        BBU Agregation: a usecase of the unified radio and optical control
                               architecture
                    draft-zhang-ccamp-bbu-aggregation-00

Abstract

   This document specifies a usecase, called BBU aggregation, for
   integreting radio and optical networks. It aims to compact several
   low-utilized BBU cards into one BBU to improve the BBU utilization.
   A flexible optical fronthaul network is connecting BBU and RRU to
   enable BBU aggregation by recongfiguring the lightpath between BBU
   and RRU. The procedure of the usecase is based on the unified radio
   and optical control archtecture, and an extended OpenFlow protocol is
   introduced to realize the procedure.

Status of This Memo

Copyright Notice

publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with respect
to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

Table of Contents

1. Introduction

   BBU aggregation is to compact some low-utilized BBUs into one BBU to
   improve the BBU utilization.In order to do that, it changes the
   connections of current RRU-BBU pairs, which is to reassociate the
   RRUs of low-utilized BBUs with one BBU, and shut down the low-
   utilized BBUs to save the cost. This requires a flexible optical
   fronthaul network, in which the lightpath between the BBU and RRUs
   can be reconfigured.

   To realize the BBU aggregation, radio and optical resources should be
   jointly allocated, and radio and optical network devises should to be
   simutaneusly controlled. This memo introduces a mechanism to
   aggregate low-utilized BBUs in a SDN-enabled radio and optical
   control architecture. The procedure of BBU aggregation contains the
   following steps: 1) radio controller (Radio-C) sends OF messages to
   RRU_A and BBU-A to inquire the current physical properties of radio
   networks, such as taffic load and BBU utilization. 2) RRU_A and BBU_A
   reply the requested information to the Radio-C. 3) after obtaining
   the reply message, the control plane will run BBU aggregation scheme
   to get new RRU-BBU pairs with the corresponding lightpath between
   them. 4) transport controller (Transport-C) sends OF messages to
   TN_As to establish the lightpaths between the new RRU-BBU pairs.

2. Requirements Language

   The key words are "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL
   NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in
   this document.

3. Terminology

   This memo uses the following terms: BBU_A, RRU_A, TN_A, Radio-C,
   Transport-C.

4. Motivation

   For real operational wireless networks, mobile subscribers have shown
   a strong time-geometry pattern, causing the base station utilization
   to fluctuate over time and area. For example, when users are moving
   to other areas, the BBU just stays in idle with a large amount of its
   processing power wasted.In addition, with the emerging of massive

small cells, dynamic RRU association is gaining more and more
attention to improve the energy saving and coordination efficiency.

To solve these problem, turning off the low-utilized BBUs and
migrating their RRUs to other active BBUs is an efficient method.
However, it is hard to realize dynamic RRU-BBU reassociation because
of the independent control of radio and optical networks.

Therefore, this memo give a usecase to realize the BBU aggregation by
joint allocation radio and optical resources in an unified control
plane.

5. Overview of BBU aggregation

An architectural protocol overview (the big picture of the protocol)
is provided in this section. Protocol details can be found in
further sections.

5.1. Problem

Because the BBU aggregation is realized by the lightpath
reconfiguration, a communication protocol between the BBU_A and RRU_A
is needed. The communication protocol is designed specifically for
communications between a BBU_A and a controller. A controller may
use the communication protocol to send a lightpath reconfiguration
request to a BBU_A, and the BBU_A may reply with a set of
reconfigured lightpaths if the lightpaths satisfying the set of
constraints.

5.2. New Messages

The communication protocol operates over TCP, which fulfills the
requirements for reliable messaging and flow control without further
protocol work.

This memo define the following new communication protocol messages
for BBU aggregation:

Controller Request Message for RRU Feature (RRU_Feature_Req): A
message sent by a Controller to a RRU to request RRU Feature which
contains RRU_ID and RRU_IP. A Controller MUST send RRU Feature
request message to a RRU at initialization phase to get the
information about traffic load,wavelength and corresponding BBUs.
The details of RRU_Feature_Req message is described in Section 6.1.

RRU Reply Message for RRU Feature (RRU_Feature_Rep): A message sent
by a RRU to a Controller to reply specific RRU_Feature_Req message,
which contains the features of the RRUs, such as traffic load and

corresponding BBUs. A RRU sends RRU Feature reply message if and
only if it received related RRU_Feature_Req message. The details of
RRU_Feature_Rep message is described in Section 6.2.

Controller Request Message for BBU Feature (BBU_Feature_Req): A
message sent by a Controller to a BBU_A to request BBU Feature which
contains BBU_ID and BBU_IP. A Controller MUST send RRU Feature
request message to a BBU_A at initialization phase to get the
information about traffic load,wavelength and BBU status. The
details of BBU_Feature_Req message is described in Section 6.3.

BBU Reply Message for BBU Feature (RRU_Feature_Rep): A message sent
by a BBU_A to a Controller to reply specific BBU_Feature_Req message,
which contains the features of the BBUs, such as traffic load and BBU
status. A BBU sends BBU Feature reply message if and only if it
received related BBU_Feature_Req message. The details of
BBU_Feature_Rep message is described in Section 6.4.

Controller Request Message for TN Feature (TN_Feature_Req): A message
sent by a Controller to a TN to request TN Feature which contains
node_ID and node_IP. A Controller MUST send TN Feature request
message to a TN at initialization phase to get the information about
port ,wavelength and switch status. The details of TN_Feature_Req
message is described in Section 6.5.

TN Reply Message for RRU Feature (TN_Feature_Rep): A message sent by
a TN to a Controller to reply specific TN_Feature_Req message, which
contains the features of the TNs, such as port ,wavelength and switch
status. A TN sends TN Feature reply message if and only if it
received related TN_Feature_Req message. The details of
TN_Feature_Rep message is described in Section 6.6.

Controller Request Message to BBU_A for Lightpath Reconfiguration
(BBU_A_Mod): A message sent by a Controller to a BBU_A to request
lightpath reconfiguration which contains BBU_ID, node_IP and BBU
status. A Controller MAY send lightpath reconfiguration request
message to a BBU_A at any time as long as it consideres this
operation necessary. The details of BBU_A_Mod message is described
in Section 6.7.

BBU_A Reply Message for Lightpath Reconfiguration (BBU_A_Rep): A
message sent by a BBU_A to a Controller to reply specific BBU_A_Mod
message, which contains the configuration results of BBU_As. A BBU_A
sends lightpath reconfiguration reply message if and only if it
received related BBU_A_Mod message. A BBU_A_Rep message can contain
either a set of reconfigurated lightpaths if the request can be
satisfied, or a negative reply if not. The negative reply may

indicate the reason why the lightpaths can not be reconfigurated.
The details of BBU_A_Rep message is described in Section 6.8.

Controller Request Message to TN_A for Lightpath Reconfiguration
(TN_A_Mod): A message sent by a Controller to a TN_A to request
lightpath reconfiguration which contains node_ID, node_IP , port,
wavelength and switch status. A Controller MAY send lightpath
reconfiguration request message to a TN_A at any time as long as it
consideres this operation necessary. The details of TN_A_Mod message
is described in Section 6.9.

TN_A Reply Message for Lightpath Reconfiguration (TN_A_Rep): A
message sent by a TN_A to a Controller to reply specific TN_A_Mod
message, which contains the configuration results of TN_As. A TN_A
sends lightpath reconfiguration reply message if and only if it
received related TN_A_Mod message. A TN_A_Rep message can contain
either a set of reconfigurated lightpaths if the request can be
satisfied, or a negative reply if not. The negative reply may
indicate the reason why the lightpaths can not be reconfigurated.
The details of TN_A_Rep message is described in Section 6.10.

5.3. Normal Communication Procedure

5.3.1. Initialization Phase

The initialization phase consists of three subphases and each
subphase two successive steps.

In the first subphase, the two steps are (described in a schematic
form in Figure 1:

1) Establishment of a TCP connection (3-way handshake) between the
RRU_A and the Controller.

2) Establishment of a session over the TCP connection.

```
                    +-+-+-+             +-+-+-+-+-+
                    |RRU_A|             |Controller|
                    +-+-+-+             +-+-+-+-+-+
                       |                    |
                       |                    |
                       |------------------->|
      (Initialize      |                    |
       Scop of         |   RRU_Feature_Req  |
       Attributes)     |<-------------------|
                       |                    |
                       |   RRU_Feature_Rep  |
                       |------------------->|
                       |                    |
```
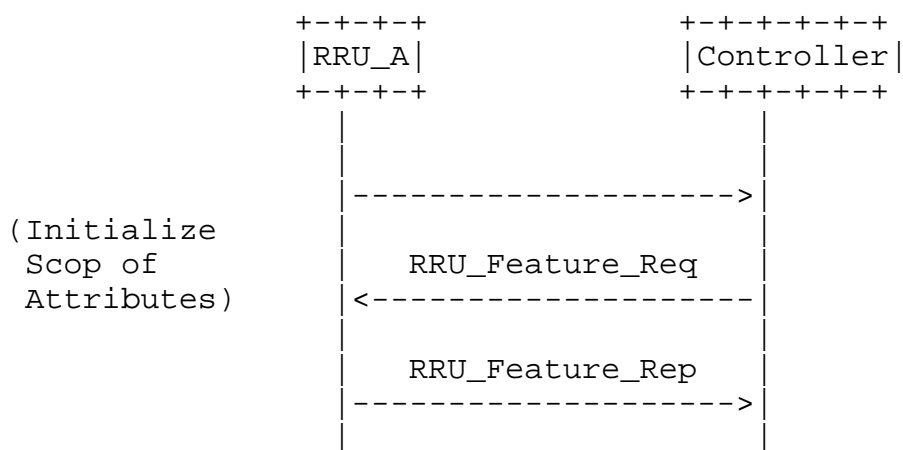
    Figure 1: Initialization Phase between the RRU_A and the Controller

Once the TCP connection is established, the Controller and the RRU_A
initiate session establishment during which various session
parameters are negotiated. The Controller sends a RRU Feature
request to the RRU (RRU_Feature_Req message).

Details about the RRU_Feature_Req message can be found in
Section 6.1.

After received the RRU_Feature_Req message, the RRU send a
RRU_Feature_Rep message including the features of the RRUs, such as
traffic load, corresponding BBUs and potentially other detailed
capabilities and policy rules that specify the conditions under which
path computation requests may be sent to the Controller.

Details about the RRU_Feature_Rep message can be found in
Section 6.2.

Similarly, in the second subphase, the two steps are (described in a
schematic form in Figure 2:

1) Establishment of a TCP connection (3-way handshake) between the
BBU_A and the Controller.

2) Establishment of a session over the TCP connection.

```
                +-+-+-+              +-+-+-+-+-+
                |BBU_A|              |Controller|
                +-+-+-+              +-+-+-+-+-+
                   |                     |
                   |                     |
                   |-------------------->|
 (Initialize       |                     |
  Scop of          |   BBU_Feature_Req   |
  Attributes)      |<--------------------|
                   |                     |
                   |   BBU_Feature_Rep   |
                   |-------------------->|
                   |                     |
```
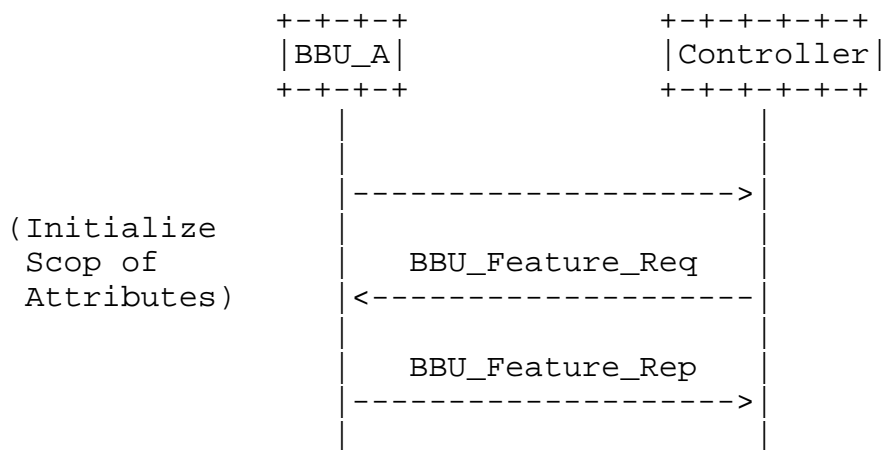
    Figure 2: Initialization Phase between the BBU_A and the Controller

Once the TCP connection is established, the Controller and the BBU_A
initiate session establishment during which various session
parameters are negotiated. The Controller sends a BBU_A Feature
request to the BBU_A (BBU_Feature_Req message).

Details about the BBU_Feature_Req message can be found in
Section 6.3.

After received the BBU_Feature_Req message, the BBU send a
BBU_Feature_Rep message including the features of the BBU_as, such as
traffic load, BBU_A status and potentially other detailed
capabilities and policy rules that specify the conditions under which
path computation requests may be sent to the Controller.

Details about the BBU_Feature_Rep message can be found in
Section 6.4.

Similarly, in the third subphase, the two steps are (described in a
schematic form in Figure 3:

1) Establishment of a TCP connection (3-way handshake) between the
TN_A and the Controller.

2) Establishment of a session over the TCP connection.

```
                +-+-+-+              +-+-+-+-+-+
                | TN_A|              |Controller|
                +-+-+-+              +-+-+-+-+-+
                   |                      |
                   |                      |
                   |--------------------->|
(Initialize        |                      |
 Scop of           |    TN_Feature_Req    |
 Attributes)       |<---------------------|
                   |                      |
                   |    TN_Feature_Rep    |
                   |--------------------->|
                   |                      |
```
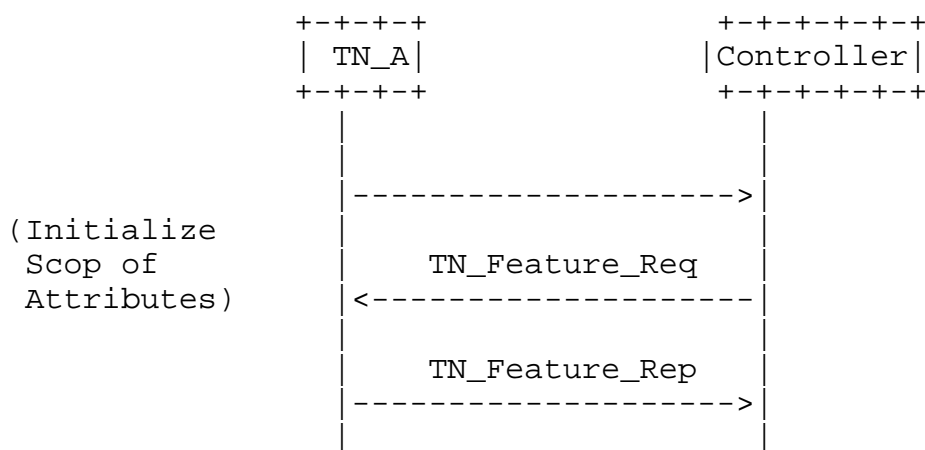
   Figure 3: Initialization Phasebetween the TN_A and the Controller

   Once the TCP connection is established, the Controller and the TN_A
   initiate session establishment during which various session
   parameters are negotiated. The Controller sends a RRU Feature
   request to the TN_A (TN_Feature_Req message).

   Details about the TN_Feature_Req message can be found in Section 6.5.

   After received the TN_Feature_Req message, the TN send a
   TN_Feature_Rep message including the features of the TNs, such as
   traffic load, corresponding TNs and potentially other detailed
   capabilities and policy rules that specify the conditions under which
   path computation requests may be sent to the Controller.

   Details about the TN_Feature_Rep message can be found in Section 6.6.

5.3.2. Lightpath Reconfiguration Request Sent by a Controller to a
        BBU_A

   Once a Controller has sucessfully established a session with one or
   more BBU_As, if a lightpath reconfiguration event is triggered that
   requires the reconfiguration of a set of lightpaths, the controller
   first selects one or more BBU_As.

   Once the Controller has selected a BBU_A, it sends a BBU_A_Mod
   message to the BBU_A. The process is shown in Figure 4.

   Details about the BBU_A_Mod message can be found in Section 6.7.

```
         +-+-+-+                    +-+-+-+-+-+
         |BBU_A|                    |Controller|
         +-+-+-+                    +-+-+-+-+-+
            |                           | 1)Lightpath Reconfiguration Event
            |                           | 2)Lightpath Reconfiguration Request
            |                           |    Sent to the BBU_A
            |<------BBU_A_Mod------|
            |                           |
            |                           |
            |                           |
```

               Figure 4: Lightpath Reconfiguration Request to BBU_A

5.3.3. Lightpath Reconfiguration Request Sent by a Controller to a TN_A

   Once a Controller has sucessfully established a session with one or
   more TN_As, if a lightpath reconfiguration event is triggered that
   requires the reconfiguration of a set of lightpaths, the controller
   first selects one or more TN_As.

   Once the Controller has selected a BBU_A, it sends a TN_Mod message
   to the TN_A. The process is shown in Figure 5.

   Details about the TN_Mod message can be found in Section 6.9.

```
         +-+-+-+                    +-+-+-+-+-+
         |TN_A |                    |Controller|
         +-+-+-+                    +-+-+-+-+-+
            |                           | 1)Lightpath Reconfiguration Event
            |                           | 2)Lightpath Reconfiguration Request
            |                           |    Sent to the TN_A
            |<------ TN_Mod -------|
            |                           |
            |                           |
            |                           |
```

               Figure 5: Lightpath Reconfiguration Request to TN_A

5.3.4. Lightpath Reconfiguration Reply Sent by a BBU_A to a Controller

   After receiving a lightpath reconfiguration request from a
   Controller, the BBU_A triggers a lightpath reconfiguration, If the
   BBU_A manages to reconfigure a lightpath satisfies the set of
   required constraints, the BBU_A returns the result to the requesting
   Controller. The process is shown in Figure 6.

```
                         +-+-+-+              +-+-+-+-+-+
                         |BBU_A|              |Controller|
                         +-+-+-+              +-+-+-+-+-+
                            |                     |
                            |                     |
                            |                     |
                            |<------BBU_A_Mod------|
        1) Lightpath Reconfiguration |                     |
           Request Received |                     |
        2) Reconfiguration Succefully |                     |
        3) Reconfiguration Result |                     |
           Sent to the Controller |                     |
                            |------BBU_A_Rep------>|
```

     Figure 6: Lightpath Reconfiguration Reply (Success) from BBU_A

   However, if no lightpath could be found that satisfies the set of
   constraints. In this case, a BBU_A may provide the set of
   constraints that led to the lightpath reconfiguration failure. Upon
   receiving a negative reply, a Controller may decide to resend a
   modified request or take any other appropriate action. The process
   is shown in Figure 7.

```
                         +-+-+-+              +-+-+-+-+-+
                         |BBU_A|              |Controller|
                         +-+-+-+              +-+-+-+-+-+
                            |                     |
                            |                     |
                            |                     |
                            |<-----BBU_A_Mod-------|
        1) Lightpath Reconfiguration |                     |
           Request Received |                     |
        2) econfiguration Unsuccefully|                     |
        3) Cause of Failure |                     |
           Sent to the Controller |                     |
                            |------BBU_A_Rep------>|
```

     Figure 7: Lightpath Reconfiguration Reply (Failure) from BBU_A

   Details about the BBU_A_Rep message can be found in Section 6.8.

5.3.5. Lightpath Reconfiguration Reply Sent by a TN_A to a Controller

   After receiving a lightpath reconfiguration request from a
   Controller, the TN_A triggers a lightpath reconfiguration, If the
   TN_A manages to reconfigure a lightpath satisfies the set of required
   constraints, the TN_A returns the result to the requesting
   Controller. The process is shown in Figure 8.

```
                          +-+-+-+                    +-+-+-+-+-+
                          |TN_A |                    |Controller|
                          +-+-+-+                    +-+-+-+-+-+
                             |                          |
                             |                          |
                             |                          |
                             |<---- TN_A_Mod -------|
         1) Lightpath Reconfiguration  |                |
            Request Received  |                          |
         2) Reconfiguration Succefully |                 |
         3) Reconfiguration Result     |                 |
            Sent to the Controller     |                 |
                             |                          |
                             |----- TN_A_Rep ------>|
```

            Figure 8: Lightpath Reconfiguration Reply (Success) from TN_A

   However, if no lightpath could be found that satisfies the set of
   constraints. In this case, a TN_A may provide the set of constraints
   that led to the lightpath reconfiguration failure. Upon receiving a
   negative reply, a Controller may decide to resend a modified request
   or take any other appropriate action.' The process is shown in
   Figure 9

```
                          +-+-+-+                    +-+-+-+-+-+
                          |TN_A |                    |Controller|
                          +-+-+-+                    +-+-+-+-+-+
                             |                          |
                             |                          |
                             |                          |
                             |<----- TN_A_Mod ------|
         1) Lightpath Reconfiguration  |                |
            Request Received  |                          |
         2) econfiguration Unsuccefully|                 |
         3) Cause of Failure           |                 |
            Sent to the Controller     |                 |
                             |------ TN_A_Rep ----->|
```

            Figure 9: Lightpath Reconfiguration Reply (Failure) from TN_A

   Details about the TN_A_Rep message can be found in Section 6.10.

6. the communication protocol Messages for BBU aggregation

   The communication protocol Messages for BBU aggregation consists of a
   common header followed by a variable-length body made of a set of
   objects. For each message type, rules are defined that specify the
   set of objects that the message can carry.

6.1. The RRU_Feature_Req message

```
<RRU_Feature_Req message> ::= <Common Header>
                                <RRU-information>
```

   Where:

```
<RRU_information> ::= <RRU-ID>
                       <RRU-IP>
```

6.2. The RRU_Feature_Rep message

```
<RRU_Feature_Rep Message> ::= <Common Header>
                                <RRU-feature-reply>
```

   Where:

```
<RRU-feature-reply> ::= <RRU-information>
                          <RRU-feature>
```

```
<RRU_information> ::= <RRU-ID>
                        <RRU-IP>
```

```
<RRU-feature> ::= <related-BBU-ID>
                    <traffic load>
                    <wavelength>
```

6.3. The BBU_Feature_Req message

```
<BBU_Feature_Req message> ::= <Common Header>
                                <BBU-information>
```

   Where:

```
<BBU_information> ::= <BBU-ID>
                        <BBU-IP>
```

6.4. The BBU_Feature_Rep message

```
<BBU_Feature_Rep Message> ::= <Common Header>
                             <BBU-feature-reply>
```
Where:

```
<BBU-feature-reply> ::= <BBU-information>
                        <BBU-feature>
```

```
<BBU_information> ::= <BBU-ID>
                      <BBU-IP>
```

```
<BBU-feature> ::= <BBU Status>
                  <traffic load>
                  <wavelength>
```

6.5.   The TN_Feature_Req message

```
<TN_Feature_Req message> ::= <Common Header>
                             <TN-information>
```
Where:

```
<RRU_information> ::= <node-ID>
                      <node-IP>
```

6.6.   The TN_Feature_Rep message

```
<TN_Feature_Rep Message> ::= <Common Header>
                             <TN-feature-reply>
```
Where:

```
<TN-feature-reply> ::= <TN-information>
                       <TN-feature>
```

```
<TN_information> ::= <node-ID>
                     <node-IP>
```

```
<TN-feature> ::= <port>
                 <switch-status>
                 <wavelength>
```

6.7.   The BBU_A_Mod message

```
<BBU_A_Mod message> ::= <Common Header>
                        <Controller-reconfiguration-request>
```
Where:
```
<Controller-reconfiguration-request> ::= <BBU-ID>
                                         <node-IP>
                                         <BBU-status>
```

6.8.   The BBU_A_Rep message

```
<BBU_A_Rep message> ::= <Common Header>
                        <Controller-reconfiguration-reply>
Where:
<Controller-reconfiguration-reply> ::= <BBU-ID>
                                       <node-IP>
                                       <BBU-config-reply>
```

6.9.   The TN_A_Mod message

```
<TN_A_Mod message> ::= <Common Header>
                       <Controller-reconfiguration-request>
Where:
<Controller-reconfiguration-request> ::= <node-ID>
                                         <node-IP>
                                         <TN-feature>

<TN-feature> ::= <port>
                 <switch-status>
                 <wavelength>
```

6.10. The TN_A_Rep message

```
<TN_A_Rep message> ::= <Common Header>
                       <Controller-reconfiguration-reply>
Where:
<Controller-reconfiguration-reply> ::= <node-ID>
                                       <node-IP>
                                       <TN-config-reply>
```

7. Object Formats

A object carried within a communication protocol messages for BBU aggregation, which consists of one or more 32-bit words with a common header.

7.1. Initialization Phase Object

7.1.1. RRU feature request TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      version    |     type=40    |        Message Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              xid                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            RRU_ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            RRU_IP                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
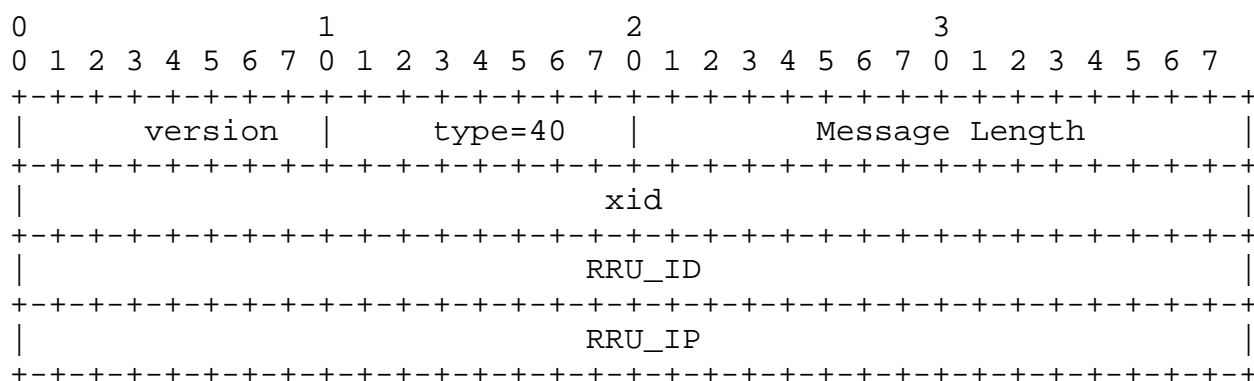
                 Figure 10: RRU feature request TLV format

   The common header consists of version, type and message length.

   Version (8 bits): The version number. Current version is version 1.

   Type (8 bits): A number indicates the message type. The
   "RRU_Feature_Req" message is the type 40.

   Message Length (16 bits): Total length of the message including the
   common header, expressed in bytes.

   The RRU_Feature_Req object body consists of the hardware id (xid),
   RRU id(RRU_ID) and RRU ip (RRU_IP).

7.1.2. BBU feature request TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      version    |     type=48    |        Message Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              xid                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            BBU_ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            BBU_IP                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                 Figure 11: BBU feature request TLV format

   The common header is similar with the RRU feature request object.
   The "BBU_Feature_Req" message is the type 48.

The BBU_Feature_Req object body consists of the hardware id (xid), BBU id(BBU_ID) and BBU ip (BBU_IP).

7.1.3. TN feature request TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       version |     type=41    |           Message Length      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              xid                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            node_ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            node_IP                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 12: TN feature request TLV format

The common header is similar with the RRU feature request object. The "TN_Feature_Req" message is the type 41.

The TN_Feature_Req object body consists of the hardware id (xid), node id(node_ID) and node ip (node_IP).

7.1.4. RRU feature reply TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       version |     type=42    |           Message Length      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              xid                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            RRU_ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            RRU_IP                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         target_BBU_ID                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         traffic load                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          wavelength                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 13: RRU feature reply TLV format

The common header is similar with the RRU feature request object.
The "RRU_Feature_Rep" message is the type 42.

The RRU_Feature_Rep object body consists of the hardware id (xid),
RRU id(RRU_ID), RRU ip (RRU_IP),corresponding BBUs, traffic load and
wavelength.

7.1.5. BBU feature reply TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      version  |    type=49    |          Message Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             xid                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            BBU_ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            BBU_IP                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          BBU_Status                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          traffic load                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          wavelength                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
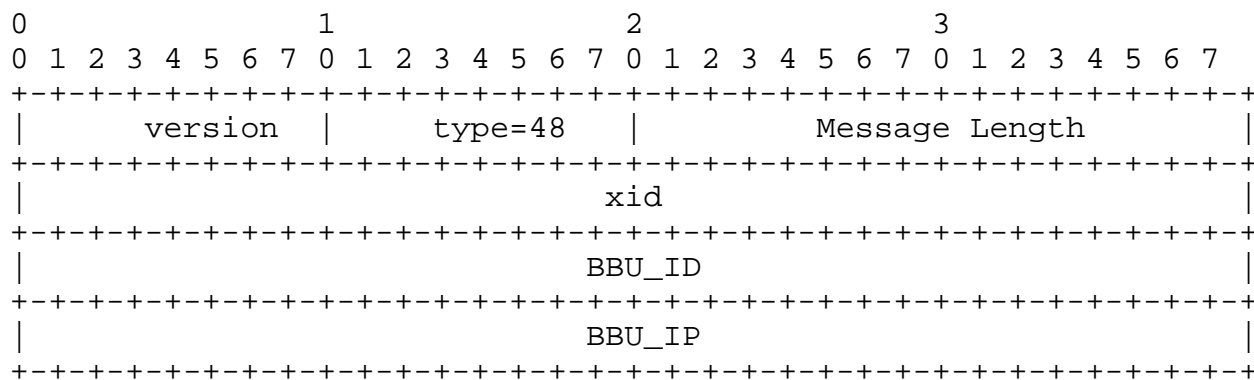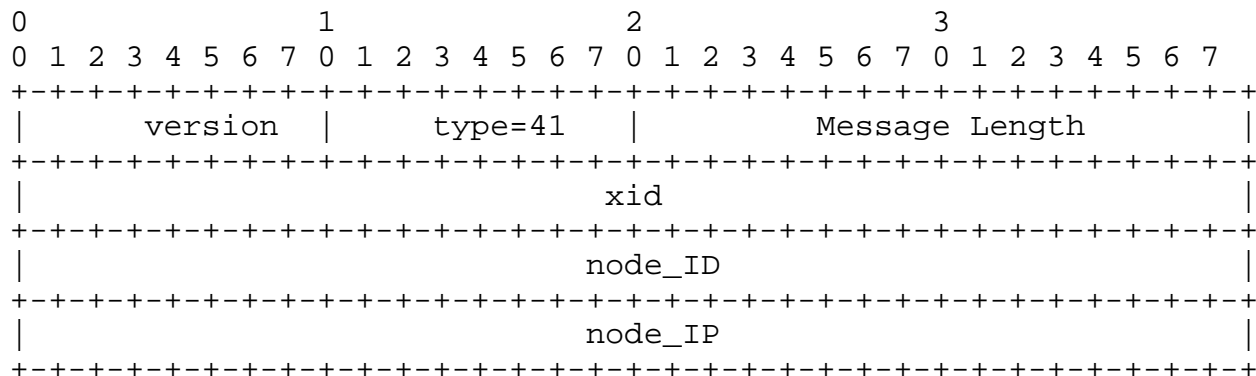
Figure 14: BBU feature reply TLV format

The common header is similar with the BBU feature request object.
The "BBU_Feature_Rep" message is the type 49.

The BBU_Feature_Rep object body consists of the hardware id (xid),
BBU id(BBU_ID), BBU ip (BBU_IP),BBU Status, traffic load and
wavelength.

7.1.6. TN feature reply TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      version    |    type=43      |        Message Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              xid                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            node_ID                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            node_IP                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             port                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         switch_status                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          wavelength                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
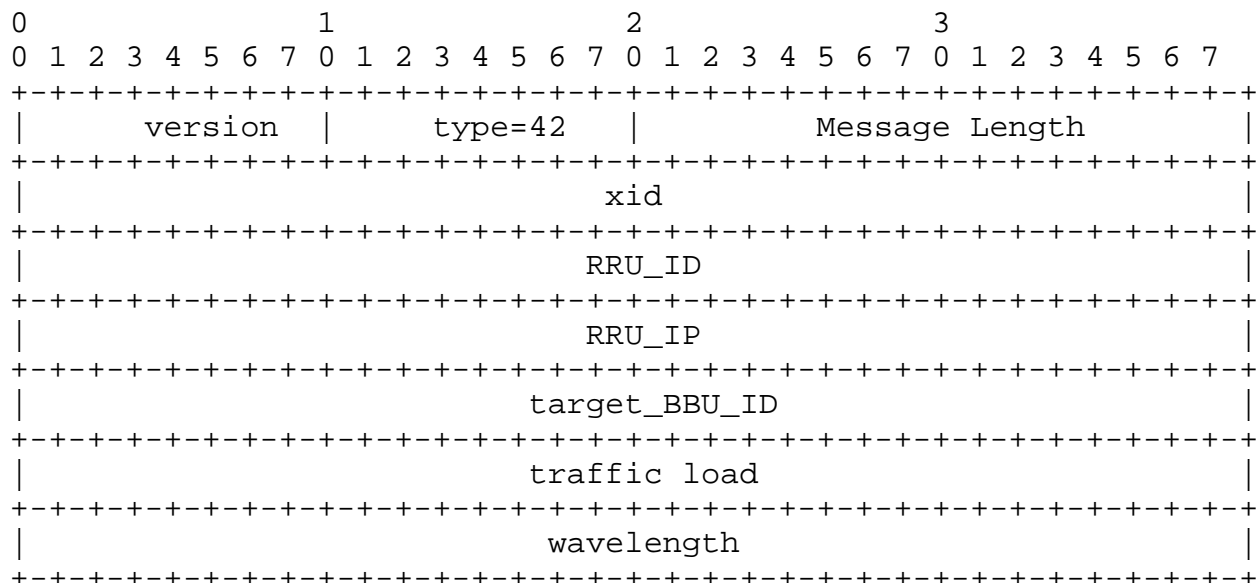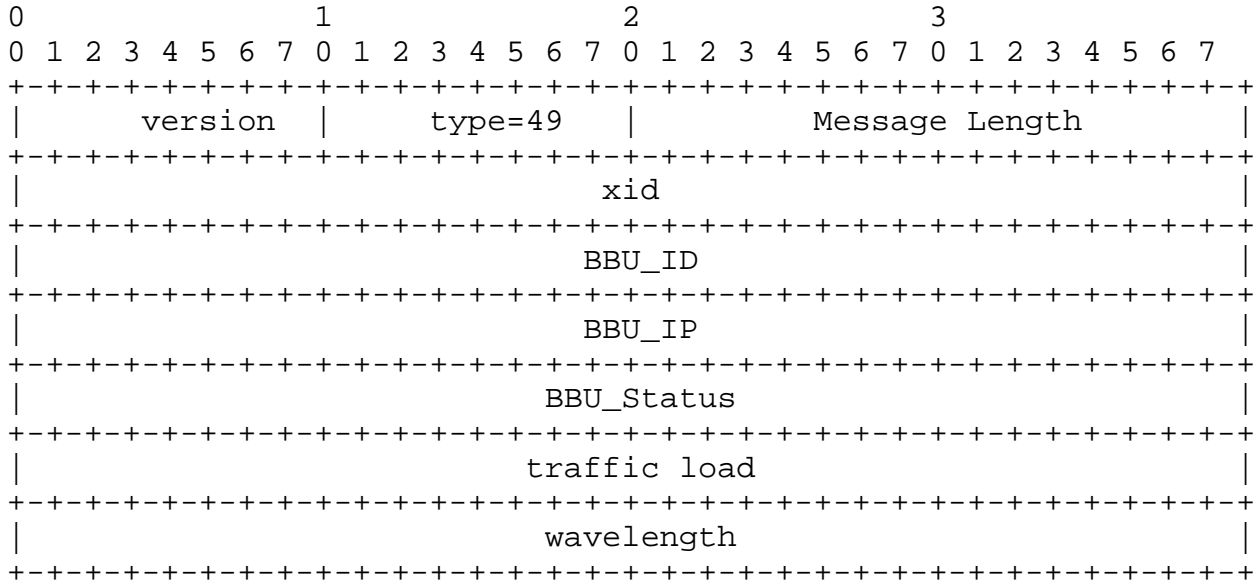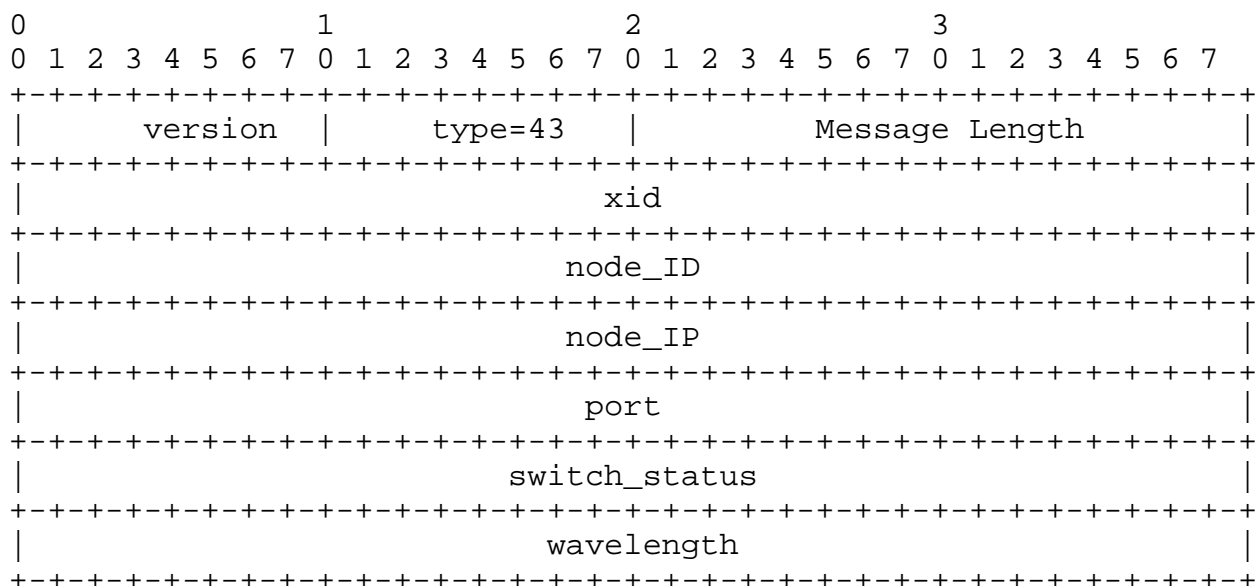
Figure 15: TN feature reply TLV format

The common header is similar with the RRU feature request object.
The "TN_Feature_Rep" message is the type 43.

The TN_Feature_Rep object body consists of the hardware id (xid),
node id(node_ID), node ip (node_IP),port, switch_status and
wavelength.

7.2. Lightpath Reconfiguration Phase Object

7.2.1. BBU modification request TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      version    |    type=44      |        Message Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              xid                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            BBU_ID                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            node_IP                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          BBU_status                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
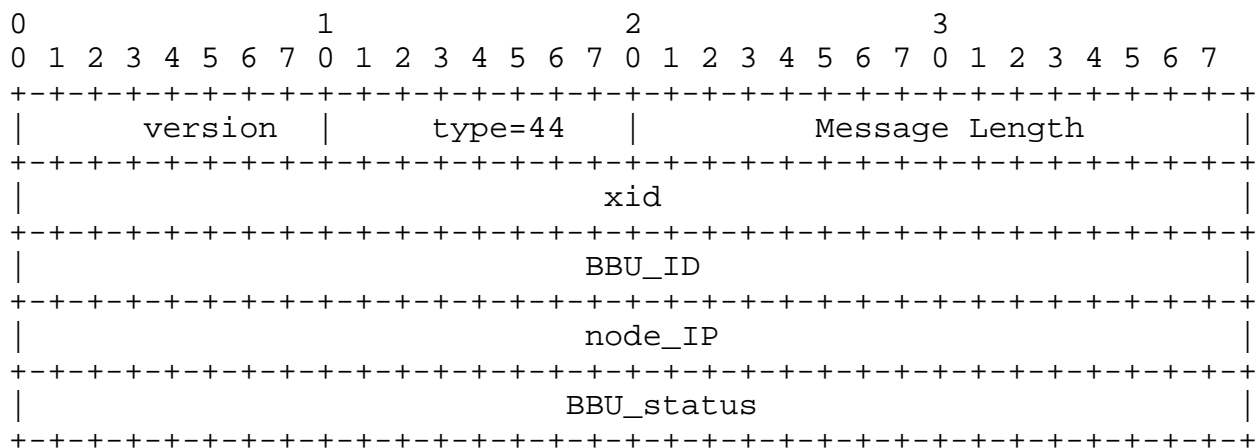
Figure 16: BBU modification request TLV format

The common header is similar with the RRU feature request object.
The "BBU_A_Mod" message is the type 44.

The BBU_A_Mod object body consists of the hardware id (xid), BBU
id(BBU_ID), node ip (node_IP) and BBU status(BBU_status).

7.2.2. TN modification request TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     version     |     type=45     |         Message Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              xid                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            node_ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            node_IP                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              port                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         switch_status                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          wavelength                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
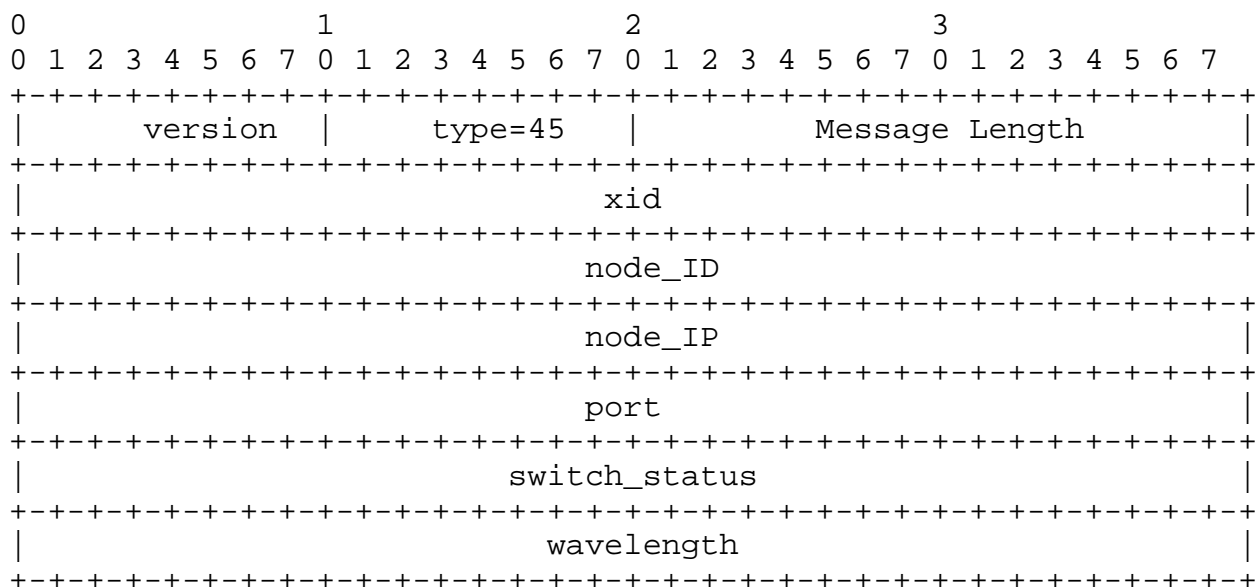
                Figure 17: TN modification request TLV format

The common header is similar with the RRU feature request object.
The "TN_A_Mod" message is the type 45.

The TN_A_Mod object body consists of the hardware id (xid), node
id(node_ID), node ip (node_IP), port, switch_status and wavelength.

7.2.3. BBU modification reply TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      version    |     type=46     |        Message Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            xid                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          BBU_ID                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          node_IP                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       BBU_config_reply                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
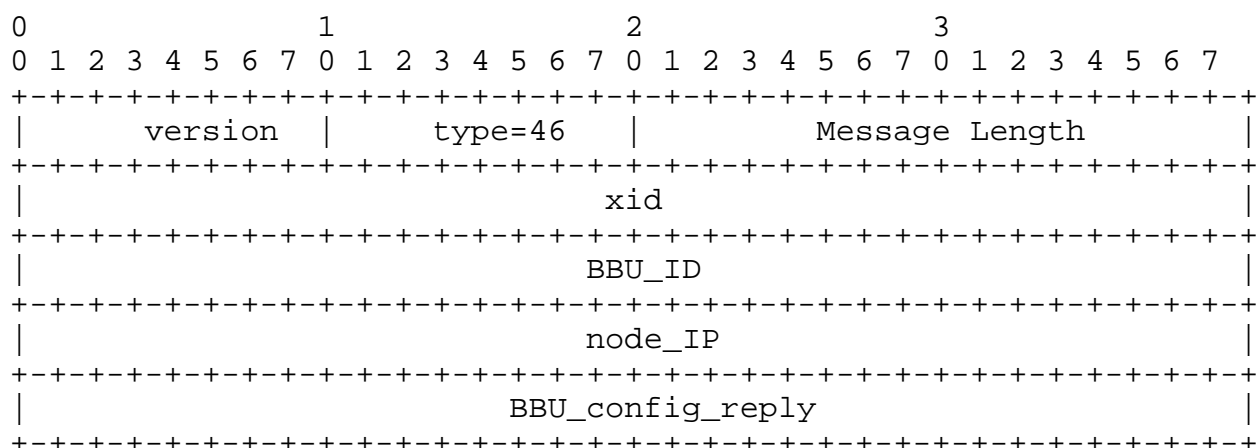
            Figure 18: BBU modification reply TLV format

   The common header is similar with the RRU feature request object.
   The "BBU_A_Rep" message is the type 46.

   The BBU_A_Mod object body consists of the hardware id (xid), BBU
   id(BBU_ID), node ip (node_IP) and BBU configuration
   reply(BBU_config_reply).

   The BBU_config_reply is used to report the result of BBU
   configuration. Two values are currently defined: "1" is a successful
   state while "0" is a failure state.

7.2.4. TN modification reply TLV

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      version    |     type=47     |        Message Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            xid                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          node_ID                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          node_IP                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       TN_config_reply                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
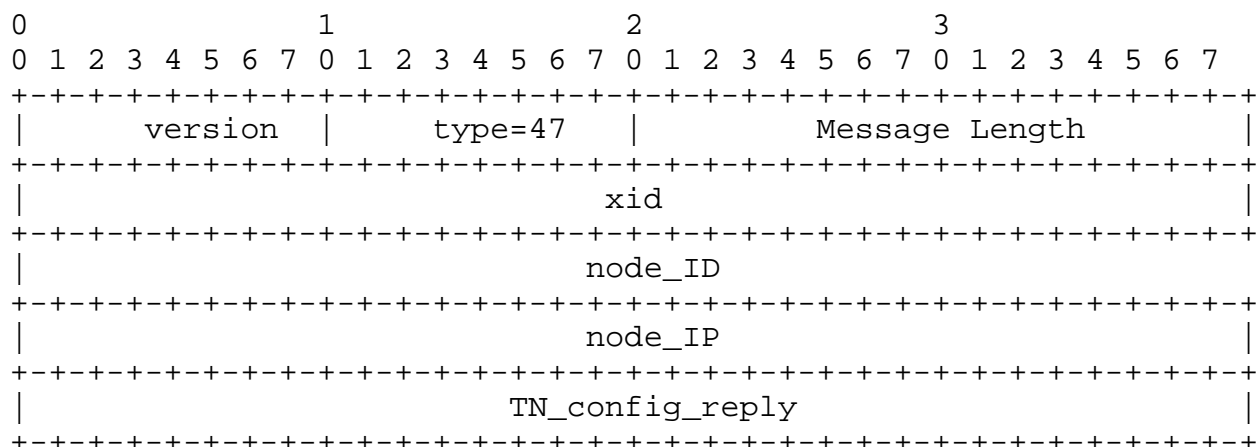
            Figure 19: TN modification reply TLV format

   The common header is similar with the RRU feature request object.
   The "TN_A_Rep" message is the type 47.

The TN_A_Mod object body consists of the hardware id (xid), node id(node_ID), node ip (node_IP) and TN configuration reply(TN_config_reply).

The TN_config_reply is used to report the result of TN configuration. Two values are currently defined: "1" is a successful state while "0" is a failure state.

8. Acknowledgments

9. Contributors

Authors' Addresses

    Jiawei Zhang (editor)
    Beijing University of Posts and Telecommunications
    Xitucheng Road
    Beijing, Haidian Dist 100876
    China

    Phone: +86-010-61198422
    Email: zjw@bupt.edu.cn

    Sainan Liu (editor)
    Beijing University of Posts and Telecommunications
    Xitucheng Road
    Beijing, Haidian Dist 100876
    China

    Phone: +86-010-61198422
    Email: lsn0429@bupt.edu.cn

    Zhen Liu (editor)
    Beijing University of Posts and Telecommunications
    Xitucheng Road
    Beijing, Haidian Dist 100876
    China

    Phone: +86-010-61198422
    Email: liuzhen207@bupt.edu.cn

      Yuefeng Ji (editor)
      Beijing University of Posts and Telecommunications
      Xitucheng Road
      Beijing, Haidian Dist 100876
      China

      Phone: +86-010-61198422
      Email: jyf@bupt.edu.cn