International Telecommunication Union

ITU-T Technical Paper

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(19 October 2018)

GSTP-NTSU

NT software upgrade for one image

Summary

Annex S of [ITU-T G.9701] specifies a procedure to upgrade the executable software of a Network Termination (NT). This procedure requires the support of two images. However, not all NTs have the capability to store two images. This Technical Paper describes a software upgrade procedure for NTs with one image only.

Keywords

Software upgrade, network termination, NT, single image

Change Log

This document contains Version 1 of the ITU-T Technical Paper on "NT software upgrade for one image" agreed at the ITU-T Study Group 15 meeting held in Geneva, 8 - 19 October 2018.

Editor:

Les Brown
Huawei Technologies
China

Tel: +1 (905) 826-4248
Cell: +1 (647) 290-1900
E-mail: lesbrown@sympatico.ca

Contents

Technical Paper ITU-T GSTP-NTSU

NT software upgrade for one image

1 Scope

Annex S of [ITU-T G.9701] provides means to upgrade the software of an NT where the executable software can be upgraded with a single vendor-specific image file. This software management process requires the support of two images, however, not all NTs have the capability to store two images. This Technical Paper describes a software upgrade procedure for NTs with one image only.

> NOTE – An upgrade with only one image has a higher risk that an upgrade may fail. Depending on the implementation, an unsuccessful upgrade could make the device non-functional. Therefore, the support of two software images and the usage of a software management process according to Annex S of [ITU-T G.9701] is strongly recommended.

The executable software may exist in multiple parts of the equipment (e.g., DSP firmware and higher layer application software). The contents of this software image file and the upgrade of individual components of an NT are not addressed by this Technical Report.

2 References

[ITU-T G.9701] Recommendation ITU-T G.9701 (2014), *Fast access to subscriber terminals (G.fast) – Physical layer specification*.

3 Definitions, abbreviations and acronyms

3.1 Definitions

This Technical Paper does not define any terms.

3.2 Abbreviations and acronyms

This Technical Paper uses the following abbreviations and acronyms:

DPU	Distribution Point Unit
ME	Management Entity
MIB	Management Information Base
NMS	Network Management System
NT	Network Termination
OAM	Operation, Administration and Maintenance

4 Overview

In cases where the NT is not capable of supporting two software images, the management process described in Annex S of [ITU-T G.9701] is not applicable. A modified procedure for the case where the NT can only store one instance of a software image is described in this Technical Paper. All actions are executed on image 0 only. The text in this Technical Paper describes the differences from the generic software management process for two images in Annex S of [ITU-T G.9701]. If not specified otherwise, the generic procedure is applied with the only exception that the usage is restricted to image 0 and all actions address image 0 only. All messages addressing image 1 are responded to with reason code 0101_2 “unknown managed software image instance” (see clause S.5 of [ITU-T G.9701]) if a response is required.

5 Reference model

The reference model for the NT software upgrade with one image is shown in Figure 1. It contains the associated OAM entities of the DPU and NT. The DPU entity contains the software image management function as a part of the ME-O. The ME-O will act as a master for the software image upgrades at the ME-R, which contains only one instance of a software image (image 0).

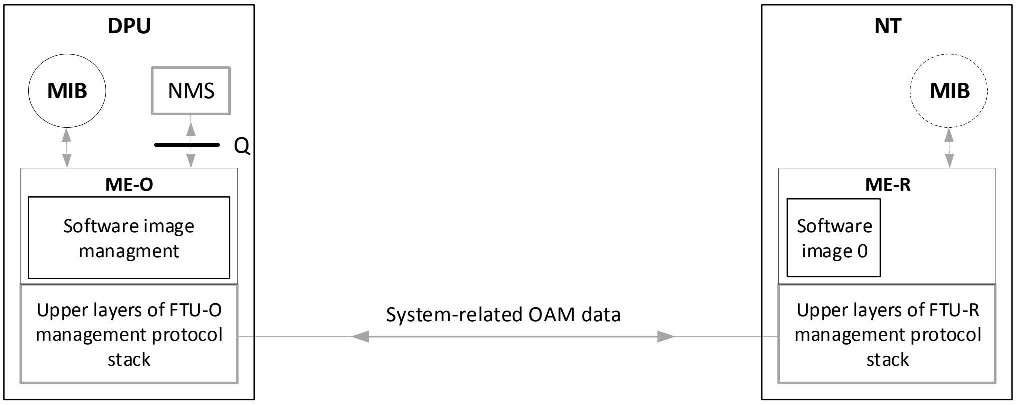

DPU
MIB
NMS
Q
ME-O
Software image managment
Upper layers of FTU-O management protocol stack
System-related OAM data
NT
MIB
ME-R
Software image 0
Upper layers of FTU-R management protocol stack

Figure 1 – Software image management reference model for one image

The status of the software image at the ME-R is reflected in the DPU-MIB. The ME-O and ME-R use a transparent communication channel to exchange software management commands and data between them as system-related OAM data. The following clauses of this Technical Paper describe the software image management process and the details of messages exchanged.

6 Software image management process

The ME-O manages executable software images stored in the NT (documented in clause S.4.1 of [ITU-T G.9701] as its fundamental usage).

6.1 Software image attributes for fundamental usage

The manageable attributes are as specified in Annex S of [ITU-T G.9701] with the following differences:

Managed software image: Since only one image is supported, the only valid value is 0.

Is committed: Since only one software image is supported by the NT, it is normally committed (1).

Is active: Since only one software image is supported by the NT, it is normally active (1). On the other hand, the software image could be inactive (0) after a successful software download while the new image has been stored in non-volatile memory and the old volatile code image is still executed.

6.2 Actions supporting the software upgrade process

All of the following actions are needed for NTs with remotely manageable software.

Get software image: Retrieve attributes of a software image instance.

Start software download: Initiate a software download sequence.

Download section: Download a section of a software image. This action is valid only for a software image instance that is currently being downloaded (state S2, state S2' as shown in Figure 2).

End software download: Signal the completion of a download image sequence, providing CRC for final verification. This action is valid only for a software image instance that is currently being downloaded (state S2, state S2' as shown in Figure 2).

Activate image: Load/execute a software image. When this action is applied to the software image, the image is stored in non-volatile memory if this has not been done at the end of the software download process. Then execution of the current code image is suspended, the associated software image is loaded from non-volatile memory, and execution of this new code image is initiated (that is, the associated entity reboots). Set the ***is active*** attribute value to 1 for the software image instance. This action is only valid if the software image is valid.

Commit image: Set the ***is committed*** attribute value to 1 for the software image instance. This causes the committed software image to be loaded and executed by the boot code upon subsequent start-ups. This action is only applicable when the software image is valid.

NOTE 1 – The software upgrade process using the above actions is exemplified in clause 9.

NOTE 2 – Since only one software image is supported by the NT, it is committed when the image is written to non-volatile memory and execution of this action is not required afterwards.

7 OAM data for software management

See clause S.4.3 of [ITU-T G.9701].

8 Message set

8.1 Get software image command

See section S.5.1 of [ITU-T G.9701].

8.2 Get software image response

See section S.5.2 of [ITU-T G.9701].

8.3 Start software download command

See section S.5.3 of [ITU-T G.9701].

8.4 Start software download response

See section S.5.4 of [ITU-T G.9701].

8.5 Download section command

See section S.5.5 of [ITU-T G.9701].

8.6 Download section response

See section S.5.6 of [ITU-T G.9701].

8.7 End software download command

See section S.5.7 of [ITU-T G.9701].

8.8 End software download response

See section S.5.8 of [ITU-T G.9701].

Implementations that are not able to write to the non-volatile memory at the end of the software download process respond with 8 (1000_2) in byte 8 if the CRC is correct.

8.9 Activate image command

See section S.5.9 of [ITU-T G.9701].

8.10 Activate image response

See section S.5.10 of [ITU-T G.9701].

8.11 Commit image command

See section S.5.11 of [ITU-T G.9701].

8.12 Commit image response

See section S.5.12 of [ITU-T G.9701].

9 Software upgrade

9.1 Overview

The software image management is described in clauses 6 and 7. The NT creates one software image instance (image 0). The image has three Boolean attributes: ***is valid*, *is committed***, and ***is active***. An image is valid if the contents have been verified to be an executable code image. The image is normally committed and will be loaded and executed upon reboot of the NT. An image is active if it is currently loaded and executing in the NT. It is inactive if a new image has been stored in non-volatile memory and the old volatile code image is still executed.

An NT goes through a series of states to download and activate a software image as shown in Figure 2. Each state is determined by the value of the attributes of the software image. For example, S3 is the state where a new image has been stored in non-volatile memory and the old volatile code image is still executed. Therefore, image 0 is committed but inactive. State S0 is a conceptual initialization state.

The value of the attributes (***is valid***, ***is active***, ***is committed***, ***version***) of the software image will be kept during the software download process until the software image is written to non-volatile memory.

If the device is able to do so, the image is written to non-volatile memory at the end of the software download process. If the action was successful, the NT sets a new version, marks image 0 as valid and committed but not activated. A reboot (e.g., a power cycle) of the NT would cause an activation of the new image and sets the ***is active*** attribute to 1. This behavior is shown in Figure 2.

Implementations that are not able to store the image to the non-volatile memory directly at the end of the software download process will write the image to the non-volatile memory as part of the activation process as shown in Figure 3. In this case, it is recommended to execute the activation directly after the end of the software download process because a reboot (e.g., a power cycle) of the NT will result in a loss of the downloaded image in the volatile memory.

The DPU controls the state of the NT through a series of commands and responses specified in clause 6. For example, upon reception of an activate image command with the managed software image instance field set to 0, an NT in state S3 will transition to state S4 after sending an activate image response with the message contents field set to 0000_2.

The specified commands and responses are start software download, download section, end software download, activate image and commit image.

NOTE 1 – By responding with the message contents field set to 0101_2 "unknown managed software image instance" to a command with the managed software instance set to 1, the DPU can detect if the device only supports one image.

NOTE 2 – The attribute ***is active*** set to 0 for image 0 will provide the information that the NT is still executing an old image in volatile memory and that the next reboot will activate the new image.

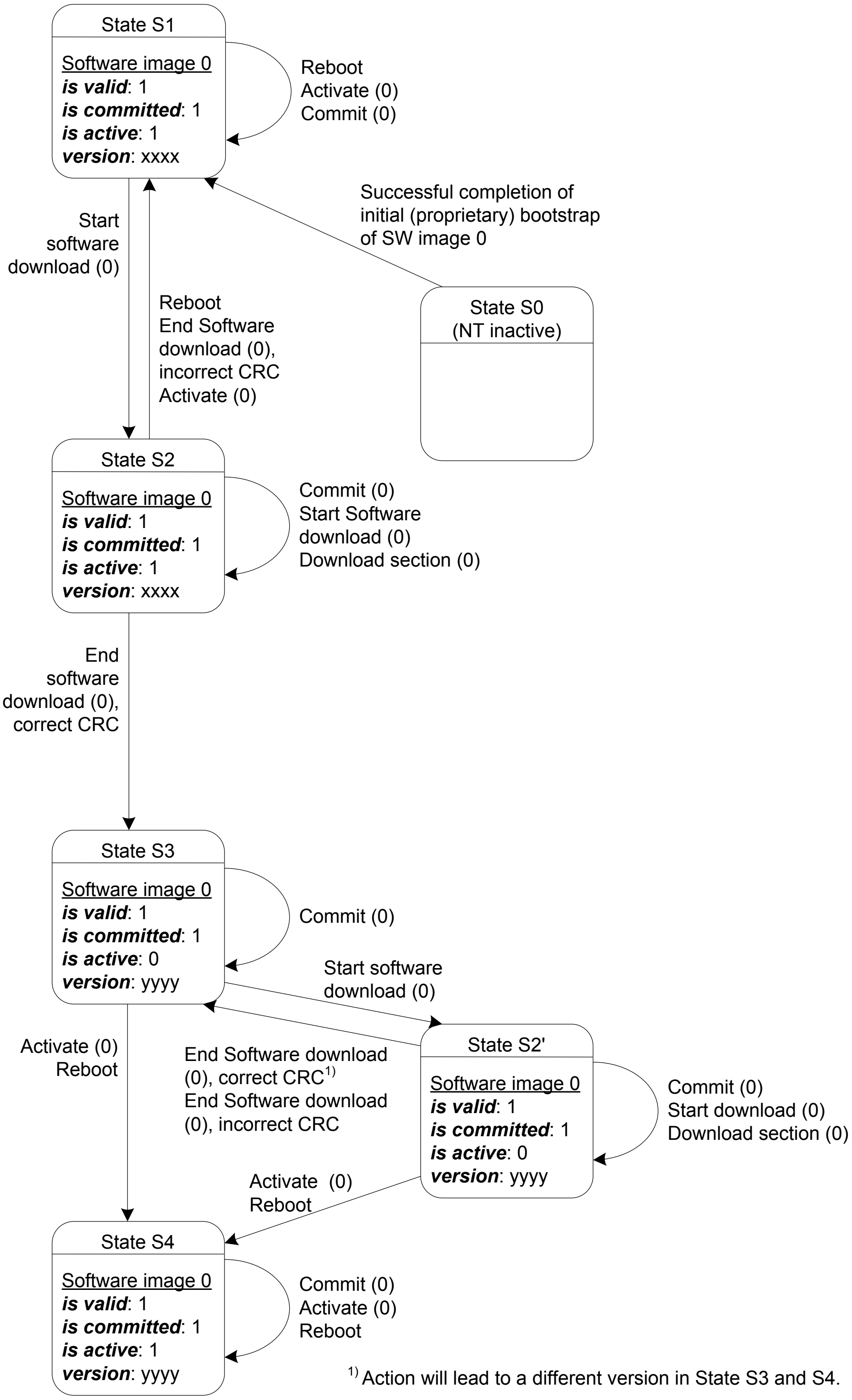
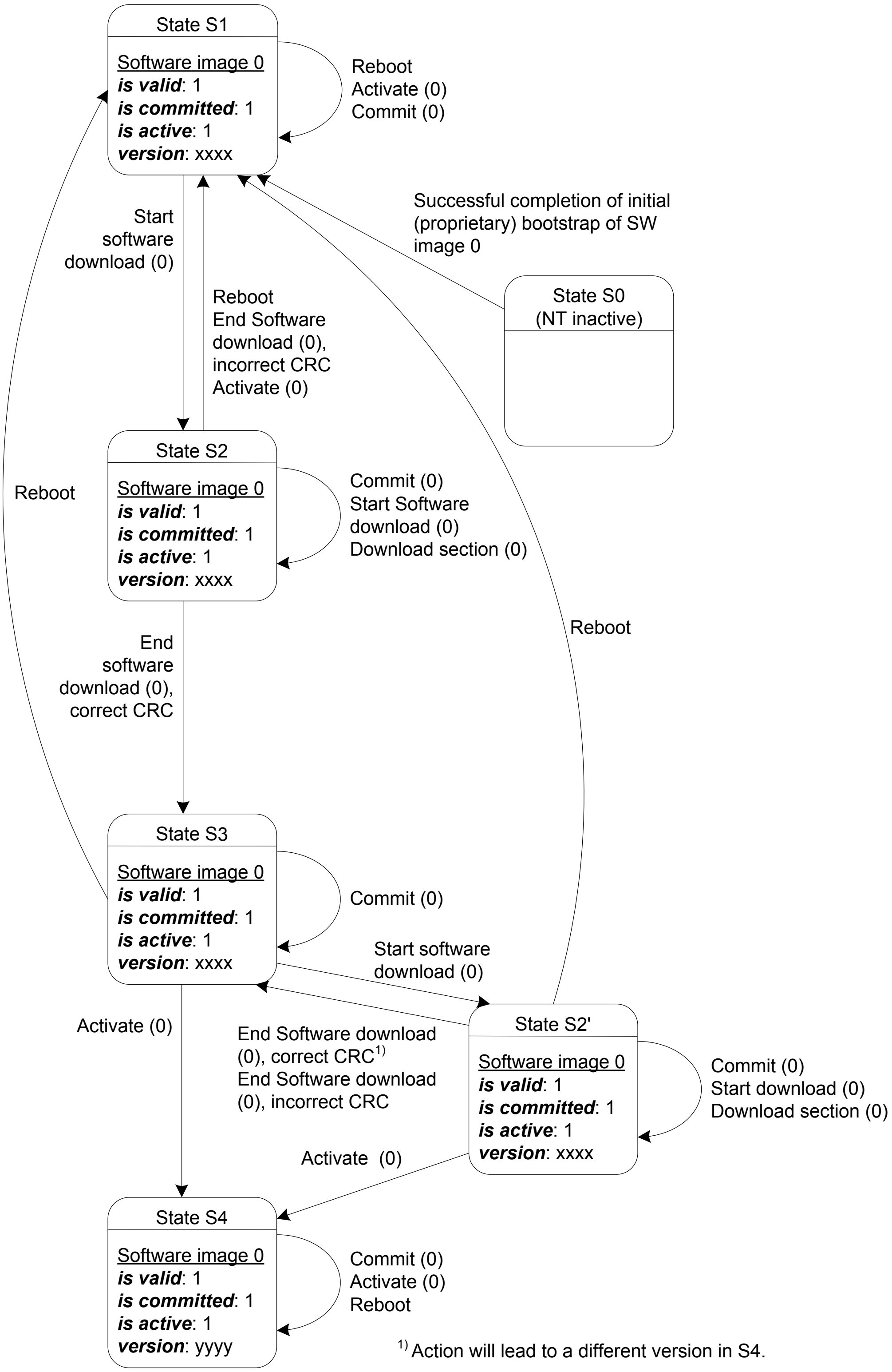


Figure 2 – Software image state diagram for the case when the image can be stored to the non-volatile memory at the end of the software download process

NOTE 3 – In Figure 2, states S1 and S2 are distinguished only for convenience in understanding the flow. Upon receipt of a start software download command, and particularly when the NT reboots, any partial downloads in progress are discarded.

NOTE 4 – In Figure 2, state transitions occur when any of the listed actions occur.



[1] Action will lead to a different version in S4.

Figure 3 – Software image state diagram for the case when the image can't be stored to the non-volatile memory at the end of the software download process

9.2 Software image download

The software image download process is described in clause S.6.2 of [ITU-T G.9701].

Figure 4 illustrates this process for one software image.

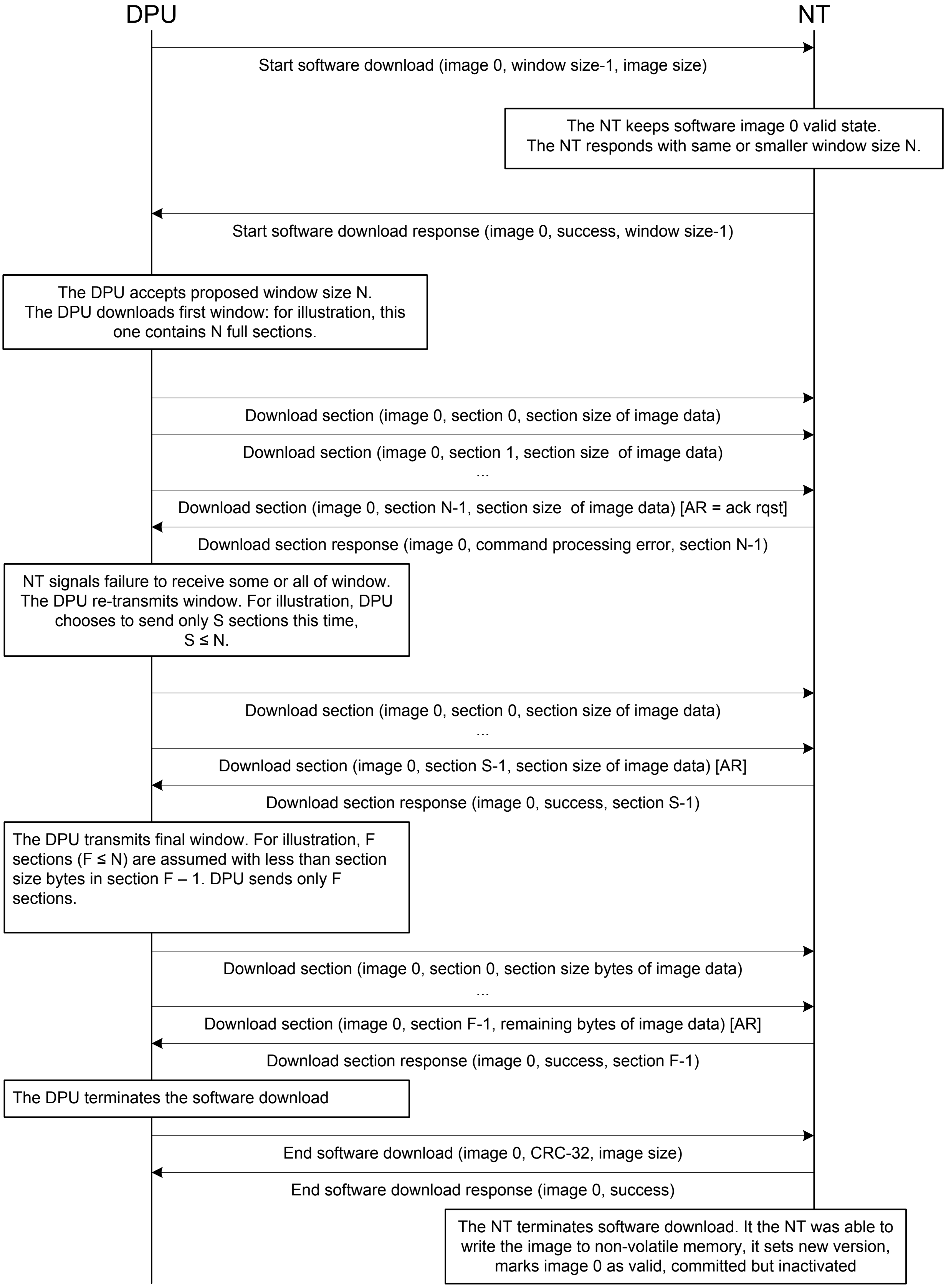
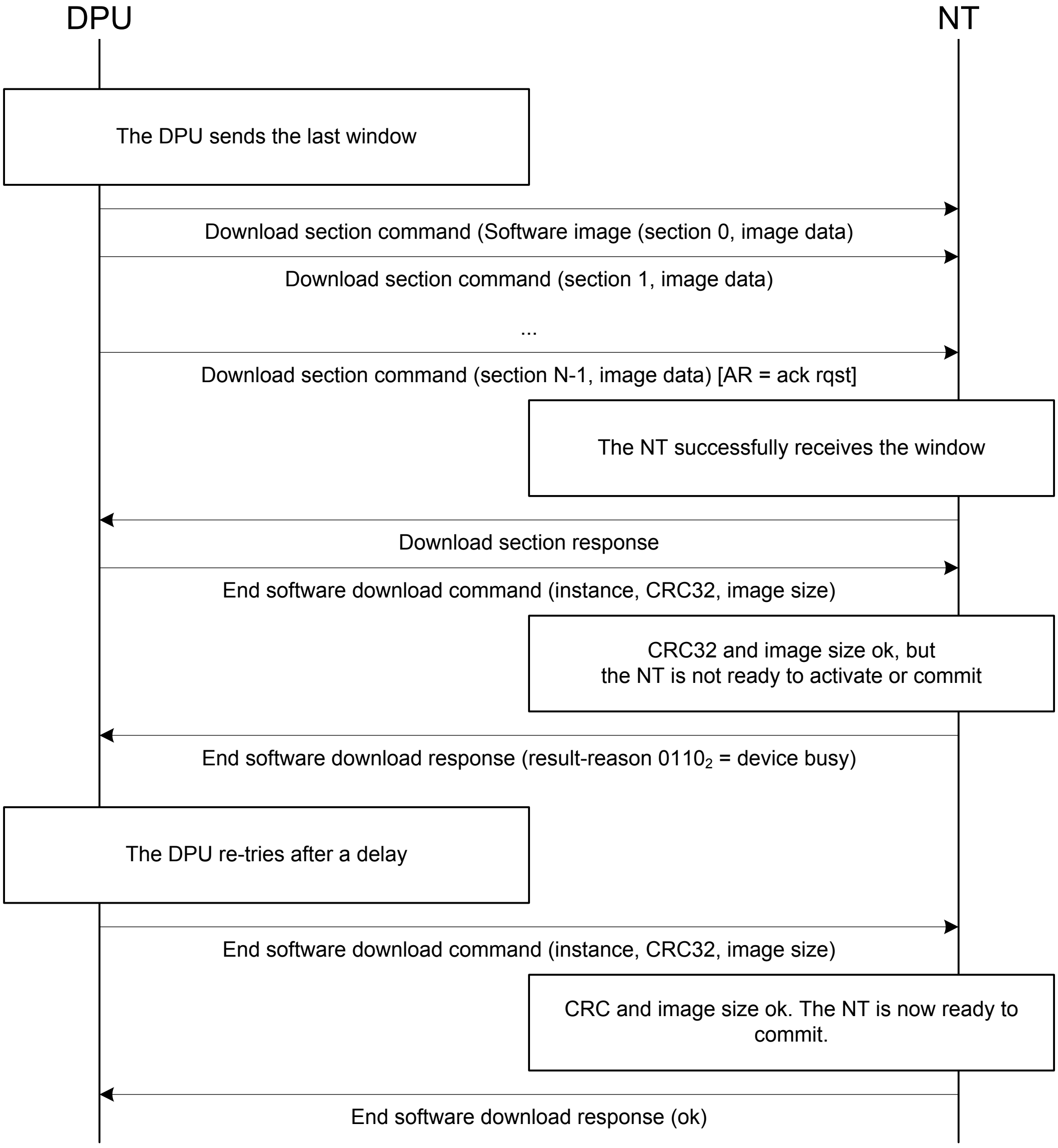

DPU
NT
Start software download (image 0, window size-1, image size)
The NT keeps software image 0 valid state.
The NT responds with same or smaller window size N.
Start software download response (image 0, success, window size-1)
The DPU accepts proposed window size N.
The DPU downloads first window: for illustration, this one contains N full sections.
Download section (image 0, section 0, section size of image data)
Download section (image 0, section 1, section size of image data)
...
Download section (image 0, section N-1, section size of image data) [AR = ack rqst]
Download section response (image 0, command processing error, section N-1)
NT signals failure to receive some or all of window.
The DPU re-transmits window. For illustration, DPU chooses to send only S sections this time, S ≤ N.
Download section (image 0, section 0, section size of image data)
...
Download section (image 0, section S-1, section size of image data) [AR]
Download section response (image 0, success, section S-1)
The DPU transmits final window. For illustration, F sections (F ≤ N) are assumed with less than section size bytes in section F – 1. DPU sends only F sections.
Download section (image 0, section 0, section size bytes of image data)
...
Download section (image 0, section F-1, remaining bytes of image data) [AR]
Download section response (image 0, success, section F-1)
The DPU terminates the software download
End software download (image 0, CRC-32, image size)
End software download response (image 0, success)
The NT terminates software download. It the NT was able to write the image to non-volatile memory, it sets new version, marks image 0 as valid, committed but inactivated

Figure 4 – Software download

The busy response handling for one image is illustrated in Figure 5, the NT responds with a device busy result code until these operations are complete, and the DPU periodically retries the end software download command. The DPU includes a timeout to detect an NT that never completes the download operation.



DPU
NT
The DPU sends the last window
Download section command (Software image (section 0, image data)
Download section command (section 1, image data)
...
Download section command (section N-1, image data) [AR = ack rqst]
The NT successfully receives the window
Download section response
End software download command (instance, CRC32, image size)
CRC32 and image size ok, but the NT is not ready to activate or commit
End software download response (result-reason 0110_2 = device busy)
The DPU re-tries after a delay
End software download command (instance, CRC32, image size)
CRC and image size ok. The NT is now ready to commit.
End software download response (ok)

Figure 5 – Busy response handling

The nested state machines in the DPU and NT can conceivably get out of step in a number of unspecified ways, and it isn't specified how to escape from a loop of transmission failure and retry. As a recovery mechanism from detectable state errors, it is recommended that the NT respond with command processing error result codes to both the acknowledged download section and end software download commands, and that the DPU send a final end software download command with a known bad CRC and image size (e.g., all 0s), whereupon both the DPU and NT reset to the state in which no download is in progress, that is, state S1 of Figure 2. Likewise, the DPU can abort the download operation at any time by sending an end software download command with invalid CRC and image size.

9.3 Software image activate and commit

Figure 6 shows the details of the software image activation process. When the NT has downloaded and validated a new software image and stored it in the non-volatile memory, it is automatically committed but not activated. The DPU may then send the activate image command to the NT. After the NT sends a positive activate image response, the NT then loads and executes the new software image. The time between the download and activate phases is not specified. This behavior is shown in Figure 6.

If the new image is stored in volatile memory at the end of the download process, the old image is still committed and activated. On reception of the activate image command, the NT sends a positive activate image response. Then the image is written to the non-volatile memory. If the action was successful, the NT sets the new version, marks image 0 as valid and committed. The NT then loads and executes the new software image. This behavior is shown in Figure 7.

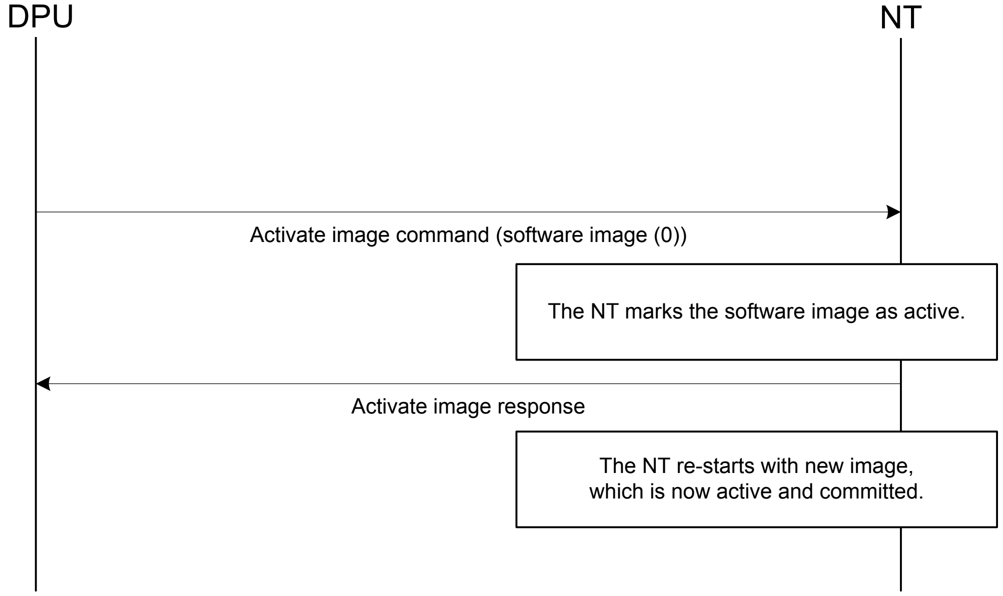

DPU
NT
Activate image command (software image (0))
The NT marks the software image as active.
Activate image response
The NT re-starts with new image,
which is now active and committed.

Figure 6 – Software activate and commit for the case when the image can be stored to the non-volatile memory at the end of the software download process

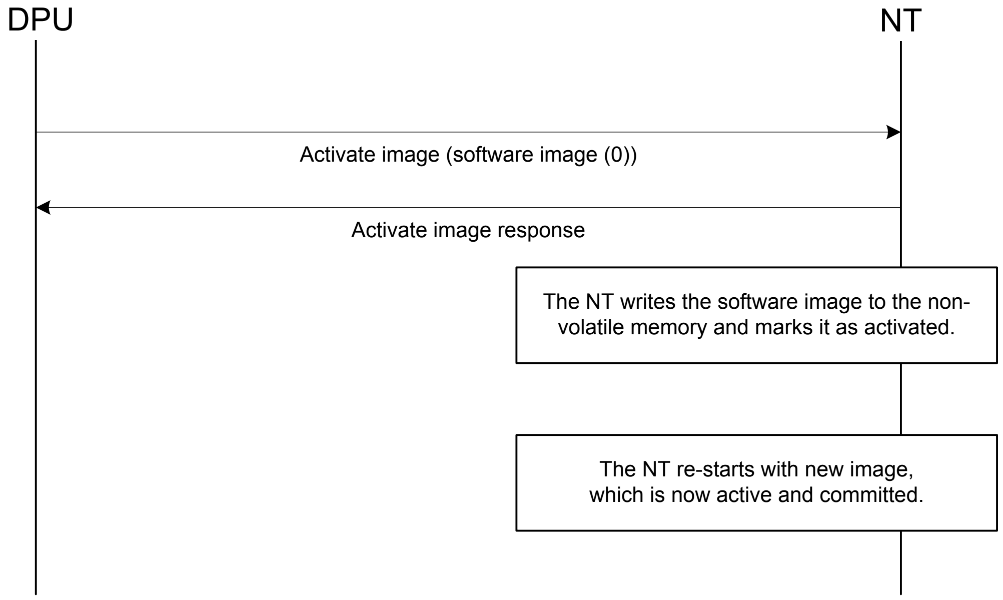

DPU
NT
Activate image (software image (0))
Activate image response
The NT writes the software image to the non-volatile memory and marks it as activated.
The NT re-starts with new image, which is now active and committed.

Figure 7 – Software activate and commit for the case when the image can't be stored to the non-volatile memory at the end of the software download process

____