interface). Both the Data Center controller and the Transport network controller monitor the activated service and notify the CSO controller of

[2] The transport network may provide more than one interface to both the client and the data center. Further the client and or data center may have more than one location. From the perspective of the transport network these are the service end points.
[3] Service activation also requires that the appropriate data plane labels (or timeslots) are identified.

any failures. If a failure is reported, the CSO controller may request an alternate service (to recover the client service) or just notify the client of the failure.

3.1.1 Interface abstraction

Because of the abstraction presented by the client context in each of the server controllers the details of the underlying resources are, at least to some extent, hidden from the CSO controller. The service context for each interface defines the level of visibility (or abstraction) provided to the CSO controller by the client context.

For example, the transport network SDN controller may be representing a set of transport networks. These transport networks may be in the same administration, or in different administrations. However these details may not be visible to the CSO controller.

Similarly the Data Center controller may be acting as a “broker” for several different data centers, possibly with different owners.

3.2 CSO with multiple server controllers

A different use case where the CSO has a direct control relationship with three data center controllers and three transport network controllers is shown in Figure 3.4.

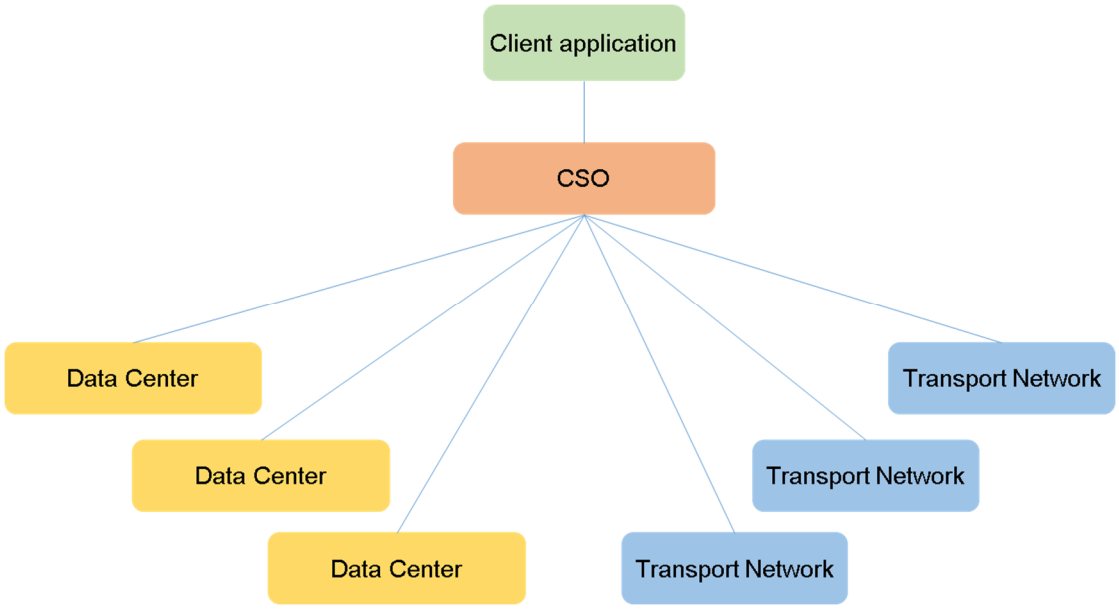

Figure 3.4: CSO with multiple server controllers

The mapping to the SDN architecture and the relationship between the data planes is shown in Figure 3.5.

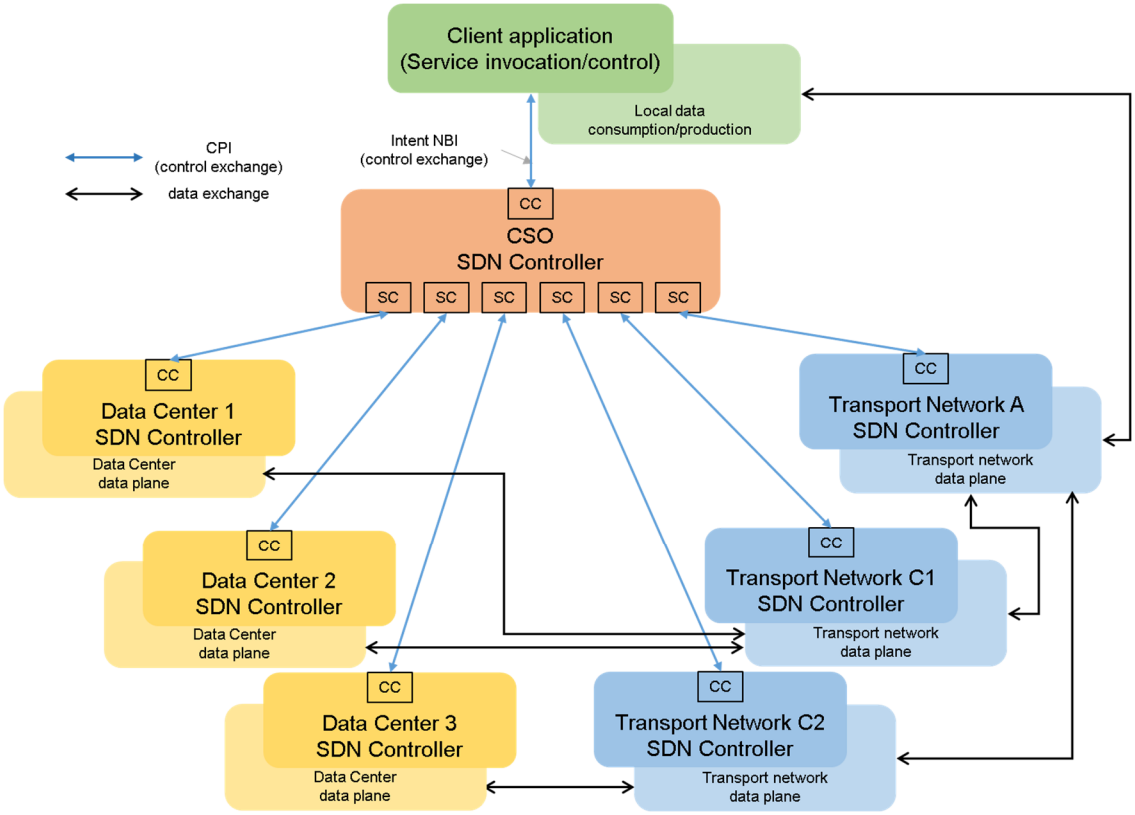

Figure 3.5: Mapping to the SDN architecture and the relationship between the data planes

Note: Each of the data centers and transport networks may be owned by different entities and will therefore may be in different administrative domains.

In this example use case the client only has data plane connectivity to transport network A. Transport network A has a data plane connectivity to Transport networks C1 and C2. Transport network C1 has data plane connectivity to Data centers 1 & 2. Transport network C2 has data plane connectivity to Data center 3. Typically this data plane connectivity is supported by more than one logical interface and may be provided at multiple locations.

In this scenario the CSO SDN controller receives a request from the client application (in terms of the parameters understood by the application). This is mapped into requests for “compute service” and “connectivity service”. The CSO controller may submit the “compute service” request to all three data centers. The connectivity service request must be further decomposed into requests to Transport networks A, C1 and C2. Note that all of the requests from the CSO controller must identify the location of the data plane interfaces and/or the networks that will be used. When the responses are received (note 1) the CSO controller selects the “optimum” responses, activates the appropriate services (note 2) checks that the activation was successful and notifies the client (note 3). The Data Center controller and the Transport network controllers monitor the activated service and notify the CSO controller of any failures. If a failure is reported

the CSO controller may request alternate services (to recover the client service) or just notify the client of the failure.

Note 1: These responses may include the identification of specific data plane interfaces with connection parameters (e.g. cost).

Note 2: The service activation request must include the identity of the selected data plane interfaces that are within the domain of that controller.

Note 3: This notification must include the identity of the data plane interfaces that are within the client domain.

3.3 CSO in different domains

A use case where the client has a relationship to the CSO in network domain A and network provider A has a relationship to the CSO in network domain B is shown in Figure 3.6.

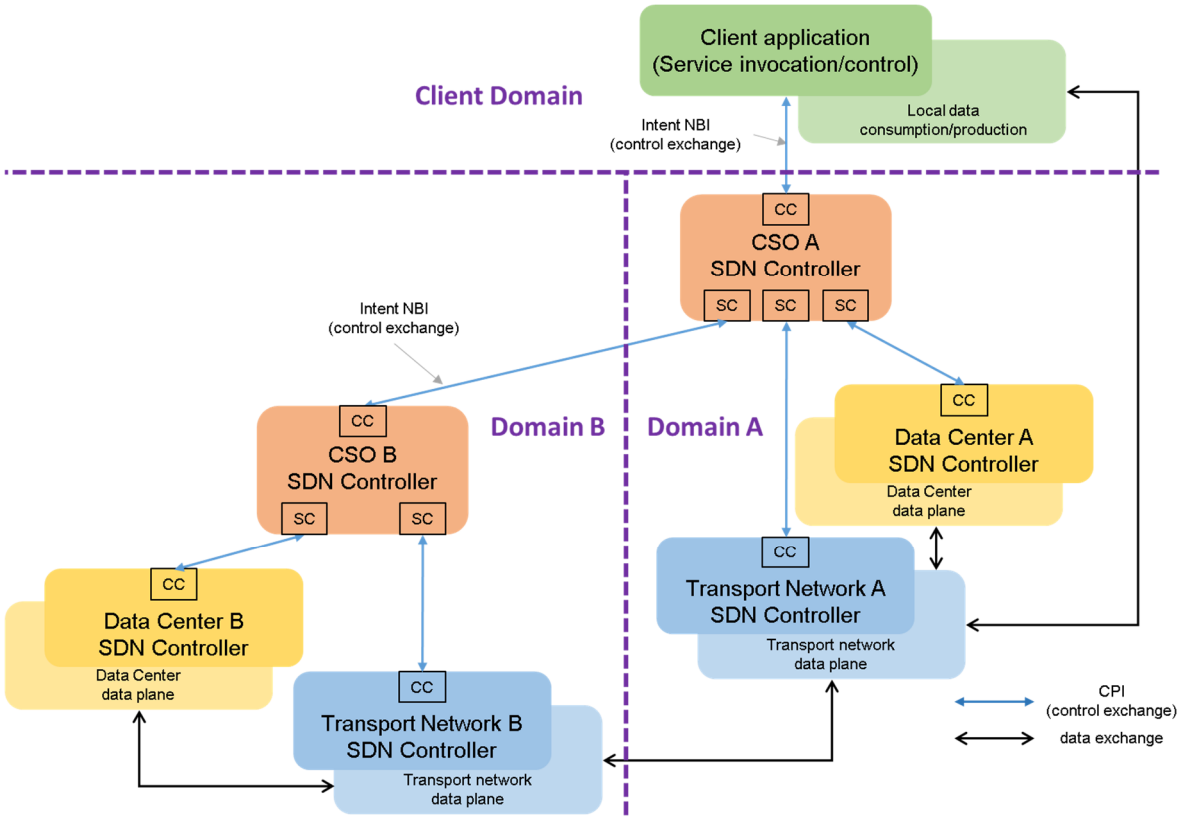

Figure 3.6: CSO in different network domains

In this example domain B only provides a CSO interface to domain A, the data centers and transport networks in domain B are not visible from domain A. From the perspective of the domain B CSO controller domain A appears to be a client application with local data consumption/production.

The CSO controller in domain A has direct relationships with of some transport networks and data centers. Only one of each is shown in Figure 3.6. However, as shown in Figure 3.5 there could be multiple transport networks and data centers and they may be in different domains.

In this scenario CSO A SDN controller receives a request from the client application (in terms of the parameters understood by the application). If appropriate this request is mapped into requests for “local” “compute service” and “connectivity service” which are passed to the “local” server controllers as described in 3.1 or 3.2 above. The client service request (or some subset of the request) may also be mapped to the intent NBI interface provided by the CSO B controller. The CSO B controller process this request as described in 3.1 or 3.2 above. Note that all of the requests from the CSO A controller must identify the location of the data plane interfaces and/or the networks that will be used. When the responses are received the CSO A controller selects the “optimum” responses, activates the appropriate services checks that the activation was successful and notifies the client. The Data Center A controller, Transport network A controller and CSO B controller monitor the activated service and notify the CSO A controller of any failures. If a failure is reported the CSO A controller may request alternate services (to recover the client service) or just notify the client of the failure.

A further possible scenario for multi domain CSO is that the client has a direct relationship to the CSO controller in more than one provider network. In this case none of the client application is responsible for the coordination of the service requests across the provider networks. The provider networks are not aware of this coordination.

4 References

[1] TR-521 SDN architecture version 1.1 March 2016
[2] TR-523 Intent NBI – Definition and Principles
[3] TR-527 Transport API: Functional Requirement

LIST OF CONTRIBUTORS

Malcolm Betts, ZTE

Dave Hood, Ericsson

Young Lee, Huawei

Ricard Vilalta, CTTC