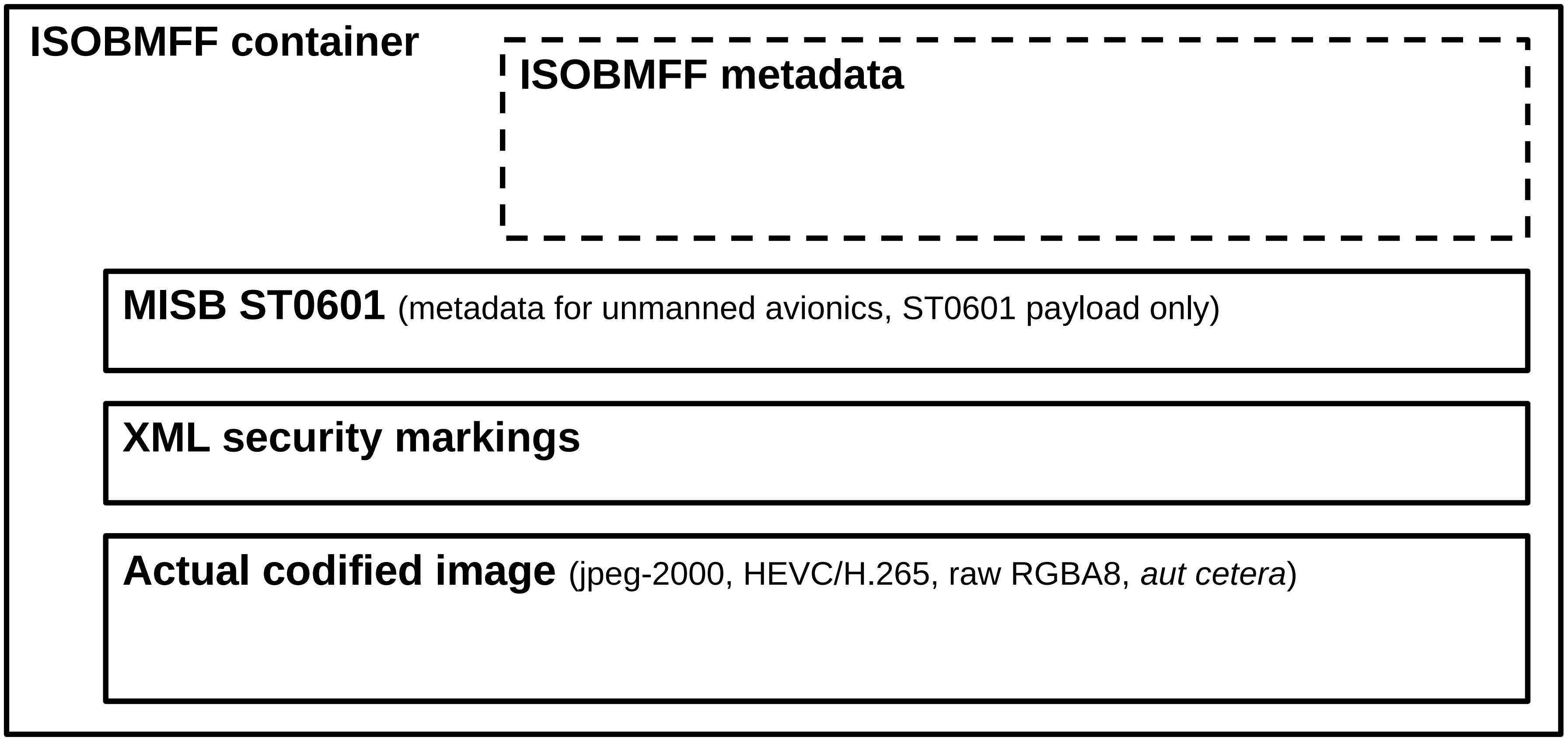
Informal report about GIMI profile

Iván Sánchez Ortega <ivan@sanchezortega.es>

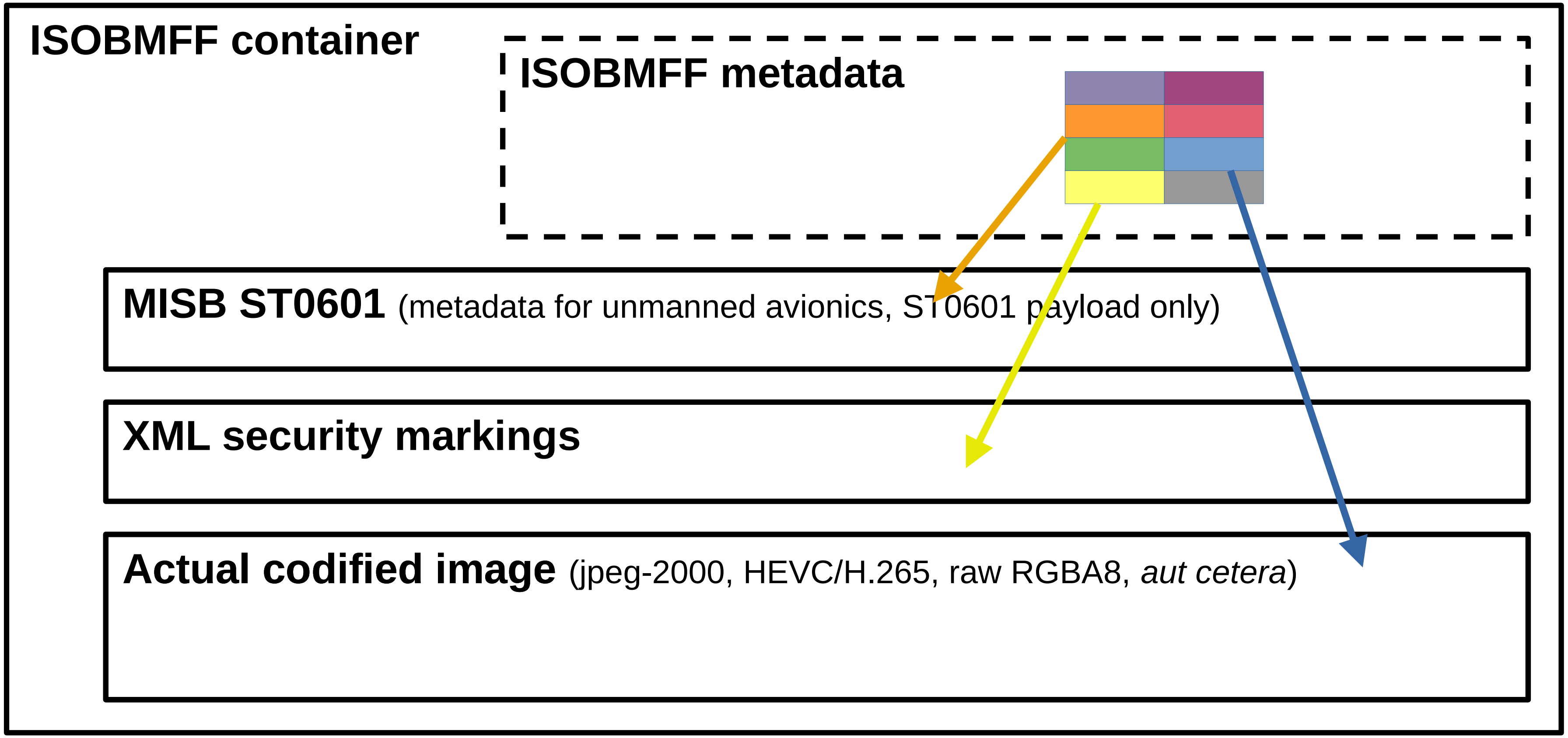
Made hastily during the October 2023 OGC open standards codesprint

Disclaimer: May contain personal biased opinions and incorrect assumptions

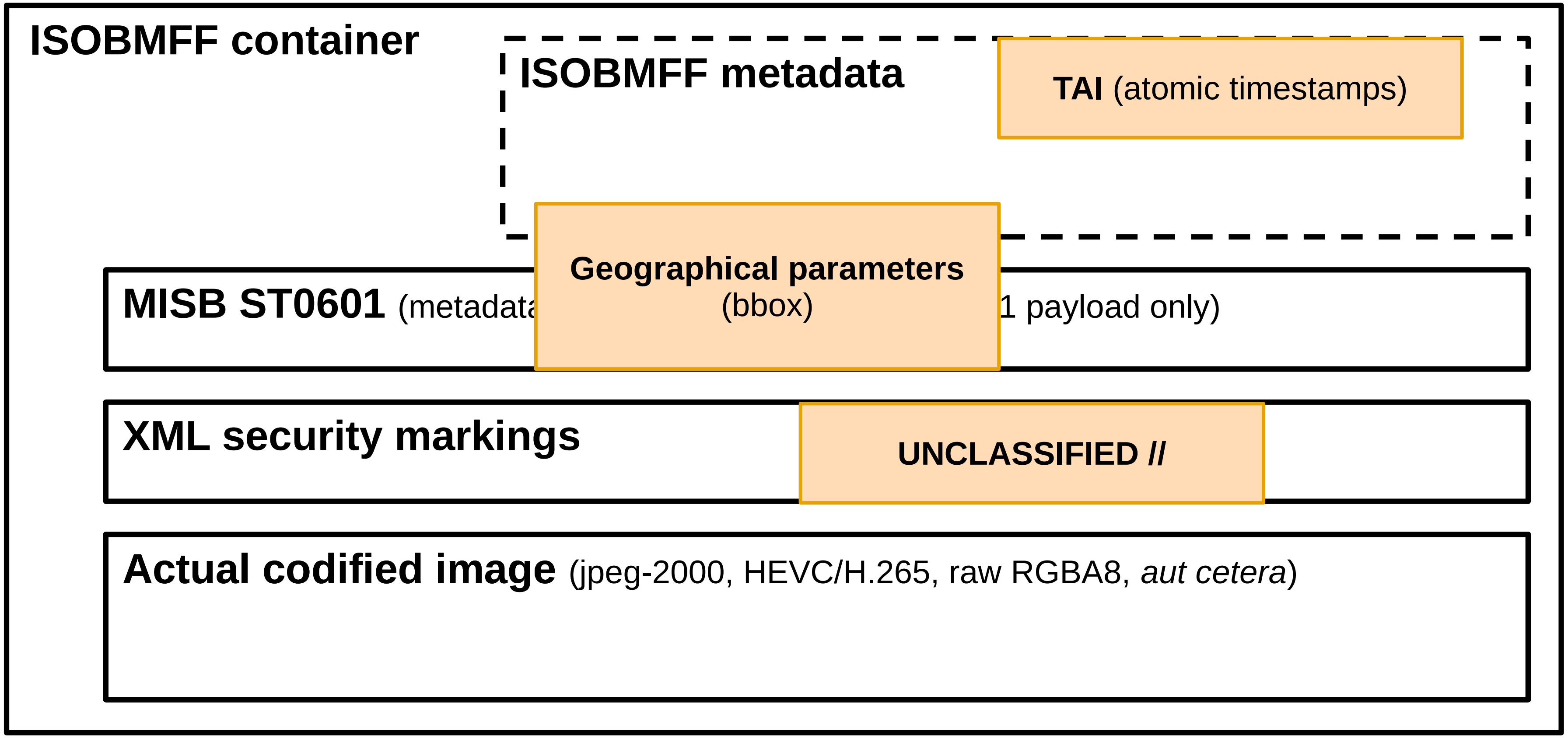
My mental model of a GIMI file


ISOBMFF container
ISOBMFF metadata
MISB ST0601 (metadata for unmanned avionics, ST0601 payload only)
XML security markings
Actual codified image (jpeg-2000, HEVC/H.265, raw RGBA8, *aut cetera*)

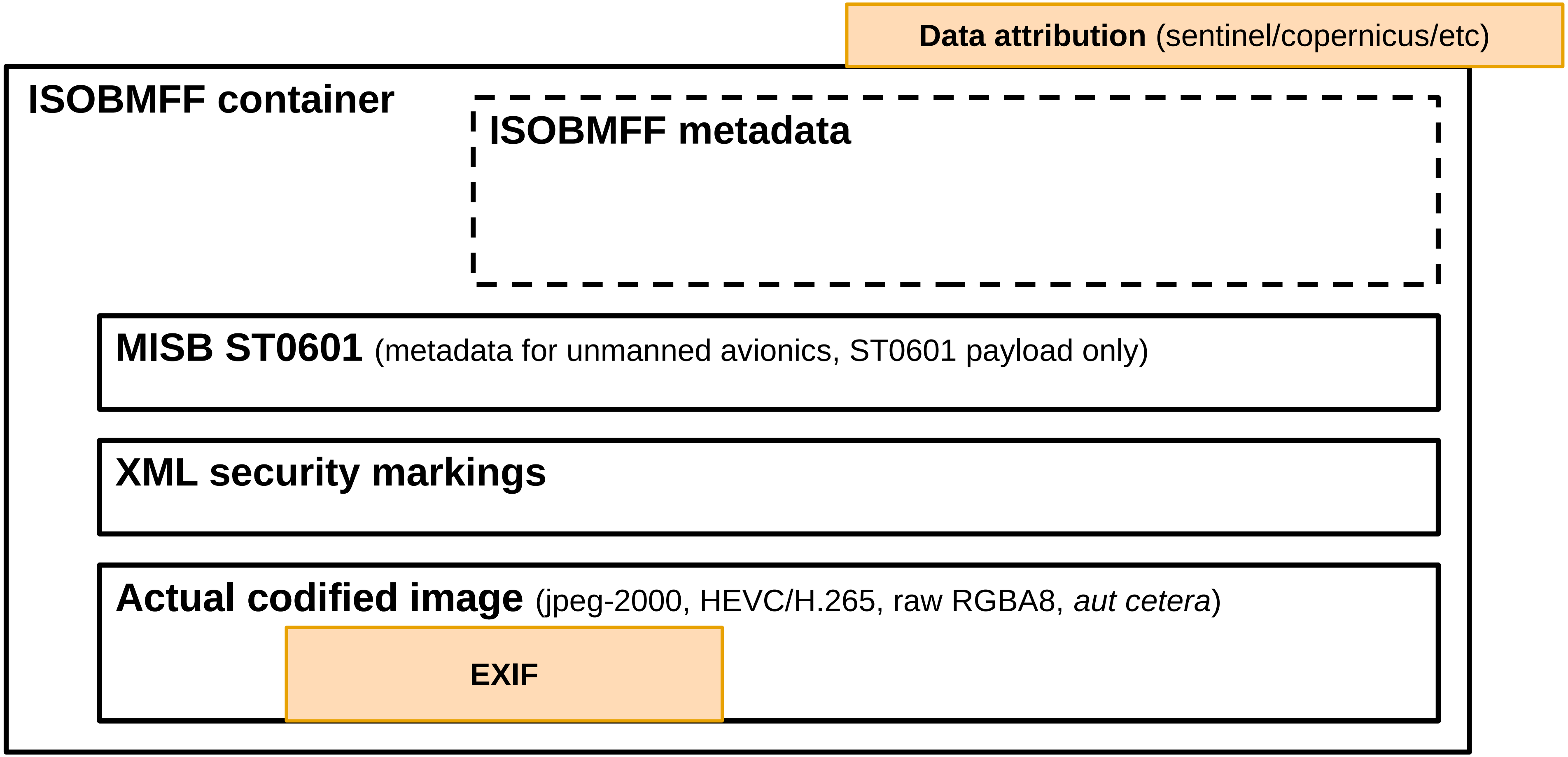
ISOBMFF metadata has both stand-alone info, and references to ST0601/XML/image


ISOBMFF container
ISOBMFF metadata
MISB ST0601 (metadata for unmanned avionics, ST0601 payload only)
XML security markings
Actual codified image (jpeg-2000, HEVC/H.265, raw RGBA8, *aut cetera*)

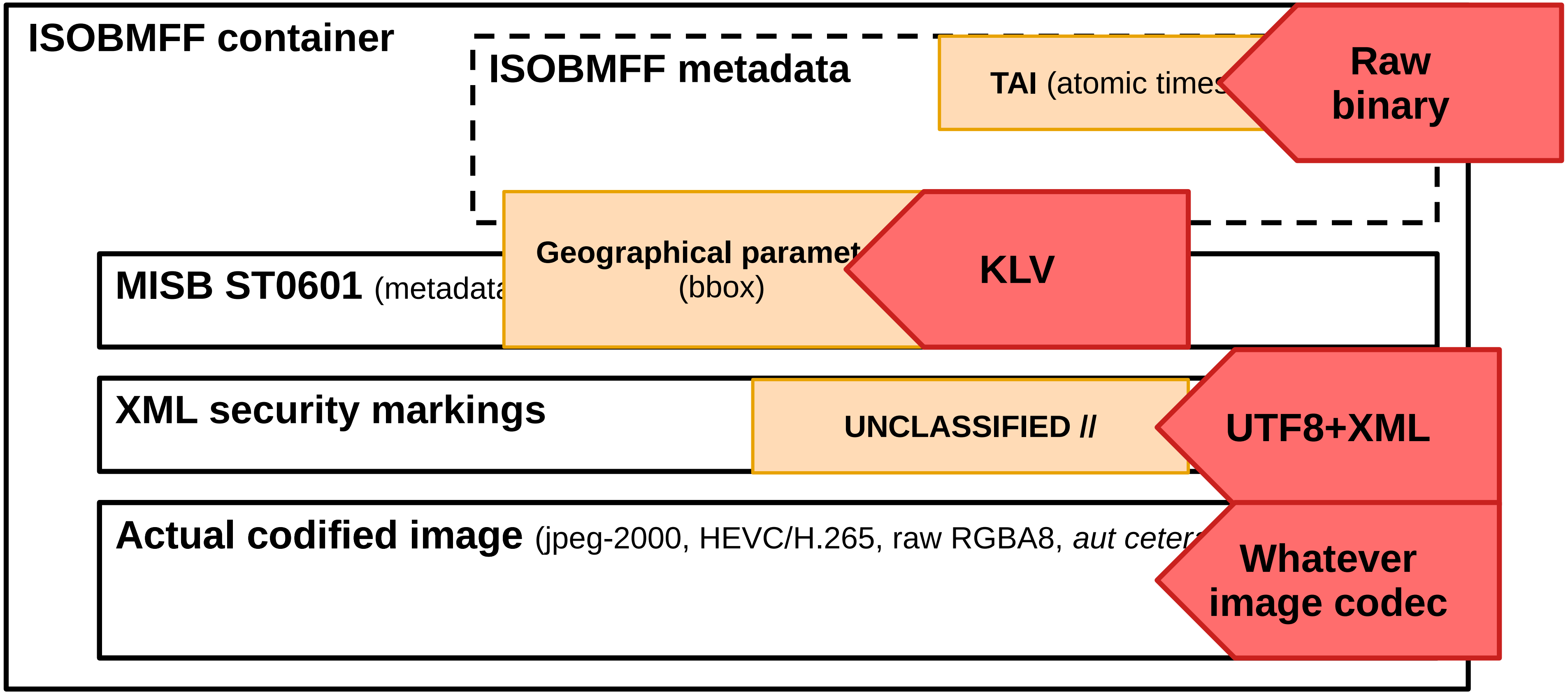
Useful metadata is split along different parts of the file


ISOBMFF container
ISOBMFF metadata
TAI (atomic timestamps)
Geographical parameters (bbox)
MISB ST0601 (metadata
1 payload only)
XML security markings
UNCLASSIFIED //
Actual codified image (jpeg-2000, HEVC/H.265, raw RGBA8, *aut cetera*)

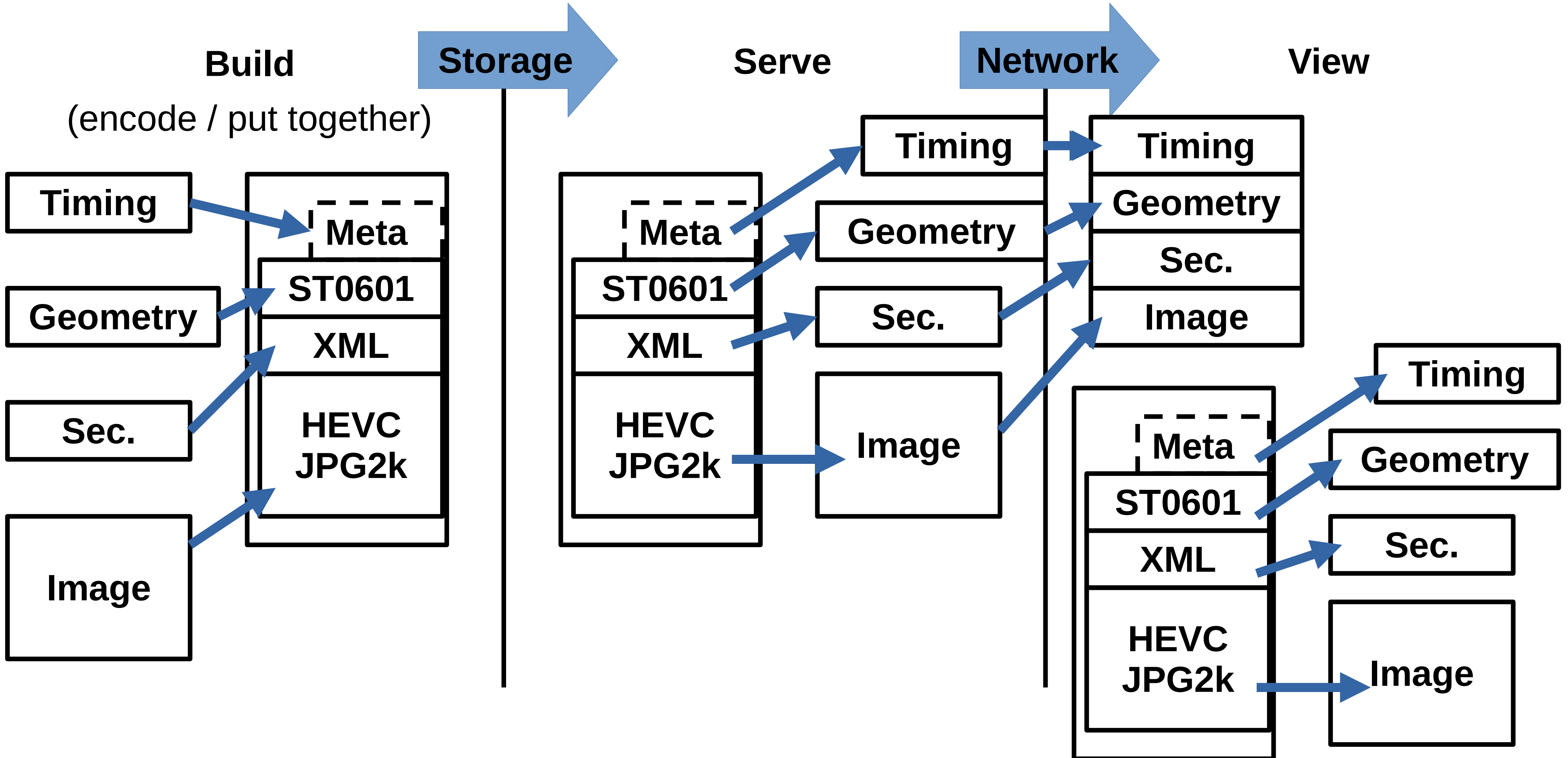
There’s both extra (unused) and missing metadata


Data attribution (sentinel/copernicus/etc)
ISOBMFF container
ISOBMFF metadata
MISB ST0601 (metadata for unmanned avionics, ST0601 payload only)
XML security markings
Actual codified image (jpeg-2000, HEVC/H.265, raw RGBA8, *aut cetera*)
EXIF

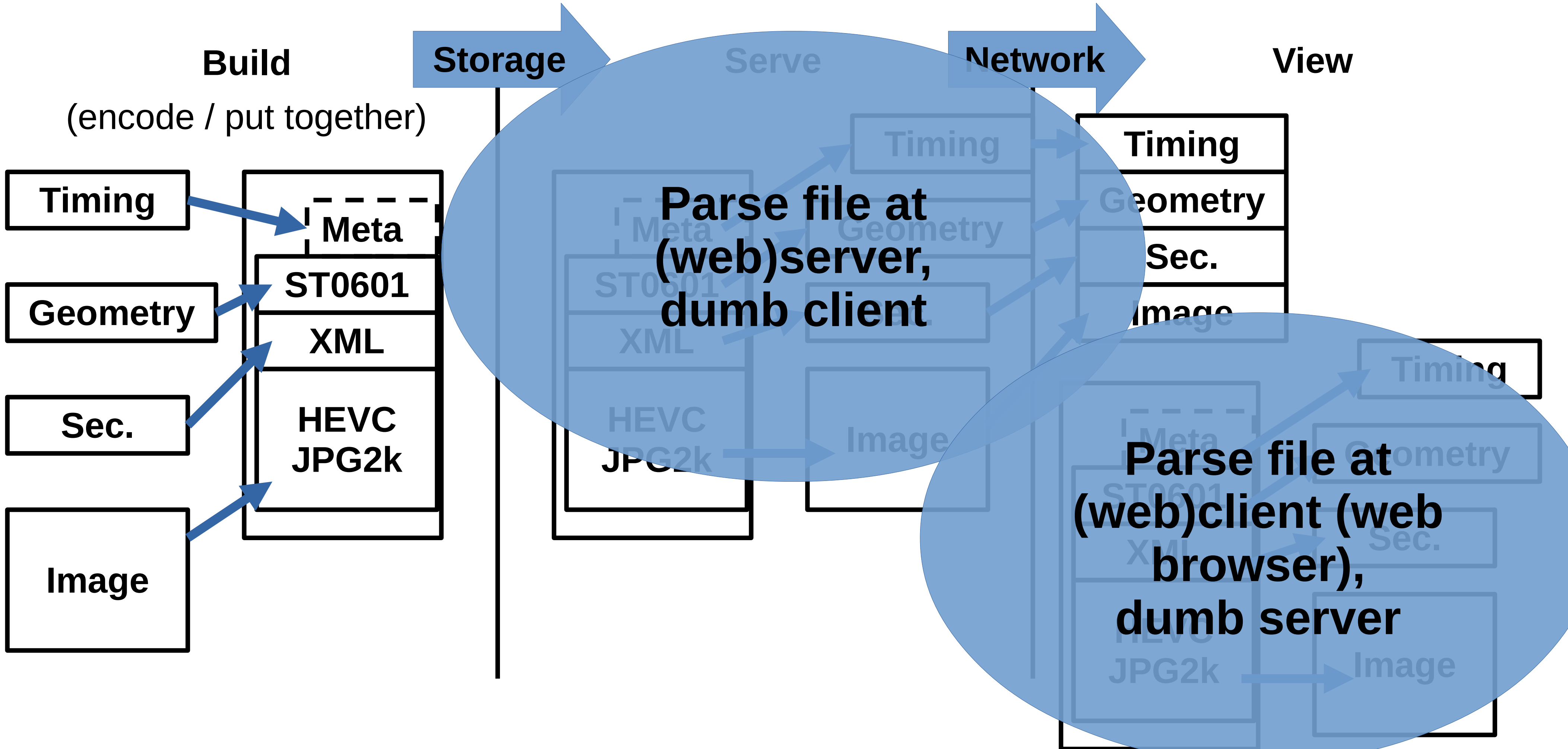
Each piece of metadata uses a different encoding



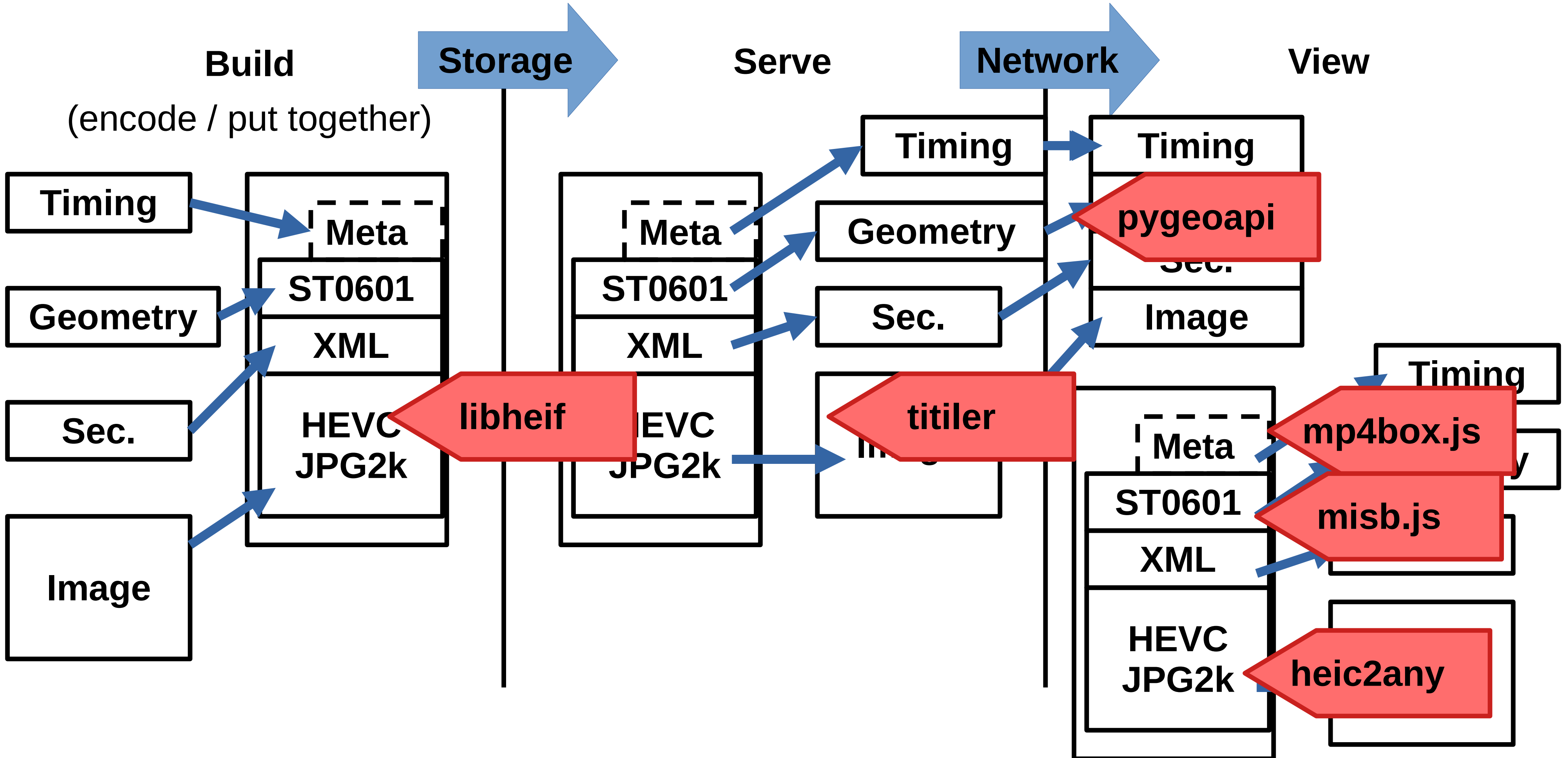
My mental model of the tasks to be done



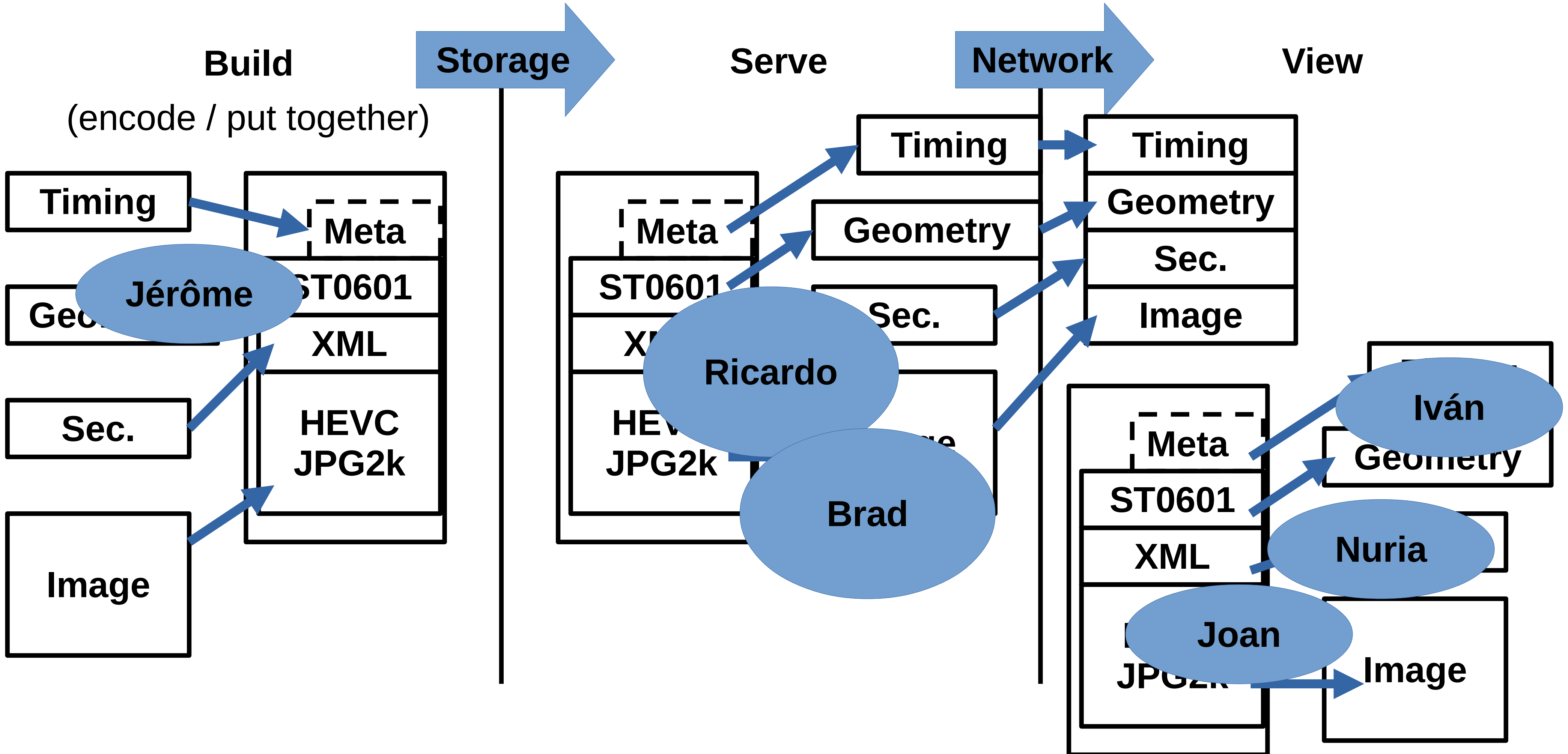
Different strategies to have consumable data



Different tools at different stages



Different people on different problems



Which strategy is “better”?

(decode metadata at server / decode at client)

Do we have tools for all parts of the workflow?

Which strategy is “better”?

(decode metadata at server / decode at client)

Do we have tools for all parts of the workflow?

From a web client perspective, we need a lot of tools & libraries

- ISOBMFF parser/walker
- KVL+ST0601 parser
- XML decoder
- TAI decoder
- image codecs

From a web client perspective, we need a lot of tools & libraries *in Javascript*

• ISOBMFF parser/walker	mp4box.js
• KVL+ST0601 parser	misb.js
• XML decoder	(built-in)
• TAI decoder	
• image codecs	heif2any, etc

From a web client perspective, we need a lot of tools & libraries *in Javascript* *working well in web browsers*

• ISOBMFF parser/walker	mp4box.js	□ buggy
• KVL+ST0601 parser	misb.js	needed work
• XML decoder	(built-in)	easy(ish)
• TAI decoder		trivial
• image codecs	heif2any, etc	fiddly

Specific (sub?)tasks done during codesprint

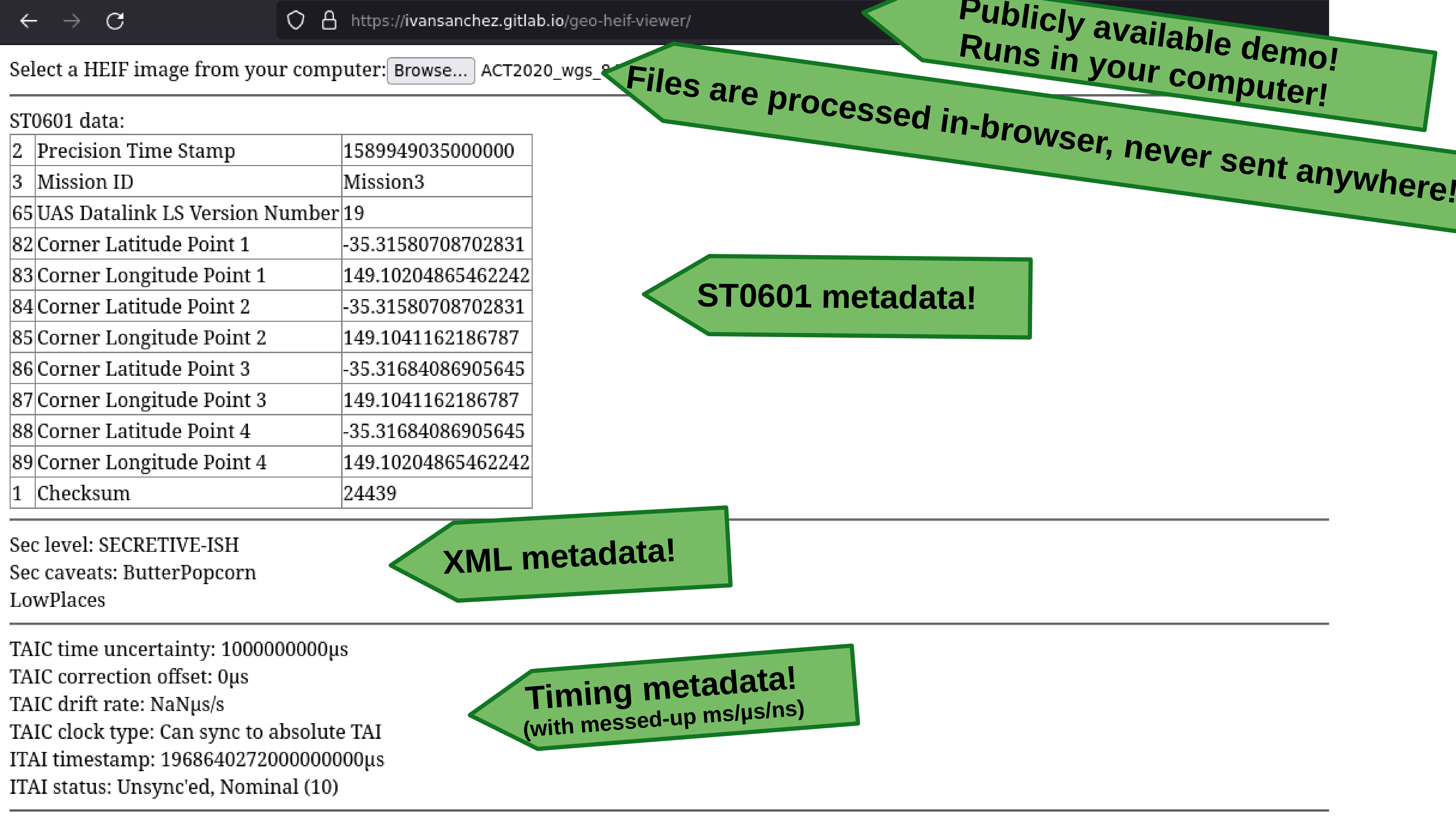
mp4box.js
- Fixed byte alignment bugs (thanks, Brad!)

vidterra’s misb.js
- Implemented browser support

Putting together everything

Introducing:

https://gitlab.com/IvanSanchez/geo-heif-viewer



Publicly available demo!
Runs in your computer!

Files are processed in-browser, never sent anywhere!

Select a HEIF image from your computer: Browse... ACT2020_wgs_8

ST0601 data:

2	Precision Time Stamp	1589949035000000
3	Mission ID	Mission3
65	UAS Datalink LS Version Number	19
82	Corner Latitude Point 1	-35.31580708702831
83	Corner Longitude Point 1	149.10204865462242
84	Corner Latitude Point 2	-35.31580708702831
85	Corner Longitude Point 2	149.1041162186787
86	Corner Latitude Point 3	-35.31684086905645
87	Corner Longitude Point 3	149.1041162186787
88	Corner Latitude Point 4	-35.31684086905645
89	Corner Longitude Point 4	149.10204865462242
1	Checksum	24439

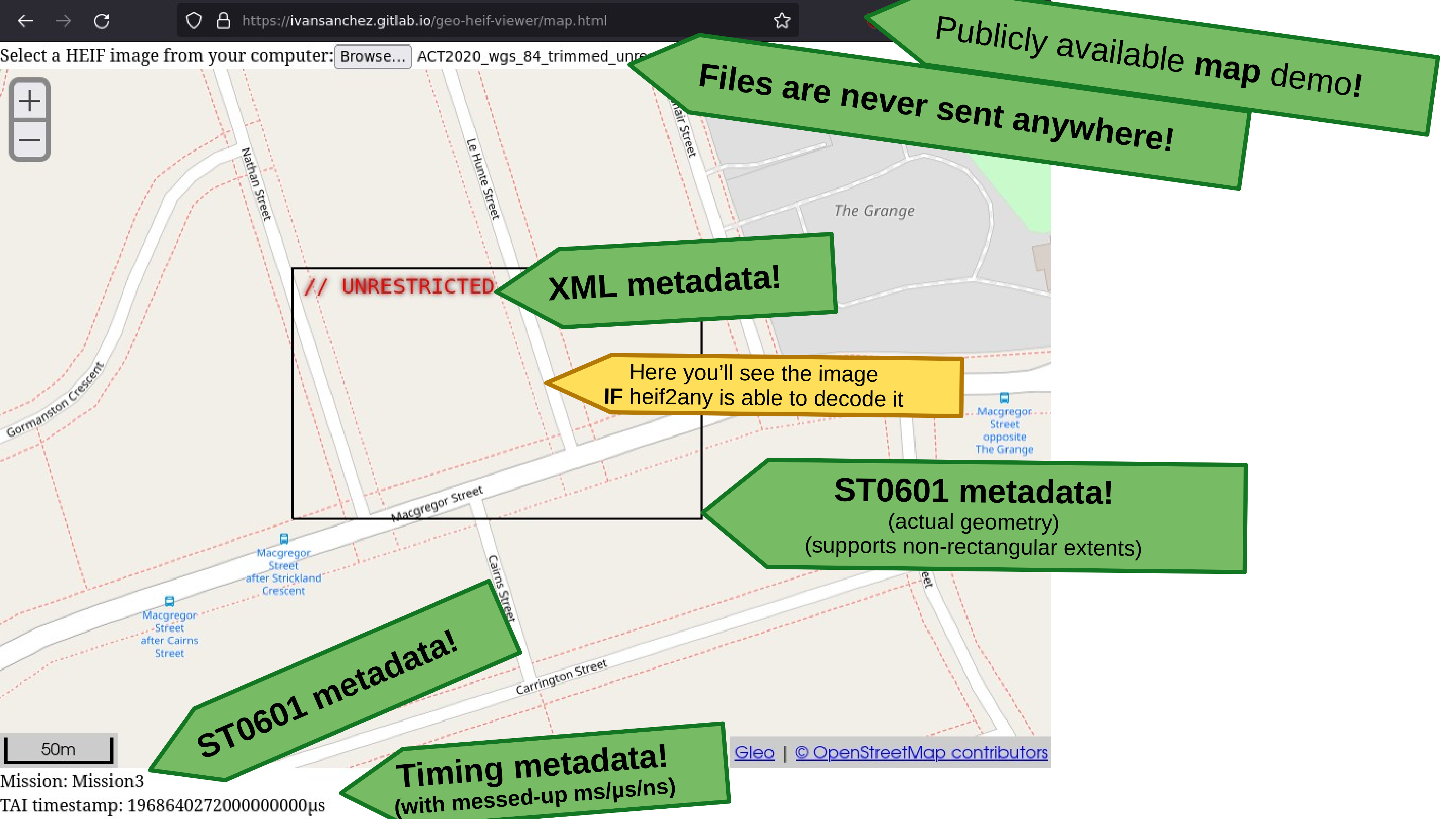
ST0601 metadata!

Sec level: SECRETIVE-ISH
Sec caveats: ButterPopcorn
LowPlaces

XML metadata!

TAIC time uncertainty: 1000000000µs
TAIC correction offset: 0µs
TAIC drift rate: NaNµs/s
TAIC clock type: Can sync to absolute TAI
ITAI timestamp: 1968640272000000000µs
ITAI status: Unsync'ed, Nominal (10)

Timing metadata!
(with messed-up ms/µs/ns)



https://ivansanchez.gitlab.io/geo-heif-viewer/map.html

Select a HEIF image from your computer: Browse... ACT2020_wgs_84_trimmed_unre

Publicly available **map** demo!

Files are never sent anywhere!

// UNRESTRICTED

XML metadata!

Here you'll see the image
IF heif2any is able to decode it

ST0601 metadata!
(actual geometry)
(supports non-rectangular extents)

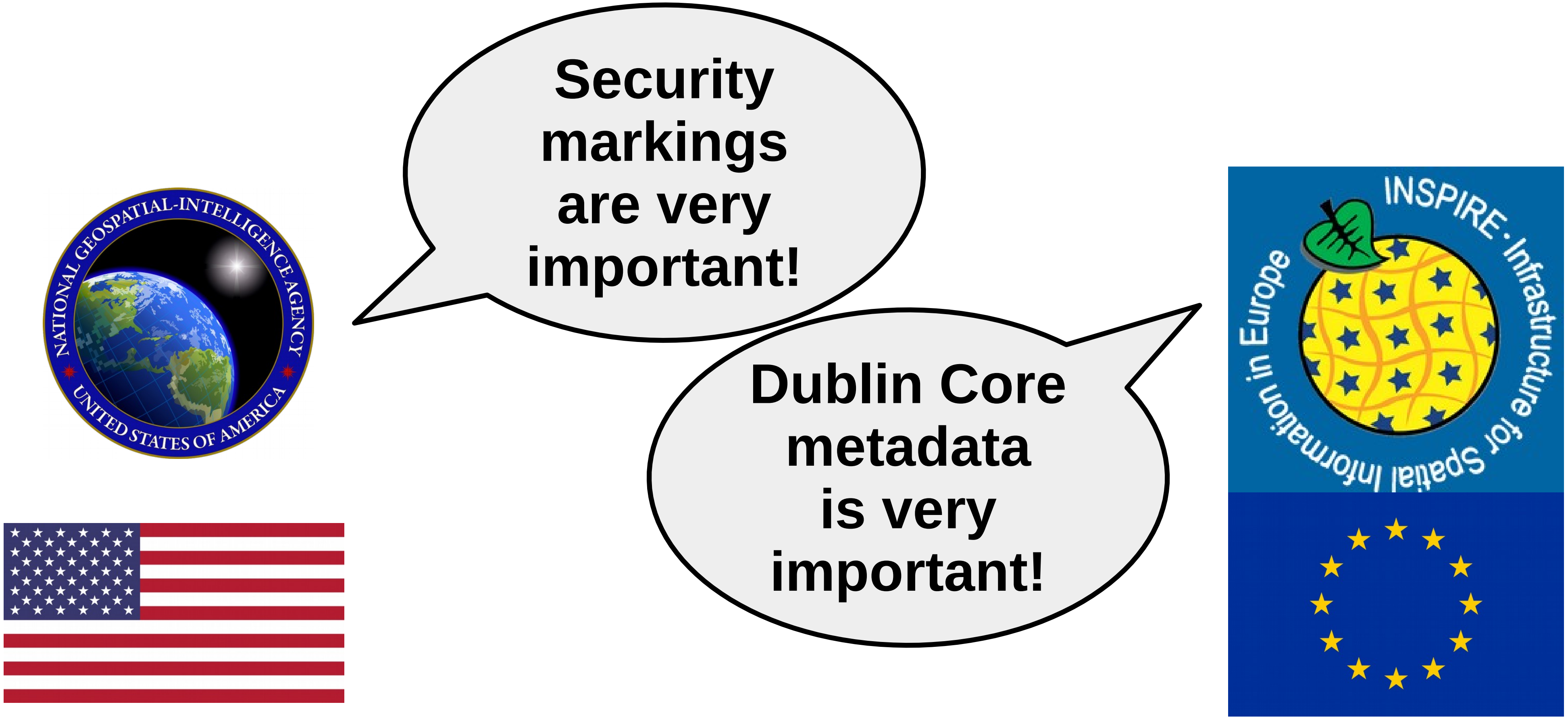
ST0601 metadata!

Timing metadata!
(with messed-up ms/µs/ns)

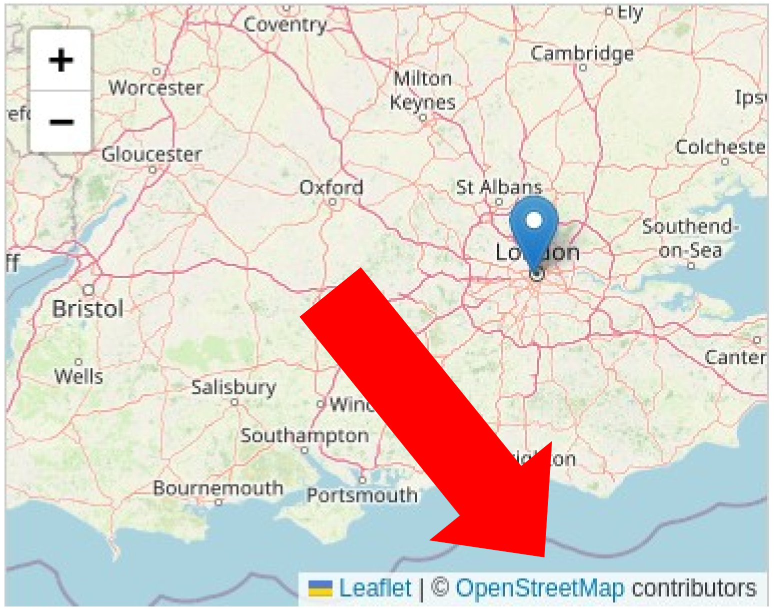
Mission: Mission3
TAI timestamp: 1968640272000000000µs

Nathan Street
Le Hunte Street
Street
The Grange
Gormanston Crescent
Macgregor Street
Macgregor Street opposite The Grange
Macgregor Street after Strickland Crescent
Macgregor Street after Cairns Street
Cairns Street
Carrington Street
50m
Gleo | © OpenStreetMap contributors

A bit of GIMI criticism


Security markings are very important!
Dublin Core metadata is very important!
NATIONAL GEOSPATIAL-INTELLIGENCE AGENCY
UNITED STATES OF AMERICA
INSPIRE · Infrastructure for Spatial Information in Europe

What do I put in the corner?
Is it ©? CC-by-sa? other?


+
−
Coventry
Ely
Cambridge
Worcester
Milton Keynes
Gloucester
Oxford
St Albans
Colcheste
Southend-on-Sea
London
Bristol
Wells
Salisbury
Southampton
Bournemouth
Portsmouth
Canter
Leaflet | © OpenStreetMap contributors

GIMI does not specify attribution/access

Dublin Core does not specify “classified” markings

There is no consensus on attribution markings

HTML? UTF-8? logotypes?

Some random ideas

- Storing tiled images in a single GIMI file
  - Like COG does (with TIFF overviews) but with ISOBMFF frames/boxes/whatever
- Duplicated metadata
  - Bounding box implicit in name/URL (e.g. OGC API tiles) and ST0601
  - Possibly band info + bit depth in ISOBMFF metadata and image codec metadata
- Why not GeoTIFF?
  - It also supports arbitrary metadata blocks
- Tooling still needs lots of work
  - There’s a bit too many moving pieces
  - Work needed for COG-like range requests in mp4box