Nomor Research GmbH

Integrated Conformance Software

Nomor Research GmbH
Munich, Germany
info@nomor.de

31 October 2018

Nomor Research GmbH
Brecherspitzstraße 8
81541 München

Table of Contents

Nomor Research GmbH
Brecherspitzstraße 8
81541 München

Table of Figures

Nomor Research GmbH
Brecherspitzstraße 8
81541 München

1 Introduction

The existing conformance software (https://github.com/Dash-Industry-Forum/Conformance-Software) has been extended according to new standards, such as CMAF, DVB-DASH, and HbbTV over time. However, these extensions have been spread over different parts of the entire software; and this makes testing, managing and further extending processes quite difficult. Therefore, a code refactoring which would clearly separate the different modules, functionalities and extensions from each other has been initiated.

This repository contains common modules (Utils) and submodules (Conformance-Frontend, DASH, CMAF, HbbTV, ISOSegmentValidator, HLS, Conformance-Frontend-HLS). Each submodule is a repository on its own; and all the submodules need the common modules.

The rest of the document describes the new conformance software tool structure, specifically, the overall functional overview, new module structure, and the detailed description of what each module is responsible for.

2 Installation

To clone the IntegratedConformance with all the submodules:

git clone --recurse-submodules https://github.com/Dash-Industry-Forum/IntegratedConformance

For the complete installation including dependencies etc, please refer [Installation guide]

3 Architecture

The functional diagram is drawn for the refactoring design and it is as the provided in Figure 1. This diagram contains individual modules, the scripts each module contains, and their connection to each other.

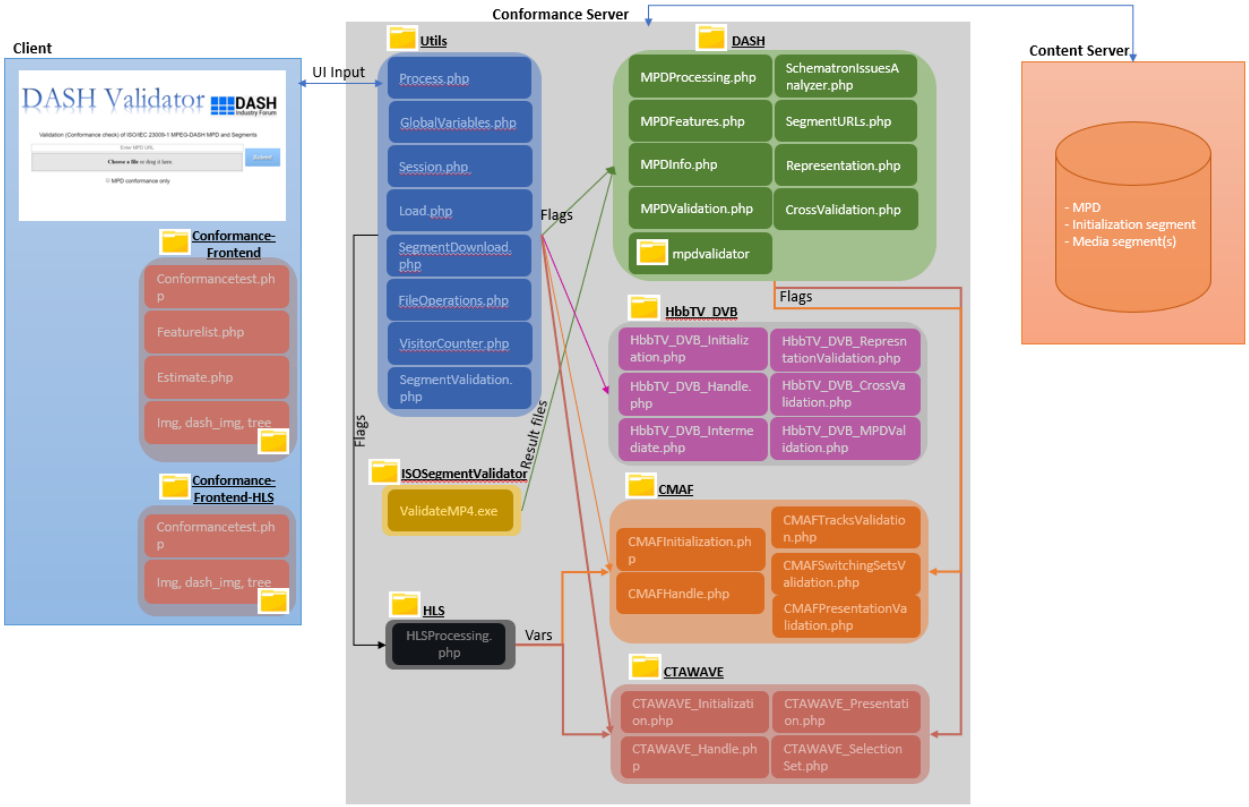


Figure 1: Functional diagram of Integrated Conformance Software.

4 Modules

The envisioned module architecture is as shown in Figure 2 below with red-dashed rectangles. As can be seen, there are eight separate modules:

1. Integrated Conformance, the main module, containing the all source code of every other module as submodule
2. DASH, containing DASH related functions and DASH MPD conformance checks
3. ISOSegmentValidator, that contains the segment validation software
4. CMAF, containing CMAF related functions and CMAF conformance check
5. HbbTV_DVB, containing HbbTV-DVB related functions and the conformance check
6. HLS, containing HLS manifest conformance
7. Conformance-Frontend, containing the web browser UI for DASH manifests
8. Conformance-Software-HLS, containing the web browser UI for HLS manifests

Nomor Research GmbH
Brecherspitzstraße 8
81541 München

These modules have separate Github repositories; however, the first module contain the other modules as submodules.

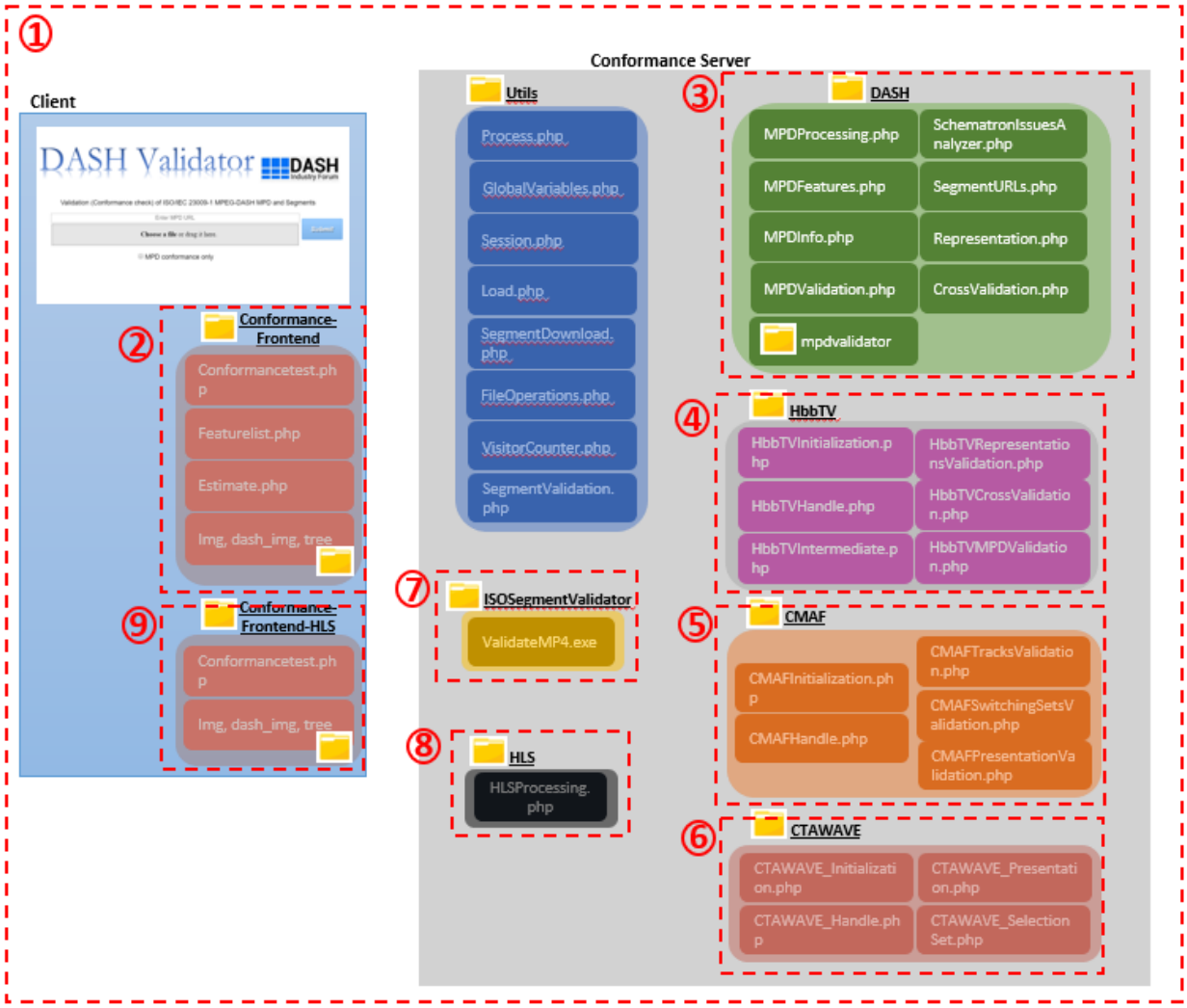


Figure 2: Modules in Integrated Conformance Software.

4.1 Conformance-Frontend

This module is the client UI for DASH manifests, containing all the UI-related functionalities to be provided to the user.

During conformance testing, the users can interact with the UI to obtain information on the progress, and intermediate results.

For each conformance testing, a different session is created, i.e. separated by session folders. Each session folder contains the content and the reports generated for the corresponding test. The storage of session folders resides here as well.

4.1.1 Conformancetest.php & Conformancetest.html

This is the main script in this module that provides the client UI and responsible for initiating the conformance server processes and updating the client about the overall progress as well as the conformance results.

It initially interacts with “Process.php” in “Utils” block as soon as Submit button is clicked on or local MPD file is provided. After the conformance check process starts, the intermediate and final results are sent to the frontend module for display. These results contain:

- MPD validation results
- MPD feature list
- Representation checks
- Cross-representation checks

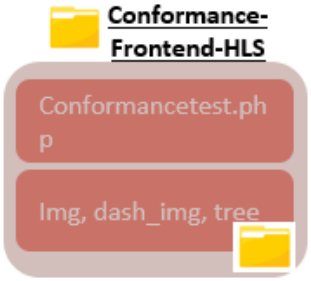
4.1.2 Featurelist.php

This script is responsible for generating the MPD features to be sent to the frontend module for display, just as in the original conformance software.

4.1.3 Estimate.php

This script provides information on minimum buffer time and bandwidth based on the information provided in each representation.

4.2 Conformance-Frontend-HLS



This module is the client UI for HLS manifests, containing all the UI-related functionalities to be provided to the user.

After conformance testing, the users can interact with the UI to obtain information on the final results.

The UI is quite similar to the UI used for DASH manifests (Conformance-Frontend) except the following:

- ***Profile enforcement:*** Only CMAF and CTA WAVE profiles enforcement is allowed.
- ***UI tree update:*** UI tree providing the validation reports is visible only after all the tracks are downloaded. This is because of the design choice selected for HLS support, which is explained in Section **Error! Reference source not found.**.
- ***Validation scope:*** Only HLS content pointed to by HLS manifest is validated. HLS manifest conformance is out of scope of this requirement.

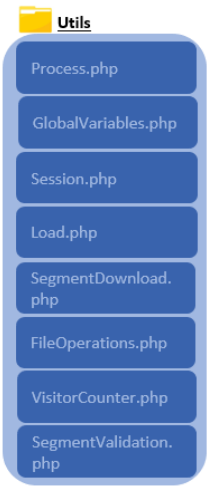
4.2.1 Conformancetest.php & Conformancetest.html

This is the main script in this module that provides the client UI and responsible for initiating the conformance server processes and updating the client about the overall progress as well as the conformance results.

As in Conformance-Frontend, it initially interacts with “Process.php” in “Utils” block as soon as Submit button is clicked on or local HLS file is provided. After the conformance check process starts, the final results are sent to the frontend module for display. These results contain:

- Representation checks
- Cross-representation checks

4.3 Utils



This block is the closest point to the frontend submodules. It contains useful functionalities that will be needed for DASH, HLS and extensions (CMAF, HbbTV, DVB, and CTA WAVE).

The scripts in this block are as seen in the Figure nearby. The detailed explanation of each script is as the following:

4.3.1 Process.php

This script is the initial script to be called from the frontend submodule when the conformance testing is requested to be performed.

It initiates the session creation (temporary storage folder) in the corresponding frontend module for the individual request by calling Session.php in this block. After the session creation, it initiates the conformance testing by calling MPDProcessing.php in DASH module or HLSProcessing in HLS module.

4.3.2 GlobalVariables.php

It stores generic global variables and filename templates.

4.3.3 SegmentDownload.php

It is responsible for fetching segments and storing them.

4.3.4 Session.php

It creates and deletes temporary session folders.

4.3.5 FileOperations.php

This script provides some useful helper functions regarding file operations, such as opening, closing and renaming a file, directory creation and deletion, computing the relative path of a file, and a modified version of command line execution process which enables both STDERR and STDOUT output to be visible to PHP. As a side note, the relative path computing is done for reporting the Client about the locations of generated HTML files.

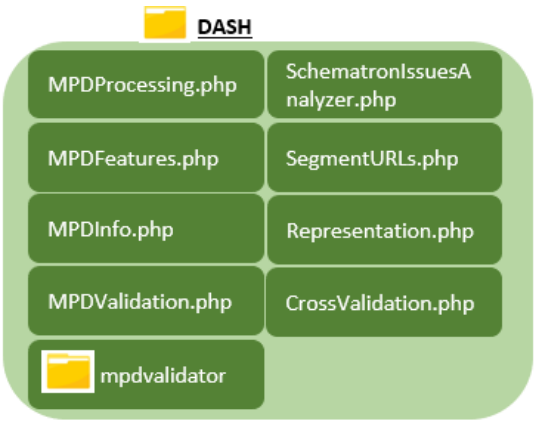
4.3.6 Load.php

This script is used to load provided MPD file into a DOM XML structure or any well-formed file into a DOM XML structure.

4.3.7 SegmentValidation.php

This script is responsible for the segment validation. It is called either from MPDProcessing.php for every representation or HLSProcessing for every media content. As input it requires segment(s) URL(s) corresponding to the specific representation/media content. After this, it downloads the segment(s) by calling download_data() in SegmentDownload.php in this block. If successfully downloaded, it adds the flags that will be passed to the ISOSegmentValidator module. It calls the executable binary in ISOSegmentValidator module for segment validation; the resulting reports (whose names are saved in GlobalVariables.php in Utils block already) from the ISOSegmentValidator submodule are renamed and reported to the frontend module for display.

4.4 DASH



This module contains all the related DASH conformance functionalities, such as parsing MPD, MPD validation, extracting segment(s) URL(s), segment validation, etc. Whenever a DASH manifest is provided from Conformance-Frontend module, this module is used for conformance testing. The scripts in this submodule can be seen in the nearby figure. Detailed explanation about each script is given below.

4.4.1 MPDProcessing.php

This script contains the main functionalities needed for the DASH conformance and the other extensions' conformances. The main function in this script is process_MPD() which is called from Process.php in Utils block right after a session is created. This function in itself calls the functionalities that load the provided MPD (Load.php in Utils block), parse it (MPDFeatures.php in this submodule), perform the MPD validation (MPDValidation.php in this submodule), compute the segmentURLs (SegmentURLs.php in this submodule), perform segment validation (SegmentValidation.php in Utils block) and cross checks (CrossValidation.php in this submodule).

4.4.2 MPDFeatures.php

This script is responsible for loading the MPD XML DOM structure to an array. Currently, this array is the exact representation of the MPD XML DOM. But it is designed for making the actual presentation compact (meaning that the common elements and attributes on different hierarchical levels can be merged into one for each representation). This script is called from MPDProcessing.php as mentioned above.

4.4.3 MPDInfo.php

This script extracts MPD information such as the determining the current period, determining the start times and durations of each period in the MPD, time parsing of the "xs:duration" attributes, determining the available segments for dynamic MPDs, and determining the profiles for each representation in each adaptation set in the current period. The last one for example can be moved MPDFeatures.php for obtaining a compact presentation for the MPD.

4.4.4 MPDValidation.php

This script is called for MPD validation including DASH MPD conformance and schema validation (the called java executable is located in mpdvalidator folder in this submodule). It generates MPD validation report that is reported to frontend submodule for display.

4.4.5 SchematronIssuesAnalyzer.php

This script is used to determine the schematron issues and report them to the frontend submodule for display, just as in the original conformance software

4.4.6 SegmentURLs.php

This script computes the segment(s) URL(s) for each representation for each adaptation set in the current period from MPD URL, BaseURL and segment access information. Segment access information corresponds to parsing the information provided in the MPD at any level through either SegmentBase or SegmentTemplate with or without SegmentTimeline element. SegmentList is not considered as it was the case in the original conformance software tool. The information extracted from here can actually be moved to MPDFeatures.php to obtain a compact presentation for the MPD.

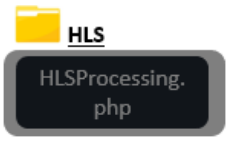
4.4.7 Representation.php

It generates the DASH flags to be passed to the segment validation executable for each representation.

4.4.8 CrossValidation.php

This script contains the DASH cross-representation checks. It is called from MPDProcessing.php after an adaptation set processing finishes and the result reports are then reported to frontend submodule for display.

4.5 HLS

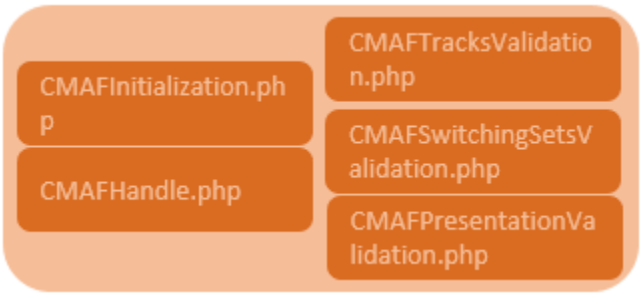


This module contains all the related HLS content conformance functionalities, such as extracting segment(s) URL(s) and segment validation. Whenever an HLS manifest is provided from Conformance-Frontend-HLS module, this module is used for conformance testing.

4.5.1 HLSProcessing.php

This script contains the main functionalities needed for the HLS content conformance and the other extensions' conformances. The main function in this script is process_HLS() which is called from Process.php in Utils block right after a session is created. This script itself loads the m3u8 manifest, extracts the segment(s) URL(s). After this, it performs segment validation for each media content (SegmentValidation.php in Utils block) and cross checks if any profile enforcement is done from the frontend module.

4.6 CMAF



This submodule is one of the extension conformances and contains the related functionalities for CMAF conformance. It makes use of Utils and DASH submodule for MPD processing as it is the same procedure. Different from these submodules, CMAF submodule introduces additional global variables and conformance checks specific to CMAF.

4.6.1 CMAFInitialization.php

The start of the CMAF process first starts when the profile enforcement box for CMAF is checked. When this is provided on the frontend module and sent to Process.php in Utils block with other input parameters and when Process.php determines this, Process.php includes CMAFInitialization.php script in this submodule. This script contains the include statements of the remaining scripts in this submodule and the global variables $function_name and $when_to_call and initialization of the other CMAF-related global variables that are defined as null in Utils block. $function_name is a string containing ‘CMAFHandle’ and $when_to_call is an array containing strings ‘Tracks’, ‘SwitchingSets’ and ‘Presentation’. The idea behind these global variables is to provide the CMAFHandle() function in CMAFHandle.php to be called at specific times determined by the $when_to_call array. This way, CMAF is made a black box with an interface.

4.6.2 CMAFHandle.php

When CMAFHandle() is called with the a string from the $when_to_call array, it calls the corresponding script in this submodule and returns the result that is reported from the corresponding script. This script is called either from MPDProcessing.php in DASH module or HLSProcessing.php in HLS module.

4.6.3 CMAFTracksValidation.php

This script contains the representation checks and called from MPDProcessing.php in DASH submodule or HLSProcessing.php in HLS submodule via CMAFHandle with the string ‘Tracks’. This procedure is called for each representation/media content.

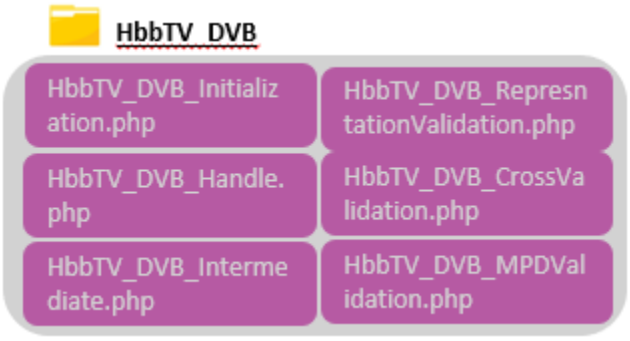
4.6.4 CMAFSwitchingSetsValidation.php

This script contains the adaptation set checks and called from MPDProcessing.php in DASH submodule or HLSProcessing.php in HLS submodule via CMAFHandle with the string 'SwitchingSet'. This procedure is called for each adaptation.

4.6.5 CMAFPresentationValidation.php

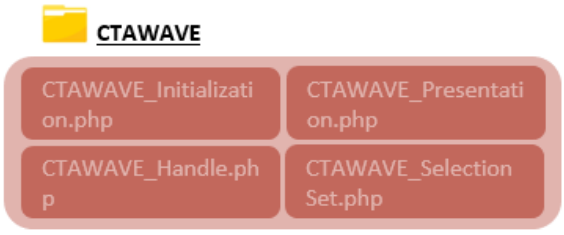
This script contains the cross adaptation set checks and the overall presentation checks. It is called from MPDProcessing.php in DASH submodule or HLSProcessing.php in HLS submodule via CMAFHandle with the string 'Presentation'. This procedure is called after the processing of the current period finishes.

4.7 HbbTV_DVB



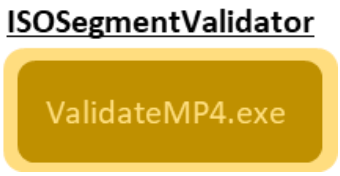
This submodule is another extension conformance and contains the related functionalities for HbbTV and DVB-DASH conformance. The start of this validation first starts when the profile enforcement box for HbbTV and/or DVB is checked. The submodule structure and the flow are same as the one described in CMAF module.

4.8 CTA WAVE



This submodule is another extension conformance and contains the related functionalities for CTA WAVE conformance. The start of this validation first starts when the profile enforcement box for CTA WAVE is checked. The submodule structure and the flow are same as the one described in CMAF module.

4.9 ISOSegmentValidator



This submodule contains the segment validation software known as the Backend. This module is responsible for the atom/box validation of the ISO Base Media File Format packaged media content. This module is called by the main module to validate the contents after downloading the content pointed to by URLs. After the execution, it provides information and error log files to be interpreted and reported by the DASH/HLS.

Note: Only the Linux part of ISOSegmentValidator, as we do not support Windows anymore.