



Refactoring the DASH-IF **Conformance** Tool : the rise of conformance for the masses



Presenter:
romain.bouqueau@motionspell.com

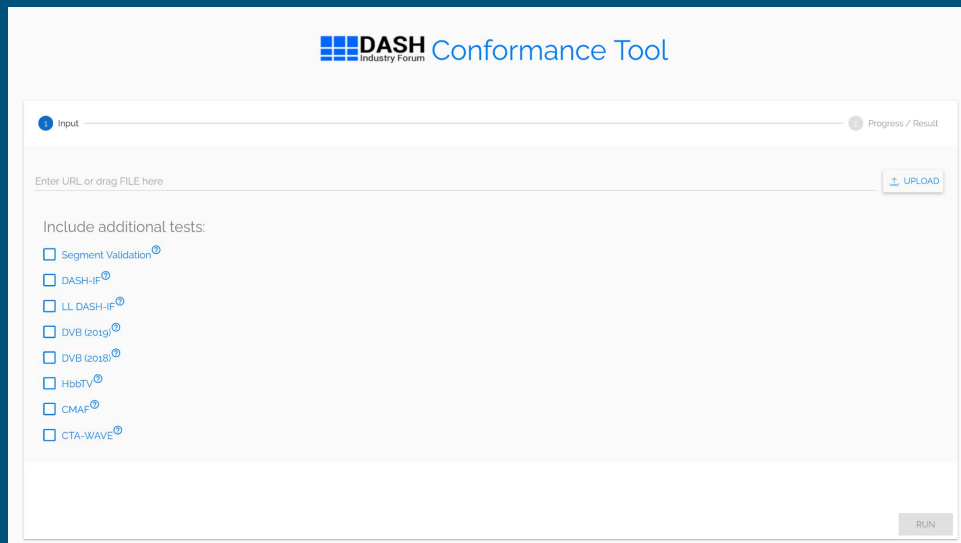


What this presentation is about:

- The importance of **conformance**
- How the **standardization process** interacts with conformance
- Leveraging **open-source** to support new technology
- A **refactoring** to on an existing project to make it more attractive and sustainable

Overview of the DASH-IF Conformance Tool

- Well known **tool**
- Mostly used **online**
- **Popular**: ~180 input per day
- Mostly **maintained by DASH-IF**
 - Industry contributions through consortia: **DVB**, **HbbTV**, CMAF (by the **CTA WAVE** initiative)
- Licensed as **GPLv3**:
 - + other dependencies
 - In practice not an issue as it is deployed on servers



The screenshot shows the web interface of the DASH-IF Conformance Tool. At the top, the logo for the DASH Industry Forum and the text 'DASH Conformance Tool' are visible. Below the header, there is a progress bar with two steps: 'Input' (active) and 'Progress / Result'. The main area contains a text input field labeled 'Enter URL, or drag FILE here' with an 'UPLOAD' button to its right. Below this, a section titled 'Include additional tests:' lists several checkboxes: 'Segment Validation', 'DASH-IF', 'LL DASH-IF', 'DVB (2019)', 'DVB (2018)', 'HbbTV', 'CMAF', and 'CTA-WAVE'. At the bottom right of the interface is a 'RUN' button.

Conformance: why?

- MPEG specifications describe the **client behaviour** [1]
 - Typically a decoder
 - How to you know that the content you create is conform?
- Conformance is key for **interoperability**
 - Enforce a **common understanding**
 - Diverging can be very **costly**
 - Handle more **complex technology**
 - **Faster time-to-market**

[1] <https://www.motionspell.com/2013/10/14/standard-cover-mpeg-dash-case-study/>

Conformance: who?

Spec editor

- Writes normative statements.
- Wants to illustrate the specification with test vectors.
- Wants to enforce the specification with a conformance checker.
- **Solution: may want to annotate the document with a link to the code of the check + associated test vectors.**

Conformance checker

- Coverage of spec normative statements.
- Robust. **Unit tested.**
- May aggregate reports from several specs (codecs, container, manifests, etc.)
- **Solution: each check associated with metadata**

Unit Test

Industry
content

Reference implementation
(e.g. dash.js for DASH)

Test material

- Which normative statement does it test?
 - Positive or negative
 - Which spec?
 - Complies with which rule or spec or profile? (result from the conformance checker)
- **Solution: can be either manually generated, industry provided**
- **Solution: the person/entity pushing a feature in the spec is responsible for providing the test vectors**

Conformance: how?

- Conformance can take several forms:
 - Specification: **normative statements** (shall/should/may keywords)
 - **Test vectors**
 - Reference **software** implementations e.g. a client (dash.js)
 - `/!\` Robustness principle: be liberal in what you accept `/!\`
 - **Conformance** tool:
 - Code coverage of the tool $\Leftrightarrow$ coverage of the specification
 - `/!\` Doesn't test about the presence of features, just consistency `/!\`
 - Test vectors are correct and don't provide good code coverage

Back to the DASH-IF conformance tool: Challenges

- General **maintenance**
 - Aggregate of efforts over 10 years
 - Updates needed
 - Not enough tests despite attempts
- Difficult to **contribute**
 - Only 50 github stars!
 - Documentation scattered and specific
 - Mixing many technologies (PHP, JS, Python, C++...)
- Role w.r.t. the **standardization** processes?
 - Spec editors = sponsors
 - Enforcing restrictions
- Now necessary **beyond DASH-IF**
 - Joint effort from DVB, DASH-IF, ATSC, CTA, HbbTV
 - Executed by Motion Spell and Fraunhofer

What is **refactoring**?

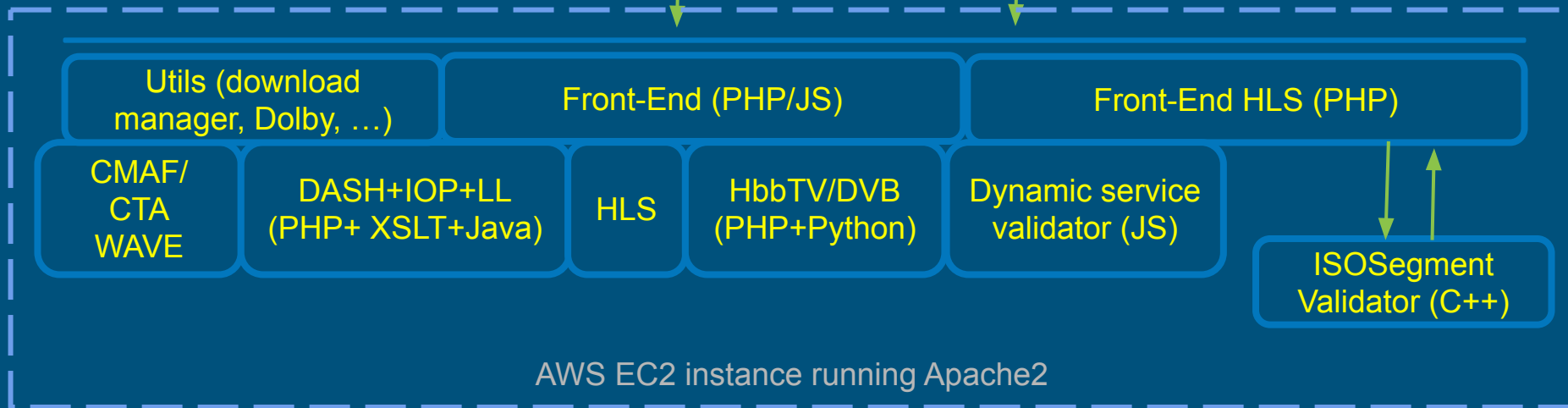
1. Moving code around without changing the functional scope.
2. Improving the tools by tracking **metrics**.
3. Confidence is key. **Tests are confidence**.
4. How do we build tests?
 - a. Clear **APIs**. Re-architecture.
 - b. Accepting that **a unit test is not a test vector**.
 - c. Leveraging the power of **open-source and communities**.
5. General goal to **improve the tech innovation process**:
 - a. Writing specs.
 - b. Writing implementations.

Refactoring:

Web front-end

Command-line
(e.g. curl) -> broken

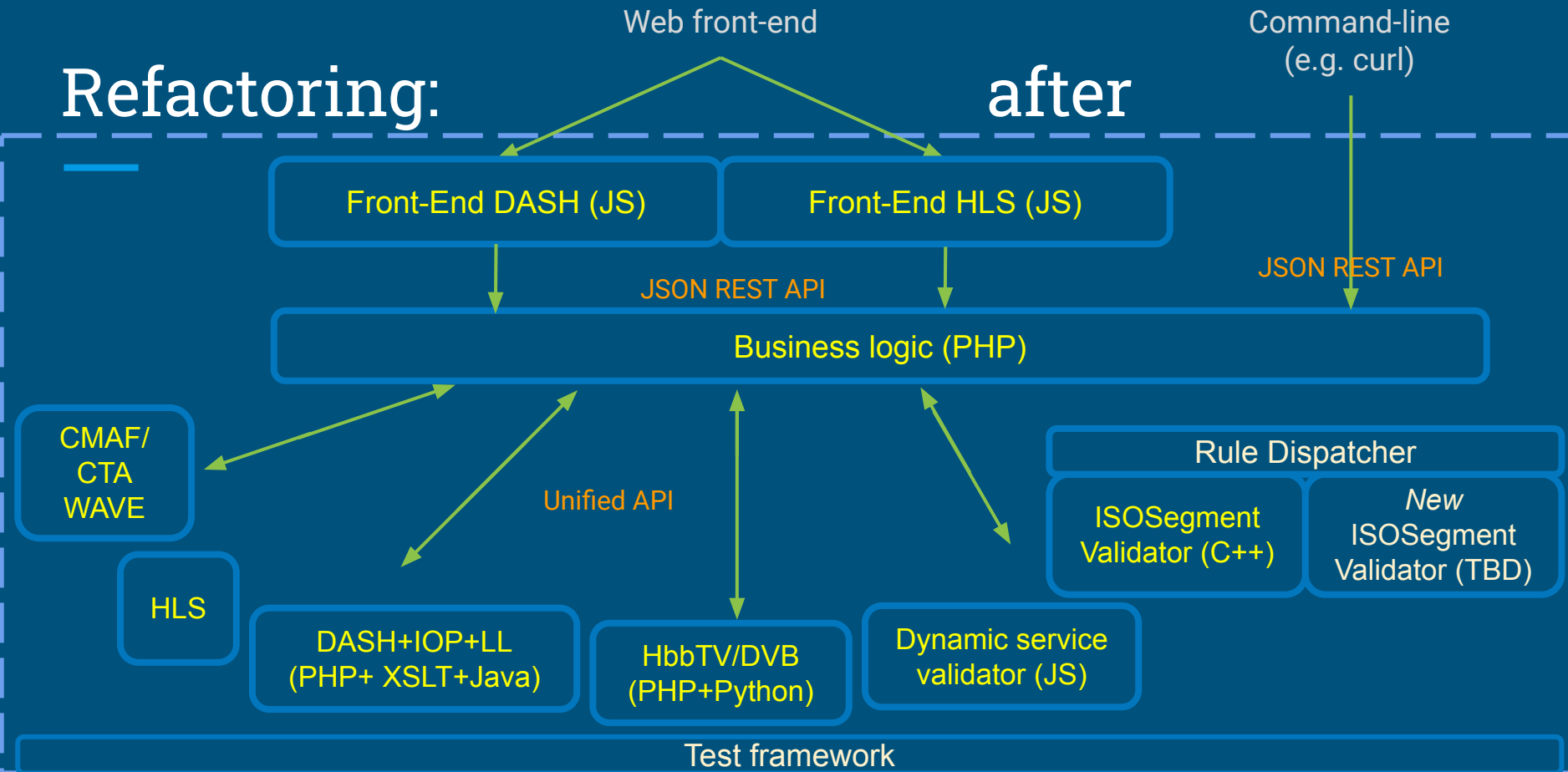
before



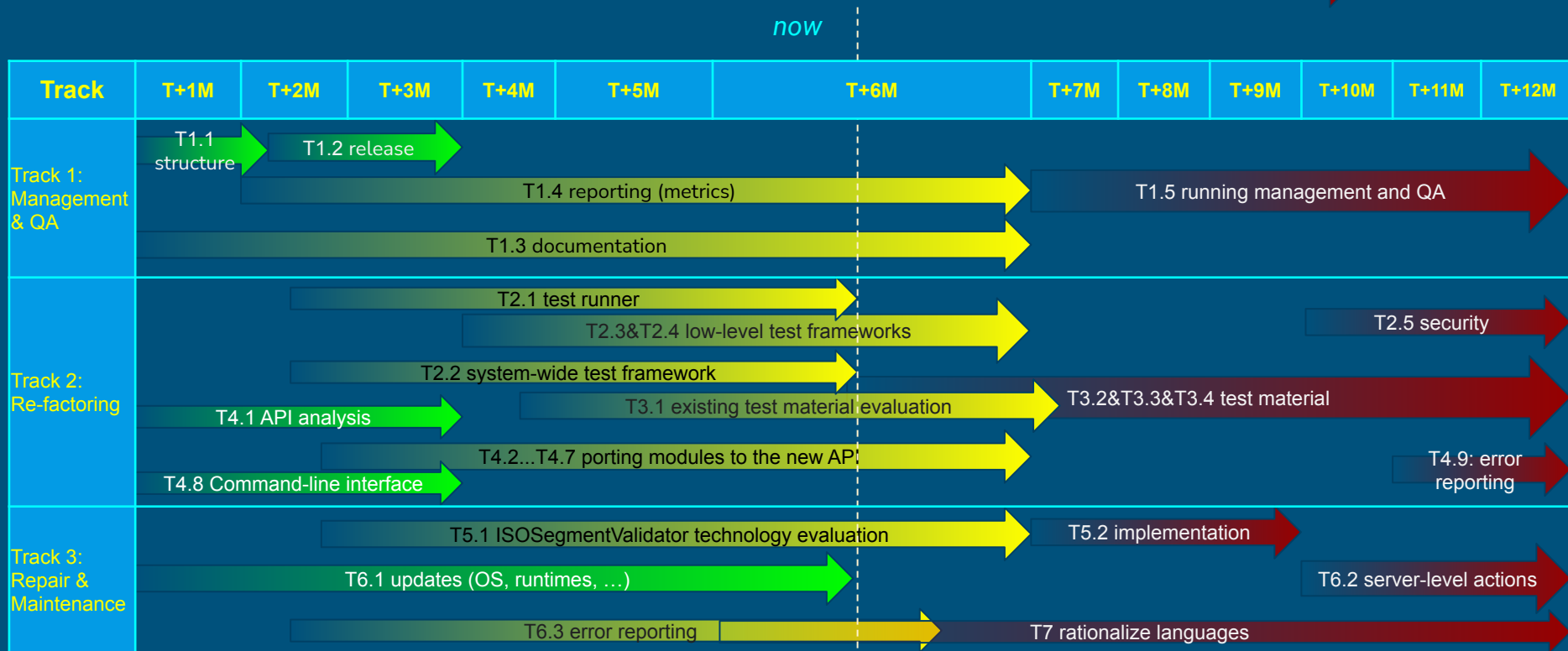
The front-end contains:
1) the UI
2) the business logic
3) is intricated with submodules

Refactoring:

after



Execution timeline



Open project

- Public visibility
- Open-source
- Documented, automatically, accessible
- Open groups: public contributions, mailing-lists, announcements
 - <https://groups.google.com/g/joint-conformance-software-project-jccp>
- Issue that **most standards are not open**

Thank you !

romain.bouqueau@motionspell.com