FILLING IN THE GAPS IN MAPS: EMBRACING OPEN GIS STANDARDS IN PLANETARY SCIENCE.

A. M. Annex[1], T. M. Hare[2], J. R. Laura[2], N. Manaud[3], J.-C. Malapert[4], B. Seignovert[5] [1]Division of Geological and Planetary Sciences, California Institute of Technology, MC 150-21, 1200 E. California Blvd., Pasadena, CA 91125, USA (annex@caltech.edu), [2]USGS Astrogeology Science Center, Flagstaff, AZ, [3]SpaceFrog Design, Toulouse, FR, [4]CNES, Toulouse, FR, [5]OSUNA/CNRS-INSU, Nantes, FR.

Introduction: The past two decades have seen the proliferation of GIS standards all with the aim of making data easier to access. The Open Geospatial Consortium (OGC) alone has published more than 30 standards for representing geospatial information. Planetary science as a field has been slow to adopt these standards due to complexities inherent to the field, the fact that many of the standards are developed with an Earth-centric perspective, and the effort it takes to update the standard to work for extraterrestrial bodies. Fortunately, recent standards have been modified to now allow for a broader planetary use. But, the challenge remains to make sure these planetary-enabled standards are also usable across popular software applications. This is particularly significant as the advent of cloud computing and cloud-ready data formats has presented new possibilities for making it easier than ever to distribute and utilize geospatial data easily and efficiently.

The inherent complexity of open source geospatial software presents some advantages and disadvantages. The major disadvantage, complexity, means that there is no one codebase to update, and knowledge of the full breadth of open geospatial software is required to identify needed changes. The advantages are twofold: the interdependency of open software means that small changes can ripple across the ecosystem, and that changes can be made and adopted rapidly.

In late 2022, an organic discussion was held between members of the OpenPlanetary organization [1] to identify problems in existing software and standards that could be solved efficiently. Below we report on some of the findings from the meeting, progress made so far, and paths forward towards resolving these issues.

IAU CRSs: An issue identified early in the meeting was the lack of standardized names for coordinate reference systems (CRSs) for planetary bodies. For Earth, standard CRSs such as EPSG:4326 and EPSG:3785 (and older EPSG:900913) exist, are well known, and natively available in most GIS software. While the International Astronomical Union (IAU) defines spheroids and ellipsoids for planetary bodies [2], well known identifiers have not been adopted by the community broadly resulting in confusing incompatibilities in GIS software with data defined with slight metadata differences. Fortunately, a prior effort by the planetary community and now the OGC Planetary domain working group (DWG) have created a set of over 4000 CRSs available at http://voparis-vespa-crs.obspm.fr:8080/web for all planetary bodies with a consistent naming convention. The creation of this list of CRSs enables the adoption of their names as a standard against which other GIS software can be tested.

Naming Authority Updates: Although the Planetary DWG has created this list of IAU CRSs, they also should be registered with the OGC's Naming Authority (OGC-NA). In support of this, the OGC has recently proxied an online planetary CRS registry hosted by the National Centre for Space Studies (CNES). Scripts to create this set are also hosted on GitHub (https://github.com/pdssp/csvforwkt). The OGC-NA is used by libraries like OSGeo's PROJ to validate and retrieve CRS definitions (https://github.com/OSGeo/PROJ). Once registered under the IAU authority, the IAU CRSs discussed earlier will be in equal standing to those from other authorities like EPSG.

TileMatrixSets: The OGC Tile Matrix Set (TMS) standard defines a way to specify regular grids for various CRSs and map projections that clients and servers can use. The *morecantile* project (https://github.com/developmentseed/morecantile) was used and patched twice as part of this effort in order to produce a preliminary set of 115 TMS specs from the IAU CRSs which includes Mercury, the Moon, Venus, Mars and numerous moons with equatorial and polar map projections. Work remains to resolve issues with generating TMS specs for a subset of bodies including Pluto and resolving upstream issues with defining the IAU CRSs with the OGC Naming Authority. The TMS specifications that were made are freely available at https://github.com/AndrewAnnex/planetcantile.

RGB Terrain Tile Encodings: Elevation data intended for use in WebGIS clients are typically provided as PNG formatted image tiles with the data quantized lossily from 32 bits to 24 bits to make efficient use of bandwidth. The popular Mapbox encoding for Earth terrain tiles has a set vertical precision of 10 cm. We identified that a quantizer for terrain data is defined only by two parameters: the minimum elevation to encode and the precision desired. We defined a set of quantizers for all inner planets and the Moon based on their elevation ranges with respect to their geoids with precisions finer than 2 mm available at https://github.com/AndrewAnnex/topography-quantizer-spec. We also defined a universal

encoding for rocky bodies that can represent all topography from -12000 meters in the inner solar system with 2.04682 mm vertical precision.

Cloud Optimized GeoTIFF (COG): A cloud optimized GeoTiff (COG) is fundamentally the familiar and ubiquitous GeoTiff. COGs extend the GeoTiff specification to allow for HTTP GetRange requests on some or all of the underlying data (https://cogeo.org). This is accomplished by placing additional information into the GeoTIFF header (https://gdal.org/drivers/raster/cog.html) and by making use of pyramid overviews whereby GET requests access and stream lower resolution representations of the data when appropriate (e.g., when the client is zoomed out). Since a COG is a GeoTiff, at full resolution it faithfully represents the original data (subject to any processing that has been applied such as calibration). COGs are a fundamental component of the cloud and web-based geospatial stack because of their ability to be efficiently tiled, compressed, and streamed over the web.

STAC: The Spatio-Temporal Asset Catalog (STAC) is an open standard for defining and distributing collections of geospatial data and metadata so that these data can be indexed and searched easily (https://stacspec.org/). The STAC API is compliant with the OGC API Features standard. The standardization provided by the STAC specification means that Application Programming Interfaces (APIs) are easily developed to support data discovery. The standard supports spatial and CQL based queries. Extensions to STAC can be written by the community to add new functionality, and a Solar System Extension Specification (SSYS) has already been created to add planetary bodies as a target field to STAC catalogs (https://github.com/thareUSGS/ssys). And fortunately, the current raster format of choice for STAC catalogs is the previously described COG format.

Many open source projects already include support for STAC including QGIS which enables rapid analysis of data provided through STAC. Both STAC server (https://github.com/stac-utils/stac-server) and Franklin (https://azavea.github.io/franklin/) implement the STAC API specification. One example of a publicly available STAC server hosting planetary data is https://stac.astrogeology.usgsgov/api/. RESTO (https://github.com/jjrom/resto) is another open source server for STAC. PySTAC (https://pystac.readthedocs.io/) is a python library for working with STAC programmatically.

TiTiler: TiTiler is an open source python framework for creating dynamic tile servers (https://developmentseed.org/titiler/). TiTiler acts as a backend server for STAC catalogs, Cloud Optimized GeoTiffs (COGs), Web GIS frameworks and clients, and can serve data to clients using the TMS specification discussed earlier. The USGS is preparing a publicly accessible endpoint (URL to be released) where users can submit HTTP requests that provide URIs for COGs and receive properly and dynamically tiled image data as a response. Dynamic tile servers are beneficial as they help minimize maintenance costs by adopting the serverless computing paradigm (https://en.wikipedia.org/wiki/Serverless_computing).

Future Needs: With the initial IAU CRSs propagated throughout the ecosystem of open source GIS software a number of future directions can be anticipated. For example, adding support for triaxial ellipsoids within PROJ would enable more accurate distance and area calculations, and would allow additional TMS specification for oblate icy moons and other small bodies. The creation of a community curated list of STAC catalogs and collections would enable scientists to quickly find and contribute data that is potentially analysis-ready across a suite of GIS tools.

The improvements and backend architecture for serving data described above make it possible to consider additional future changes to the ecosystem of popular web-based visualization software to better support planetary data. These include a number of open source javascript libraries like Deck.gl, Maplibre, Leaflet, Openlayers, Proj4js, CesiumJS, and geotiff.js. These software often use Earth specific constants such as radius and Mercator map projections, but the IAU CRSs make it possible for these projects to adapt their code to use the specific body constants being examined in a standardized way.

Conclusions: This work describes the current state of the cloud-based planetary data ecosystem, identifies key components of the ecosystem defining a software stack, and describes initial work to ensure that planetary data are well supported. One can track a path through mutually supporting standards to find a place where cloud enabled planetary spatial data simply work in off-the-shelf geospatial software. This starts with the adoption of IAU CRSs, adding support for non-EPSG projection authorities to software tools, adopting TMS and RGB encoding where appropriate and ensuring those tools support planetary data, transitioning to COGs from ordinary GeoTIFFs, using STAC metadata for raster data and OGC Feature API compliant solutions for vector data, and finally modifying front end visualization libraries to make use of these adopted standards. Although the path is long, it is a viable approach as all of this work happens across open source projects with, in our experience, maintainers that are excited to support planetary use cases.

References: [1] Manaud N. et al. *EPSC* (2022) doi: 10.5194/epsc2022-48. [2] IAU report (2015) doi: 10.1007/s10569-017-9805-5.