UNIVERSITEIT VAN AMSTERDAM

SNE
System and Network Engineering

Translation Specification of OSPFv2 Traffic Engineering LSAs to NDL

Jeroen van der Ham
vdham@science.uva.nl

February 4, 2008

Abstract
This report is an extension of THE OSPF translation to NDL [1]. In this report we specify how we translate the topology information from Open Shortest Path First protocol version 2 Traffic Engineering (OSPF-TE)[2] Link State Announcements (LSAs) to the syntax of the Network Description Language (NDL) [3, 4].

SNE technical report SNE-UVA-2008-02
`http://www.science.uva.nl/research/sne/reports/`

1 Introduction

OSPFv2 has later been extended with traffic engineering options, called OSPF-TE [2]. Together with RSVP-TE, this formed the basis for an implementation of MPLS-TE [5]. The OSPF TE extension is implemented using a generic extension method of OSPFv2, the Opaque LSA [6]. There are three types of Opaque LSAs, defined by their flooding scope:

- Link-state type 9 denotes a link-local scope. Type 9 Opaque LSAs are not flooded beyond the local (sub)network.
- Link-state type 10 denotes an area-local scope. Type 10 Opaque LSAs are not flooded beyond the borders of their associated area.
- Link-state type 11 denotes that the LSA is flooded throughout the Autonomous System (AS). The flooding scope of type 11 LSAs are equivalent to the flooding scope of AS-external (type 5) LSAs.

In this report we will only describe type 10 (area-local) LSAs. Together with type 9, these are widely used in practice. Type 9 LSAs are only used for direct communication between adjacent routers, they do not add topological information. Currently, type 11 LSAs are not used.

We will first describe the format of type 10 LSAs, followed by what kind of information values are transported through them. Finally we will describe how to translate these values into NDL. The structure of the Opaque LSAs is shown in figure 1

The only difference with the regular OSPF header is that the `Link ID` field has been replaced with `Opaque Type` and `Opaque ID`. The `Opaque Type` is an 8-bit integer value to describe the type of the Opaque LSA, values 0-127 must be registered, and values 128-255 are available for experimental and private use. The `Opaque ID` is a 24-bit integer type-specific ID value.

For almost all TE extensions type 10 LSAs are used, with `Opaque Type` 1, the `Opaque ID` value is an arbitrary value used to maintain multiple TE LSAs. The body of these LSAs (`Opaque Information`) are structured using type-length-value elements (TLVs).

As the name suggests, a TLV consists of a `Type` and a `Length` field (both 16-bit integers), followed by a `Value` field of `Length` octets long. Note that the value of a TLV can also be other TLVs, these are then called `sub-TLVs`

Values often define a certain kind of bandwidth. These are all expressed in *bytes* per second using the standard IEEE floating point notation:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|S|    Exponent   |                  Fraction                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            LS age             |     Options   |   10 or 11 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Opaque Type  |               Opaque ID                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Advertising Router                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      LS Sequence Number                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         LS checksum           |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                      Opaque Information                       |
+                                                               +
|                              ...                              |
```

Figure 1: The structure of an Opaque LSA

`S` is the sign, `Exponent` is the exponent base 2 in 'excess 127' notation, and `Fraction` is the mantissa - 1, with an implied binary point in front of it. Thus, the above represents the value: $(-1)^{\texttt{S}} * 2^{\texttt{Exponent}-127} * (1 + \texttt{Fraction})$

2 Area-Local Opaque LSAs

In this section we describe the two currently defined TLVs for area-local opaque LSAs (type 10), and their sub-TLVs as defined by RFC 3630[2], and 4203[7].

1. `Router Address` This contains a stable IP address for advertising router, on which it can always be reached on the control plane.

2. `Link` This TLV describes a single link, and contains a set of sub-TLVs, described in the list below.

Below we list the first 10 sub-TLVs (1–9 and 11, 10 is not assigned). The sub-TLVs 14, 15, and 16 are described separately, because they are not that straightforward (12 and 13 are also unassigned).

1. `Link Type` (1 octet) The value is either `1` (point-to-point) or `2` (multi-access). This sub-TLV is mandatory.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Value...                           |
.                                                               .
.                                                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: The Type-Length-Value structure

2. `Link ID` (4 octets) An identifier for the other end of the link. For point-to-point it is the router ID of the neighbor, for multi-access it is the interface address of the designated router (i.e. the `Link State ID` of the Router LSA). This sub-TLV is mandatory.

3. `Local Interface IP Address` (4N octets) The address(es) of the local interface of this link, N is the number of addresses.

4. `Remote Interface IP Address` (4N octets) The address(es) of the remote interface of this link. If it is multi-access, the value is either `0.0.0.0` or the router may choose not to send this sub-TLV.

5. `Traffic Engineering Metric` (4 octets) The TE metric of the link, this may be different from the standard OSPF metric.

6. `Maximum Bandwidth` (4 octets) The true capacity of the link.

7. `Maximum Reservable Bandwidth` (4 octets) The maximum reservable capacity in this direction. This may be greater than the true capacity.

8. `Unreserved Bandwidth` (32 octets) The amount of bandwidth available for reservation in each of the eight priority levels, starting with 0. Each value must be less than or equal to the maximum reservable bandwidth.

9. `Administrative Group` (4 octets) A bit mask assigned by the administrator. Each set bit corresponds to a group that the interface belongs to. This starts at group 0, and is also called ‘Resource Class’ or ‘Color’.

11 `Link Local/Remote Identifier` (8 octets) GMPLS also supports unnumbered links, but these have to be identified in some way. This TLV contains the local and the remote identifiers for the endpoints of this unnumbered link.

An additional sub-TLV is the `Link Protection Type` sub-TLV (type 14). It has a length of 4 octets, but only the first of the octets is currently used. The meaning of the possible values of the first octet is as follows:

`0x01` *Extra Traffic*, this link is protecting another link or links, and LSPs on this link will be lost if any of these fail,

`0x02` *Unprotected*, there is no protection for this link, and LSPs will be lost if the link fails,

`0x04` *Shared*, there are one or more links of type *Extra Traffic* protecting this link, however, these are shared between one or more links of type *Shared*,

`0x08` *Dedicated 1:1*, there is one dedicated link of type *Extra Traffic* protecting this link,

`0x10` *Dedicated 1+1*, there is one dedicated link protecting this link, which is not advertised.

`0x20` *Enhanced*, this link is protected by a scheme better than *Dedicated 1+1*, for example by a 4 fiber ring BLSR.

Another sub-TLV that is relevant to link protection is the sub-TLV `Shared Risk Link Group` (type 16). This sub-TLV has a length of 4N octets, where N is the number of groups this link belongs to. A shared risk link group (SRLG) is a group of links that share a resource whose failure may affect all links in the group, for example two fibers running in the same conduit. An SRLG is identified by a 32 bit number, that is unique within the domain.

The purpose of the SRLG identification is to allow requests for multiple diversely routed LSPs, that also do not share any SRLGs, so as to minimize the risk of failure.

2.1 Translation to NDL

In this section we describe how we translate the information in Opaque LSAs to NDL. We use letters to denote objects. In NDL these objects are identified using URIs, these names are built up using the URI of the document or namespace, a pound sign (#), followed by the name of the object. Below we reference to names using only the latter part of the URI. We reference NDL properties using *italics*, the exact meaning of these properties can be found in the NDL papers [3, 4], or the NDL Homepage [8].

From the header of any Opaque LSA we learn:

- There is a device R, named `dev + Advertising Router`

- Router R *hasInterface* I named `Advertising Router`.

Depending on the type of TLV in the Opaque LSA we can learn additional information. For example from a type 1 TLV (Router Address) we can learn only one simple fact:

- Router R *hasInterface* I named $RouterAddress$.

On the other hand, a type 2 TLV carries a lot more information:

- Device R *hasInterface* I,
- If the value of `Link Type` is `1` (point-to-point):
  - There is a link L, named `Link ID`,
  - There is an interface I' *connectedTo* L,
  - Interface I is *connectedTo* L,
  - If the sub-TLVs `Local Interface IP Address` and `Remote Interface IP Address` are defined:
    - Interface I has the address(es) `Local Interface IP Address`, if the interface I was not named yet, then the first address is used as name,
    - Interface I' has the address(es) `Remote Interface IP Address`, if the interface I' was not named yet, then the first address is used as name,
  - Otherwise, the link is unnumbered, and the sub-TLV `Link Local/Remote Identifiers` must be present:
    - I is named `Link Local Identifier`,
    - I' is named `Link Remote Identifier`,
- If the value of `Link Type` is `2` (multi-access):
  - There is a broadcast segment BC named `bc + Link ID`,
  - Interface I is *connectedTo* link L,
  - Link L is `switchedTo` broadcast segment BC,
  - Interface I has the address `Local Interface IP Address`, if the interface I was not named yet, then the first address is used as name,
- Link L has a *metric* of `Traffic Engineering Metric`,
- Link L has a *capacity* of `Maximum Bandwidth`,
- Link L has a *protectionType* of `Link Protection Type`,

- Link L has a *sharedRiskGroup* property with value `Shared Risk Link Groups`.

Currently NDL does not yet support the concept of reservable and unreserved bandwidth. The reason for this is that the reservation information is more dynamic than the network topology. Our idea is that the best way to provide the user with up to date information is to have a pointer to a certain service, where the information about the reservable bandwidth of links can be obtained. We currently also do not translate the administrative groups. We currently have no experience what the value is used for in practice.

Note that it is possible to simplify this somewhat and use a single *connectedTo* statement between the two interfaces. However, it is then not possible to use the *protectionType* and *sharedRiskGroup* properties.

3 Switching Capability

The last sub-TLV that we describe is also the most complex; which is why we dedicate a separate section to it: the `Interface Switching Capability Descriptor` (ISCD)(type 15), which has a variable length. The purpose of this TLV is to describe the switching capabilities of both interface in that link of the advertising router, as well as the switching capabilities of the routers' switching matrix. The format of this sub-TLV is described in figure 3.

The values of the `Switching Capability` and `Encoding` fields are the same as used in the request signalling [9]. The `Switching Capability` (Switching Cap) field contains one of the following values:

1	– Packet-Switch Capable-1 (PSC-1),
2	– Packet-Switch Capable-2 (PSC-2),
3	– Packet-Switch Capable-3 (PSC-3),
4	– Packet-Switch Capable-4 (PSC-4),
51	– Layer-2 Switch Capable (L2SC),
100	– Time-Division-Multiplex Capable (TDM),
150	– Lambda-Switch Capable (LSC),
200	– Fiber-Switch Capable (FSC).

The four PSC values are used to express hierarchy of LSPs tunneled within LSPs.

The `Encoding` field is an integer field, where the value means that the link has the following encoding type:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Switching Cap |   Encoding    |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Max LSP Bandwidth at priority 0              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Max LSP Bandwidth at priority 1              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Max LSP Bandwidth at priority 2              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Max LSP Bandwidth at priority 3              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Max LSP Bandwidth at priority 4              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Max LSP Bandwidth at priority 5              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Max LSP Bandwidth at priority 6              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Max LSP Bandwidth at priority 7              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Switching Capability-specific information              |
|                  (variable)                                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: The structure of an Interface Switching Capability Descriptor

1 – Packet

2 – Ethernet

3 – ANSI/ETSI PDH

5 – SDH ITU-T G.707 / SONET ANSI T1.105

6 – Digital Wrapper

7 – Lambda (photonic)

8 – Fiber

9 – FiberChannel

For each of the eight priority levels, the sub-TLV gives the maximum bandwidth this link can support for LSPs. Contrary to the `Maximum Bandwidth` value, these values are designed to be dynamic. In the future these bandwidth specifications will replace the `Maximum Bandwidth` sub-TLV described earlier. For backward compatibility the `Maximum Bandwidth` value may be set to the priority 7 bandwidth.

The last section of the sub-TLV contains information specific to the switching capability value:

`Packet-Switch Capable` The specific information for PSC switching capabilities is structured as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Minimum LSP Bandwidth                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Interface MTU       |            Padding            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The `Minimum LSP Bandwidth` specifies the minimum bandwidth that an LSP must request. The supported LSP bandwidths depend on the encoding type. The `Interface MTU` specifies the largest size packets that are supported by this interface.

`Layer-2 Switch Capable` does not carry any extra information.

`Time-Division-Multiplex Capable` The specific information for TDM switching capabilities uses the following structure:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Minimum LSP Bandwidth                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Indication  |                 Padding                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The `Minimum LSP Bandwidth` specifies the minimum bandwidth that an LSP must request. The `Indication` contains an integer value used to indicate whether the interface supports Standard SONET/SDH (`0`), or Arbitrary SONET/SDH (`1`).

`Lambda-Switch Capable` does not carry any extra information.

The `Interface Switching Capability Descriptor` sub-TLV may occur multiple times, for example to express that an interface supports different encodings.

3.1 Translation to NDL

The `Interface Switching Capability Descriptor` is very complex to translate. The ISCD describes capabilities as part of the link, and the interface, while NDL describes this as part of the device. The interpretation of the values in the ISCD also depend on the values of the related ISCD of the interface on the other side of the link, which may have different values[1]. Therefore the translation below uses both the local ISCD (describing interface I, of router R) and the remote ISCD (interface I' and R'). The values of the remote ISCD are referenced with an accent (e.g. `Encoding'`).

It is possible that the two ISCDs of a link carry different encoding values for each side. This implicitly signals that one of the two interfaces is able to do an adaptation from its encoding to the (lower) other encoding. A lower encoding layer means a higher `Encoding` value.

NDL expresses adaptations using interface objects at different layers, therefore we introduce interfaces with a layer suffix, and define the relations between them. Once the common layer between the two interfaces is identified, we define an equality with the interfaces at that layer and the original interfaces, so that the connection is described on the right layer, with all adaptations.

The relation between `Encoding`, `Switching Capability` and NDL Layers is shown in table 3.1[2]

Switching Cap	**Encoding**	**NDL Layer**
Packet-[1-4]	Packet	IP
Layer-2	Ethernet	Ethernet
TDM	ANSI PDH SONET/SDH	TDM
–	Digital Wrapper	?
Lambda	Lambda	Lambda
Fiber	Fiber	Fiber
–	FiberChannel	?

Table 1: The relation between the different layer definitions.

The translation of these interfaces and layers is as follows:

- There is an *interface* $I_{\texttt{Encoding}}$, named `Local Interface + Encoding`,
- Interface $I_{\texttt{Encoding}}$ is at NDL layer `Encoding`,
- There is an *interface* $I'_{\texttt{Encoding'}}$, named `Local Interface' + Encoding'`,

[1]We assume that the link is bidirectional.

[2]The table shows a merry mixture of layer names, technologies, and protocols

- Interface $I'_{\texttt{Encoding}'}$ is at NDL layer `Encoding'`,

Next we compare the `Encoding` values and introduce the relevant interfaces at the right layers, and define the proper equalities:

- If `Encoding` > `Encoding'`, then:
  - There is an *interface* $I_{\texttt{Encoding}'}$, which is at NDL layer `Encoding'`,
  - There is an *adaptation* $Adap(\texttt{Encoding}, \texttt{Encoding}')$ between interface $I_{\texttt{Encoding}}$, and $I_{\texttt{Encoding}'}$,
  - Interface $I_{\texttt{Encoding}'}$ and I are defined to be equal,
  - Interface $I'_{\texttt{Encoding}'}$ and I' are defined to be equal,
- If `Encoding` < `Encoding'`, then:
  - There is an *interface* $I'_{\texttt{Encoding}}$, which is at NDL layer `Encoding`,
  - There is an *adaptation* $Adap(\texttt{Encoding}', \texttt{Encoding})$ between interface $I'_{\texttt{Encoding}'}$, and $I'_{\texttt{Encoding}}$,
  - Interface $I'_{\texttt{Encoding}}$ and I' are defined to be equal,
  - Interface $I_{\texttt{Encoding}}$ and I are defined to be equal,
- If `Encoding` = `Encoding'`, then:
  - There is an *interface* $I_{\texttt{Encoding}}$, which is at NDL layer `Encoding`,
  - There is an *interface* $I'_{\texttt{Encoding}}$, which is at NDL layer `Encoding`,
  - Interface I and $I_{\texttt{Encoding}}$ are defined to be equal,
  - Interface $I'_{\texttt{Encoding}}$ and I' are defined to be equal,

The actual switching is done by the device, these translations do not depend on the other ISCD, so we define them for the general case:

- There is a *switchMatrix SM*, which is at NDL layer `Switching Capability`,
- Device *R hasSwitchMatrix SM*,
- The switching-matrix *SM hasInterface* $I_{\texttt{SwitchingCapability}}$,
- Depending on the difference between the layers of `Encoding` and `Switching Capability`, introduce interfaces and adaptations as above, however, no equalities need to be defined.

NDL currently does not have a way to express any of the information in the switching capability-specific fields.

The exact adaptation functions used to go from one GMPLS layer to the other in NDL are currently still an open issue. OSPF-TE simply does not provide enough information to deduce the exact behavior of the devices.

References

[1] Jeroen van der Ham: *Translation Specification of OSPFv2 LSAs to NDL.* Technical Report UVA-SNE-2008-01, Unversiteit van Amsterdam (January 2007). URL `http://www.science.uva.nl/sne/reports/`. (document)

[2] D. Katz, K. Kompella, and D. Yeung: *Traffic Engineering (TE) Extensions to OSPF Version 2.* RFC 3630 (Proposed Standard) (September 2003). Updated by RFC 4203, URL `http://www.ietf.org/rfc/rfc3630.txt`. (document), 1, 2

[3] Jeroen van der Ham, Freek Dijkstra, Franco Travostino, Hubertus Andree, and Cees de Laat: *Using RDF to Describe Networks. Future Generation Computer Systems, Feature topic iGrid 2005* (2006). URL `http://staff.science.uva.nl/~vdham/research/publications/0510-NetworkDescriptionLanguage.pdf`. (document), 2.1

[4] Jeroen van der Ham, Paola Grosso, Ronald van der Pol, Andree Toonk, and Cees de Laat: *Using the Network Description Language in Optical Networks.* In *Tenth IFIP/IEEE Symposium on Integrated Network Management* (May 2007). URL `http://staff.science.uva.nl/~vdham/research/publications/0606-UsingNDLInOpticalNetworks.pdf`. (document), 2.1

[5] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus: *Requirements for Traffic Engineering Over MPLS.* RFC 2702 (Informational) (September 1999). URL `http://www.ietf.org/rfc/rfc2702.txt`. 1

[6] R. Coltun: *The OSPF Opaque LSA Option.* RFC 2370 (Proposed Standard) (July 1998). Updated by RFC 3630, URL `http://www.ietf.org/rfc/rfc2370.txt`. 1

[7] K. Kompella and Y. Rekhter: *OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS).* RFC 4203 (Proposed Standard) (October 2005). URL `http://www.ietf.org/rfc/rfc4203.txt`. 2

[8] Freek Dijkstra and Jeroen van der Ham: *Network Description Language Homepage.* 2.1

[9] L. Berger: *Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description.* RFC 3471 (Proposed Standard) (January 2003). Updated by RFCs 4201, 4328, 4872, URL `http://www.ietf.org/rfc/rfc3471.txt`. 3