Portable Web Publications: Technology Challenges

Ivan Herman, W3C

W3C Track @ WWW2016

2016-04-13

These Slides are Available on the Web

See:

`http://www.w3.org/2016/Talks/W3CTrack-IH/`

(Slides are in HTML)

Is it a book? Is it a Web site?

3.3 Multiple Alternatives

Multiple if statements can be combined to evaluate complex decisions.

In Section 3.1, you saw how to program a two-way branch with an `if` statement. In many situations, there are more than two cases. In this section, you will see how to implement a decision with multiple alternatives.

For example, consider a program that displays the effect of an earthquake, as measured by the Richter scale (see Table 3).

Table 3 Richter Scale

Value	Effect
8	Most structures fall
7	Many buildings destroyed
6	Many buildings considerably damaged, some collapse
4.5	Damage to poorly constructed buildings

The Richter scale is a measurement of the strength of an earthquake. Every step in the scale, for example from 6.0 to 7.0, signifies a tenfold increase in the strength of the quake.

Credit: Extract from "Big Java", by Cay Horstmann, John Wiley & Sons, 2013

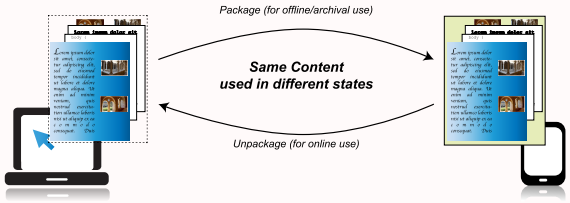
The main message:


Digital Publishing
=
Web Publishing!

put it another way...


Web Publishing
=
Digital Publishing!

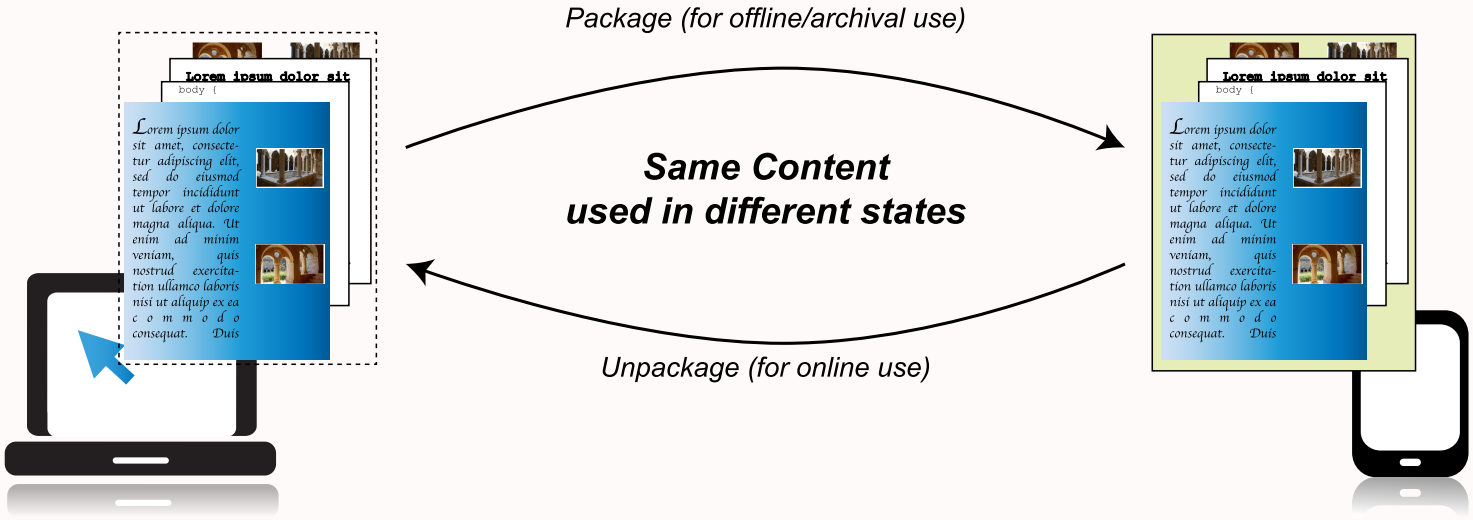
What does this mean?



Separation between publishing "online", as Web sites, and offline and/or packaged should be diminished to zero

- This means:
  - publication content on the Web can be loaded into a browser or a specialized reader, whatever the user prefers
  - a publication on a local disc can be pushed onto the Web and used without any change
  - content are authored regardless of where they are used
  - these are done without any user interaction, possibly automatically

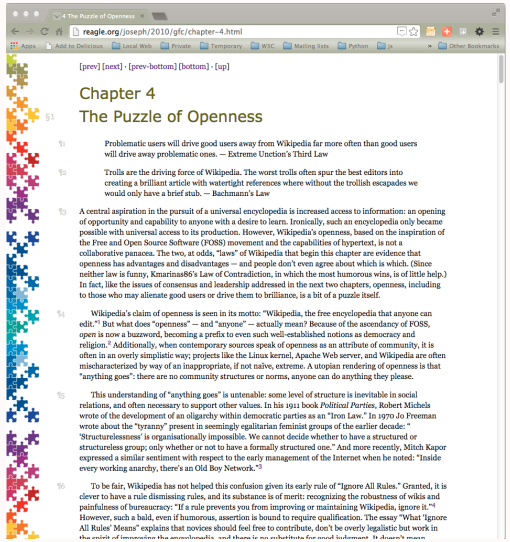
What does this mean?



Credit: ibta arabia

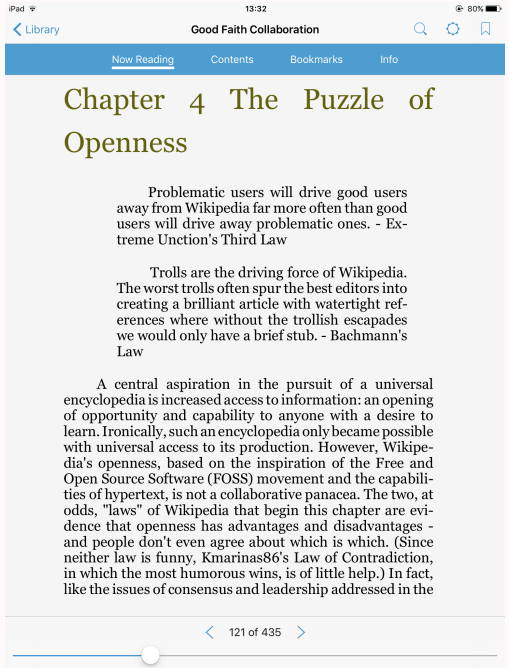
For example: book in a browser

- On a desktop I may want to read a book just like a Web page:
  - easily follow a link “out” of the book
  - create bookmarks to “within” a page in a book
  - use useful plugins and tools that my browser may have
  - create annotations
  - sometimes I may need the computing power of my desk-top for, e.g., interactive 3D content



Credit: Extract of Joseph Reagle's Book

For example: book in a browser



Credit: Extract of Joseph Reagle's Book as ePUB

- But, at other times, I may also want to use a small dedicated reader device to read the book on the beach…
- All these on *the same* book (not conversions from one format to the other)!

For example: I may not be online...

- I may find an article on the Web that I want to review, annotate, etc., while commuting home on a train
- I want the results of the annotations to be back online, when I am back on the Internet
  - note: some browsers have an "archiving" possibility, but they are not interoperable

Credit: Bryan Ong, Flickr

For example: educational publications

- What is an educational publication?
  - *a book* that requires offline access?
  - *a packaged application* with built-in interactive tests, animated examples?
  - *a Web client* reaching out to Web services for assessing test results, to encyclopedia, …?
  - *an interactive data container* storing various data for, e.g., demonstrations?

Credit: Merrill College of Journalism, Flickr

- The borderline between a “book” and a “(Web) Application” is becoming blurred…

Synergy effects of convergence

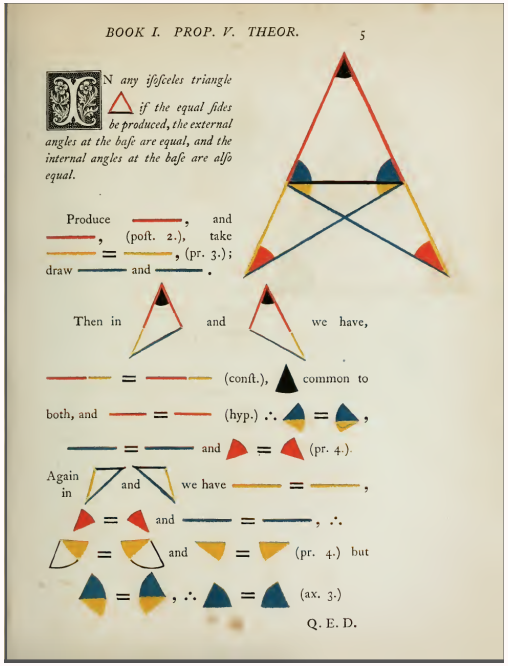
Advantage for the publishers' community

Credit: Jeffrey Zeldman, Flickr

- The main interest of publishers is to produce, edit, curate, etc, *content*
- Publishers have invested heavily into technology developments, but the Web developers' community can complement that with a wider reach and perspective
- Working closely with Web developers avoids re-inventing wheels

Advantage for the Web community

- Publishers have experience in:
  - ergonomics, typography, aesthetics…
  - publishing long texts, with the right readability and structure
- Workflow for producing complex content



Credit: Oliver Byrne's edition of Euclid, University of British Columbia

But... why not rely *only* on the Web?
(i.e., forget about downloaded content, it is outdated!)

Several reasons…

- The future may be that everyone is always connected… but the reality is different for many years to come
  - slow connections, e.g., or on a plane or bus or even in some areas
  - huge roaming prices among countries
- Current publishing business models rely on distributable entities
- Privacy or security issues may require off-line access
  - e.g., in a plane cockpit
- Archiving considerations

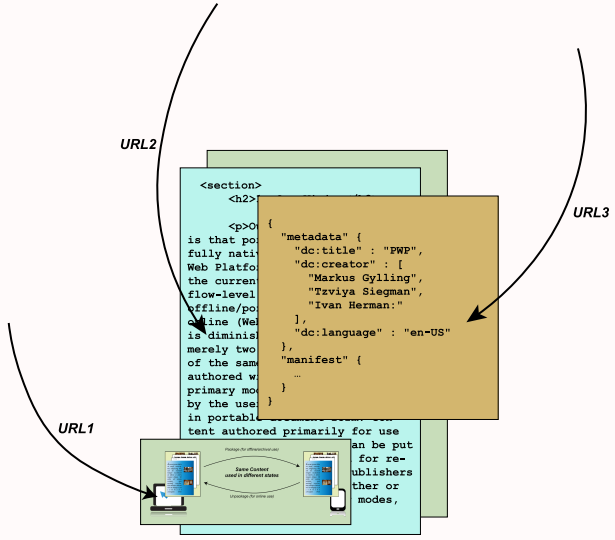

How do we get there?
(Technically)

Credit: Catherine Kolodziej, Flickr


Technical Challenge:
Fundamental Terminology

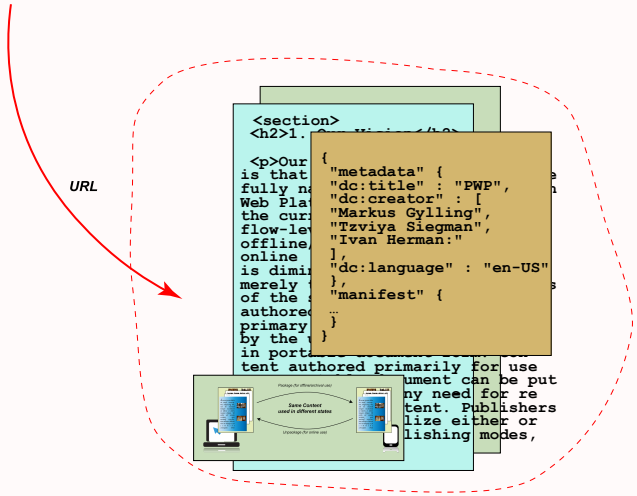
Web Publications

- The current Web has the notion of a single resource:
  - conceptually, a single piece of data
    - HTML source, metadata, CSS style sheet, etc.
  - each has its own URL
- Presentation is based on the interoperation of many such resources



Web Publications

- But publishers need the concept of a single *Publication*:
  - a *collection* of pages, together with the relevant CSS, images, video, etc., files
  - it is the *collection* that has a real distinct identity (URL), *not* its constituents



Formally

- A ***Web Publication***: an *aggregated set of interrelated Web Resources*, intended to be considered as a single entity, and which *can be addressed on the Web as a unit* (is itself a Web Resource)

Portable Web Publications

- A Web Publication may consist of resources spread all over the place (HTML on one site, CSS somewhere else)
  - the owner of the Web Publication is only a "user" and not necessarily the owner of some of those resources!
- But a publishers may want to, create, curate, move the *whole* publication, as a single unit
- The Web Publication should be, in some sense, "self consistent", not relying on external entities.
- A "self-consistent" Web Publication is therefore *Portable*

More Formally

- A ***Portable Web Publication*** is such that a *user agent can render its essential content by relying on the Web Resources within the same Web Publication*

What kinds of documents are we talking about?

- A journal or magazine article, including the relevant CSS files and images
- An educational article, including the JavaScript to do interactive exercises
- A novel or a poem on the Web, including the necessary fonts, CSS files, etc., to provide the required aesthetics

What kinds of documents are we *not* talking about?

- A Web mail application
- A social Web site like Facebook, VK, Renren, or Twitter
- A dynamic page that depends on, say, a Javascript library hosted somewhere on the cloud

But there are of course differences

- Although the same content of a PWP whether offline or on-line, but they are obviously not absolutely the same
- We refer to different *states* of the same PWP

Envisioned “states” of a Portable Web Publication

	Protocol Access	**File Access**
Packed	PWP as one archive on a server	PWP as one archive on a local disc
Unpacked	PWP spread over several files on a server	PWP spread over several files on a local disc

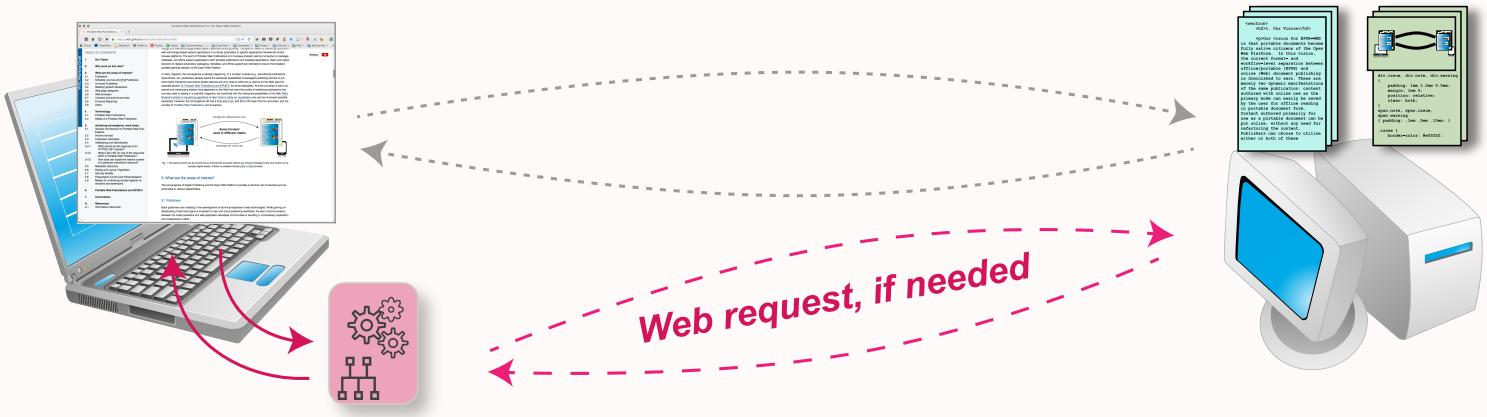

Technical challenge: an overall architecture to handle PWP-s

Envisioned architecture: a “PWP Processor”

- A conceptual, client-side processor that “hides” the PWP state differences from the rendering engine
- The “main” rendering engine operates *as if* it was connected to the Web:
  - accessing resources through HTTP(S)
  - all resources are “unpacked”
- The PWP Processor should hide the state differences, possibly cache resources, etc.

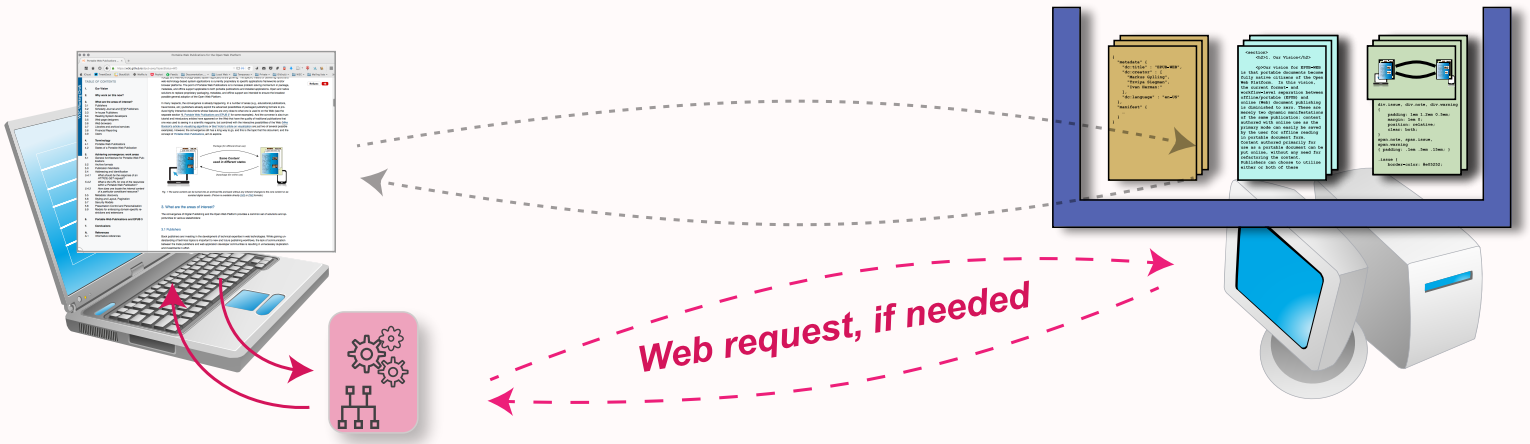
Envisioned architecture: unpacked state

Envisioned architecture: cached state



Requests handled by the PWP Processor
content possibly cached

Envisioned architecture: packed state



Requests handled by the PWP Processor content unpacked on-the-fly and possibly cached

Envisioned architecture: packed state

Draft...

Is this approach at all feasible?

Advances in modern browsers: Web and Service Workers

- Web Worker: a truly parallel thread within the browser
- A Service Worker is a special type of Web Worker, with additional features:
  - it is a *programmable network proxy*: the renderer's network calls are caught and the request/answer can be modified on-the-fly behind the scenes
  - it has an interface to handle a local cache for networked data
  - it will stay alive even if the user moves away from the main page, and can be accessed later if he/she returns to it

Advances in modern browsers: Web and Service Workers

- Web Worker: a truly parallel thread within the browser
- A Service Worker is a special type of Web Worker, with additional features:
  - it is a programmable network proxy: the renderer's network calls are caught and the requested data can be modified on-the-fly behind the scenes
  - it has the ability to handle a local cache for networked data
  - it will survive even if the user moves away from the main page, and can be accessed later if he/she returns to it

Work in progress

A PWP Processor could be implemented as a Service Worker

Not only a wild idea…

- Some prior art exists (e.g., experimentation by the Readium Consortium with Service Workers)
- An early mock-up of the current architecture has also been done
  - caveat for now: current Service Worker specification does not allow for direct, local file access
  - some extra tricks have to be found

Technical challenge: addressing, identification

Is it "addressing" or is it "identification"?

- *These two "roles" are different*
- The usual situation:
  - some form of a URI is used to *(uniquely) identify* a resource
  - an HTTP(S) URL is used to *address* (or "locate") a resource on the Web
- In many cases the two roles coincide, but not always
  - e.g., for a digital Book :
    - `URN:ISBN:1-56592-521-1` identifies the publication
    - `http://www.ex.org/ex.epub` addresses a particular copy

Is it "addressing" or is it "identification"?

- Identification issues are handled by a number of other organizations (DOI foundation, International ISBN Agency, etc.)
- The work on PWP has to concentrate on locators (i.e., addressing)

Three layers of addressing

1. Locator for the PWP itself:

   `http://www.ex.org/MyPWP/`

2. Locating a resource *within* a PWP:

   `http://www.ex.org/MyPWP/Chapter1.html`

3. Locating a target within a resource:

   `http://www.ex.org/MyPWP/Chapter1.html#section1`

- #3, i.e., “fragments" is defined for specific media types
- #2 *should* be just like any other resources on the Web, to allow for a smooth state transition

Locating the different PWP "states"

- There are, in practice, two different locators
  - to the unpacked version on the Web ($\mathbf{L_u}$):

    `http://www.ex.org/MyPWP/dir/`
  - to the package ($\mathbf{L_p}$):

    `http://www.ex.org/MyPWP.pwp`
- Which locators should one use? How would intra-resource addressing happen?
  - i.e., how should `chapter1` refer to `chapter2` ?

Canonical locators

- A PWP must have a *Canonical Locator* (**L**)
  - a state agnostic locator: `http://www.ex.org/MyPWP`
- A published PWP *must* provide metadata that includes **L**, $\mathbf{L_p}$, and $\mathbf{L_u}$
- A PWP Processor *must* have access to the full metadata
- A resource within a PWP (and, in general, resources in general) should use **L** only for internal cross-references (or use relative URL-s)

The PWP Processor can take care of the rest...

- The processor has an access to $\mathbf{L}$, $\mathbf{L_p}$, and $\mathbf{L_u}$
- It can, if needed, convert among URI requests coming from the renderer
- Remember:

What does an `HTTP GET` return for **L**?

- Possibilities are:
  - the full manifest
  - the package that *includes* the manifest at some predefined place
  - an HTML file with a link to a manifest (through a `<link>` element)
  - an HTML file with an *embedded* manifest (through a `<script>` element)
  - some Web Resource, with a link to a manifest in the `Link` header of the `HTTP` response
- A PWP Processor should consider all these possibilities and *combine* the various sources
- Different server setups are possible; a PWP specification should leave that open

Getting hold of all locators

Getting hold of all locators

Work in progress

Manifests

- Note the crucial importance of the metadata
- Some sort of a “manifest” format should be defined to hold (among others) this metadata
- The manifest will be used for other, more traditional reasons, too:
  - “traditional” metadata like author(s), right expressions, publication dates, ...
  - *identifiers*
  - correct reading order for the publication content
  - etc.

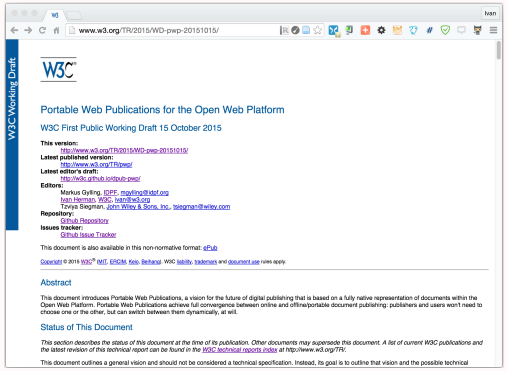
Technical challenge: presentation control (a.k.a. Personalization)

- What is the level of user control of the presentation?
- The Web and eBook traditions are vastly different:
  - in a browser, the Web designer is in full control
    - CSS alternate style sheets or user style sheets are hardly in use
    - some user interface aspects can be controlled but only for the browser as a whole
  - in an eBook reader, there is more user control
    - foreground/background color
    - choice of fonts
- There is a need to reconcile these traditions


How do we get there?
(Practically)

DPUB IG and Portable Web Publications

- "Portable Web Publications" was, originally, a separate "vision" document
- Was adopted, formally, as part of the group's work in September 2015, and is now published as an IG document



- The group will contribute to the formulation of the PWP technical challenges, to a better understanding of the requirements
- *PWP is the guiding principle for the group's further work*

IDPF, W3C, and others

- On long term, some PWP related standard-track specification work may have to be done
  - this requires a consensus and agreement of different communities
- IDPF and W3C (and maybe others?) may create the necessary groups, eventually

Some references

DPUB IG Wiki

https://www.w3.org/dpub/IG/wiki/Main_Page

Latest PWP Official Draft:

http://www.w3.org/TR/pwp/

PWP Editors' draft:

https://w3c.github.io/dpub-pwp/

PWP Issue list:

https://github.com/w3c/dpub-pwp/issues

Thank you for your attention!

This presentation:

`http://www.w3.org/2016/Talks/W3CTrack-IH/`

(PDF is also available for download)

My contact:

ivan@w3.org