MULTI-MISSION GEOGRAPHIC INFORMATION SYSTEM: AN OPEN SOURCE SOLUTION FOR PLANETARY SCIENCE OPERATIONS.

F. J. Calef III1, H. E. Abarca1, T. Soliman2, S. P. Abercrombie1, and M. W. Powell1, 1Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Dr, Pasadena, CA 91109, Fred.Calef@jpl.nasa.gov, Hal-lie.E.Gengl@jpl.nasa.gov, parker.abercrombie@jpl.nasa.gov, mark.powell@jpl.nasa.gov 2California State University Northridge, Northridge, CA, tariq.soliman@jpl.nasa.gov.

Introduction: We’ve completed our three-year project developing the multi-mission geographic information system, MMGIS, as a new operations tool for the NASA Advanced Multi-Mission Operations System (AMMOS). MMGIS provides geospatial tools as part of a spatial data infrastructure for planetary science operations. This series of programs, workflows, and web-based interfaces unite mission basemaps with science products in their proper geospatial context with ‘quicklook’ instrument results. The software has now been deployed on two active missions, the Mars Science Laboratory rover and the InSight lander, and is in nearing completion of an adaptation for the Mars2020 rover as a strategic planning tool. In effort to provide a low-cost, effective solution for planetary missions, this software is now released as an open source project.

Science Data Product Localization: MMGIS separates science data product localization into three categories: fixed-body, mast pointed, and arm attached (i.e. ‘Contact Science’) instruments (Figure 1). We’ve developed methodologies and workflows using Python and VICAR [2] for localizing/georeferencing science instrument data by linking spacecraft clock (SCLK) data labels to spacecraft localization, via translation from ij image space to ground projection, inverse-kinematics, or using the planned target database as a proxy for instrument sample location for all observations. This science product localization information is translated from the instrument coordinate frame to a spacecraft-centric coordinate frame, then into a global coordinate system that matches a mission basemap [2]. Once collected, mapping coordinate values are then produced in a format conducive to a web-based mapping interface. At the moment, most of these scripts are tuned to surface missions (e.g. MSL), though with some effort could be abstracted more to orbiters and their instruments.

Methods: Our goal was to minimize user and administrative install/upkeep for the MMGIS software package. Initial development used open source tools such as Python (python.org), GDAL (gdal.org), Leaflet (leaflet.org), D3 (d3js.org), three.js (threejs.org), and SQLite3 (sqlite.org) for rapid development of a flexible data pipeline and intuitive user interface with a short learning curve (Figure 2). Mission requirements for Mars2020 led us to expand the programs to work with node.js and Postgresql/POSTGIS to aid in cloud deployment and scalability, as well as offering a more ‘modern’ programming workflow.

The web-based mapping interface eliminates the need to download/install gigabytes of software and datasets, typical of previous operations software solutions. MMGIS supports both ‘traditional’ server based implementations as well as cloud solutions (e.g. AWS). The MMGIS web interface can be deployed via an Apache webserver with PHP or using node.js. Vector and raster data are served directly as GeoJSON format (geojson.org) and image data pre-cut as Tile Map Service (TMS) tiles (https://wiki.osgeo.org/wiki/Tile_Map_Service_Specifica-tion) in PNG format, respectively, achieving a near “serverless” configuration. GDAL is used for file translations into the aforementioned formats (ogr2ogr and gdal2tiles.py). Raw values are queried directly from map-projected files without translation GDAL Python programs. A SQLITE database (sqlite.org) is implemented for storing user annotations and administrative configuration, though newer implementations in development are utilizing PostgreSQL/POSTGIS as a true spatial database supporting a larger user community. The software depends only on Python (2.7+), GDAL (2+), and PHP (5.+ ) for server-side processing, though we’re rapidly moving functions to node.js, towards dropping the PHP and other dependencies.

Interface: Science operations lives on minimizing engineering time (e.g. “did the rover drive complete?”), “where’s the latest surface mosaic?”) and maximizing science time (i.e. “we’re seeing high calcium and sul-
interface allows measuring distances, creating 3D profiles, and viewing targets on ground mosaics, all interlinked, providing rapid review and quick analysis (Figure 2). The main interface is a 2D map view in web Mercator projection. Linked to this map is a 3D “Globe” view developed in three.js and an “InSitu” viewer for displaying ground mosaics and other datasets linked to science targets, such as images and 3D object models. A custom ‘Chemistry’ tool displays a graph of “quick look” results, e.g. from the CHEMCAM or APXS instrument onboard the Curiosity rover. The modular structure of MMGIS allows adding on additional tools, building on the existing framework.

Mission Operations: MMGIS has been deployed on both the Curiosity rover mission and adapted to the InSight lander mission, both at Mars (Figure 3). For Curiosity, the MMGIS instance serves as a way for science team members to see rover traverse progress, view the most recent ground mosaics, and where science targets were acquired. In addition, its been used successfully for strategic rover planning to help develop a safe route up Mt. Sharp given science desiresment for sampling specific geologic outcrops and evaluating mobility requirements (slope, terrain types). On InSight, we added an ‘Instrument Placement’ tool that was used to visually and qualitatively assess deployment of the seismometer (SEIS) and heat probe (HP3) instruments to the surface. We’re currently adapting MMGIS for the Mars2020 rover mission called CAMP (CAMpaign Mapping and Planning) for strategic science operations that will be a fully Dockerized and cloud-deployed version using a RDS PostgreSQL/POSTGIS database to store science team member spatial annotations as well as strategic science targets and drive paths.

Open Source Release: Our first open source release has been posted on github under an Apache 2.0 license: https://github.com/NASA-AMMOS/MMGIS. Periodic updates will be made several times a year. We’ll accept pull requests and merge branches into the master, after internal code review for consistency, security, and pertinence.

Future Work: Full migration to a node-based architecture is expected for most tools, moving away from the Apache webservice/PHP model. Our ‘Drawing Tool’ will be converted over to using PostgreSQL/POSTGIS as well as other updates for cloud support (e.g. better support for datasets in S3).