USGS Astrogeology Science Center Response to “MAPSIT Special Action Team Report on Cartography and Image Processing Software at the Astrogeology Science Center” dated November 30, 2018

Provided to:
Dr. Sarah Noble
NASA-USGS Interagency Agreement Program Manager

March 15, 2019

Table of Contents

1 Introduction

The review of the Astrogeology Science Center (ASC) software portfolio by the Special Action Team (SAT) of the Mapping and Planetary Spatial Infrastructure Team (MAPSIT) was extremely valuable and greatly appreciated. The response by the ASC was delayed due to the government shut-down from December 22, 2018 – January 27, 2019. The ASC agrees with the vast majority of these findings. Some of these findings have been addressed in the time that the ASC met with the SAT (August 2018), and the status of those actions is described in this report. Proactive steps to address the additional findings are also described. The SAT finding report is publicly available on the MAPSIT website.

2 Findings

2.1 Planning

F1 Strategic Plan

"We find that there is no publicly available long-term strategic plan for the ASC software portfolio.

An overall long-term publicly available strategic plan would support other Findings about community participation and transparency. It would also help the ASC itself to understand the alignment between the direction of their software portfolio development and NASA missions, directions, and goals, and help them to ensure that their software portfolio is supporting the concepts of planetary spatial data infrastructure.

It would be reasonable for the ASC to develop such a plan within a year."

ASC Response:

The ASC is in complete agreement with this finding. A long-term plan for the ASC software portfolio was provided to the SAT at the time of review and was based on ASC team-wide discussions regarding software priorities for a wide range of user needs. We consider this plan to be a living document that will be continually modified as software development work is completed, to indicate software efforts that are currently under development, and to be responsive to the changing needs of the planetary science community and the strategic direction of the ASC. This long-term plan will be made publicly available on the ASC website no later than September 2019, which is ahead of the requested timeline by the SAT, and this public version will be updated annually.

F2 Existing functionality and future deprecation decisions

"We find that it is important to identify what the core ISIS functions are, to maintain existing versions of the software past the time when new versions are introduced, and to ensure that existing functionality not be deprecated unless alternate solutions are common, broadly accessible, and equally capable.

The SAT notes that significant changes are occurring or planned with the ISIS code base, including modularization of libraries, implementation of new interfaces, incorporation of new core libraries (e.g. ray tracing engines), etc. Given the broad user community, many of which have existing workflows and pipelines, we find that it will be important to maintain existing versions of the software for a period of time, even when new versions are introduced, until the user community has embraced the new versions.

In the context of libraries, it will be critical that core ISIS functions continue to exist and function as they do currently while their internal components are switched to use libraries. The definition of "core" should be defined by polling ASC developers, internal users, missions, and the community at large. It might be best to err on the side of a large pool of "core" functionalities, rather than one that is too limited. This would best smooth the transition to a library-based, modularized ISIS and toward an open-source community.

In the context of graphical user interfaces (GUIs), while there is broader expertise in the community outside of the ASC for creating GUIs that can be wrapped around core ISIS functionality, and the ASC may be considering this as an area from which to eventually step away from (so their resources can be deployed elsewhere), the existing level of GUI functionality (e.g. qview, qmos, etc.) should not be deprecated unless alternative solutions are equally capable."

ASC Response:

The ASC agrees with this finding. We are currently taking steps to modularize the ISIS code base (see response to finding F8 for additional discussion). The ASC also agrees that the "definition of core should be defined by polling ASC developers, internal users, missions, and the community at large," and we believe that deeper conversations with a representative sample of our primary customers would result in more meaningful information than a survey to the broad community. These conversations would result in information that would allow the ASC to clarify the core functionality (e.g., geometric image placement, long-term sensor support, control network generation, image calibration, photometric corrections) that the community needs ISIS to continue to support. These customers will include, but are not limited to, representation from

internal users (including processors, scientists, and the ASC's PILOT data processing engine), missions, and general ISIS users. We will identify and engage these customers within 6 months of this response.

As the core functionality is defined based on both code functionality and user needs, we will be deprecating existing code that has been replaced by this modularization of ISIS, is no longer of use to the planetary community, or has been replaced by broadly available software with equal or a higher level of functionality. When deprecating code, we will include information in the ISIS release notes and will maintain both applications for a specified period of time, typically for 6 months. In addition, a general code deprecation policy will be made available on the ASC GitHub repository. Regarding the "existing level of GUI functionality (e.g., qview, qnet, qmos, etc.)", we have no current plans to deprecate this code or functionality, as it is a critical component of our processing pipelines.

2.2 Community Interaction and Documentation

F3 User base for ASC tools

"We find that there is an uncertainty about whom the primary user base should be for the ASC software portfolio.

The ASC has largely developed ISIS with a posture clearly focused on experienced technical users comfortable in non-graphical environments, while providing some support for beginners in the form of tutorials and workshops. A continued primary focus on technical users is logical and appropriate, as this allows the ASC to focus their efforts on building software that solves technical problems needed to analyze planetary data that is unique to their skill set and essential for scientific analysis. For this reason, building new graphical applications that attempt to unify the current breadth of ISIS applications into a singular super-GUI that tries to offer a comprehensive interface for the majority of the existing tools for non-expert users is not an avenue that the ASC should pursue. The current level of per-application graphical interfaces is adequate.

The core functionality of ISIS should still ship as applications or tools usable without requiring extensive additional coding. In other words, just shipping libraries and application program interfaces (APIs) without some user applications is insufficient. Perhaps the Geospatial Data Abstraction Library (GDAL) example is a good one: primarily GDAL is a software library, and there are a large variety of operations that can be accomplished with the library for a user willing to write custom software. However, GDAL also ships with a number of command line applications that provide substantial functionality easily via the command line.

The ASC should continue its primary focus on technical users."

ASC Response:

The ASC agrees with this finding. With the development of significantly more data analysis tools, both from mission teams and other groups that depend on the ISIS API to develop their software, the user community of ISIS has grown more complex in recent years. The evaluation of how our full user base interacts with our software portfolio will be pursued at least annually so that we maintain a strong connection to our users and will inform decisions of how to best support our customers. The conversations with our user community described in finding F2 will serve as a test for how to best engage our users. We will continue to distribute ASC core functionality as applications or tools usable without requiring extensive additional coding. The GDAL example provided by the SAT review provides a good model for the ASC to follow. The finding to continue our primary focus on technical users is well-received, however "technical user" has not been well-defined. We would interpret this set of users to include a broad group of those generating higher order data products (e.g., controlled mosaics), those using the ISIS API to develop tools or custom software, mission teams (e.g., sensor models and data processing pipelines), scientists, and other users.

GUI interfaces are not the ASC's primary focus of development, and we do not foresee this focus changing moving forward. However, we have no current plans to stop supporting the existing GUI applications. The GUI interfaces currently available (qview, qnet, qmos, etc.) will not be deprecated without broadly accessible and equally capable alternatives that have been fully tested with commonly used planetary data formats and existing work pipelines (see also finding F2).

F4 Open Source, Community-Driven Development Model

"We find that the ASC should continue to be the steward of ISIS and future planetary spatial data infrastructure software releases, but that more effort must be made to facilitate broad participation by external contributors.

The 2014 ETRP (External Technical Review Panel) made a similar finding. While there is evidence that the ASC is moving in this direction, we expected to see more progress in this area after four years. The ASC development model should be as the steward of a set of open source packages, meaning that they should "embrace open exchange, collaborative participation, rapid prototyping, transparency, and community development" as was stated in their own aspirational documents in 2014.

We use the term ‘steward’ here instead of the term ‘gatekeeper’ used in the 2014 ETRP. ‘Gatekeeper’ implies defensiveness, while ‘steward’ invokes service to the community. While executive decisions about the course of the software should be made at the ASC, those decisions should be guided by a strong connection to the community of users. Recent discussions with the ASC indicate that they are considering assembling a technical committee consisting of external experts in order to address this need, we support and encourage that course of action.”

ASC Response:

The ASC is in complete agreement with this finding and has initiated an Open Source community surrounding the ASC software portfolio. Our intention is to build a community of users and developers with the shared goal of providing accurate, high quality software infrastructure and tools in support of product generation and planetary science data analysis. We intend to grow and foster the development of the Open Source community following the established guidelines for contributing and by adhering to a code of conduct that encourages an open and welcoming environment with inclusive and diverse participation by the planetary community. In addition, the ASC software portfolio has been publicly available on GitHub since January 23, 2018.

Note: The 2014 ETRP (External Technical Review Panel) can be found in Appendix B of the SAT Findings document.

F5 Community and User Support Forum

“We find that there is currently no good mechanism in widespread use for user engagement with the ASC software portfolio and its developers.

The ASC had previously made available an online forum for its ISIS user community that served two functions. First, it enabled users to interact with ASC developers, ask questions, and report bugs. Second, it enabled experienced users to support fellow users (e.g. by suggesting tricks and strategies for certain tasks, answering novice questions, etc.). This forum needed to be discontinued for understandable logistical reasons, but the unfortunate consequence has been a substantial reduction in communication both within the ISIS user community and between the users and the ASC developers. We note that the ASC has recently encouraged users to start using the GitHub Issues system to report issues with ISIS, but it is not clear if the intended use is limited to bug-reporting, or if it is meant to also support discussions of broader topics.

The ASC should consider implementing a new platform to support user-user and user-developer interactions in the planetary data processing community and to advertise its use. Six months would be a reasonable timeline to develop such a platform. If such a platform cannot be hosted

within the ASC compute environment for logistical reasons, ASC should implement it elsewhere (Google Groups e-mail list, GitHub Issues, many other possibilities exist with minimal overhead)."

ASC Response:

The ASC agrees and has already taken significant steps to address this finding. The ASC has made a variety of platforms available to encourage communication within the many groups that depend on our software and believes that the use and support of these communication platforms fully satisfies this finding. These communication avenues include:

1. AstroDiscuss to be used for widespread user engagement with the ASC and the Open Source community, and to support the discussion of a broad range of topics. AstroDiscuss is an externally accessible URL with a StackOverflow style discussion forum that has effective search capabilities. This discussion forum largely addresses the need for a widespread communication mechanism described by this finding. See finding F10 for additional discussion.
2. GitHub Issues to report bugs or feature requests regarding the ASC software portfolio (including, but not limited to, ISIS). GitHub Issues can be submitted on the relevant individual project repository. See finding F10 for additional discussion.
3. GitHub Projects for the software development teams to interact with project chiefs regarding high-level tasks and project priorities.
4. Gitter for highly technical, quick turnaround discussions surrounding software development projects.
5. We also distribute a Request for Comment (RFC) regarding any significant changes to the code base for comment by the public. These RFCs are currently advertised through the Planetary Exploration Newsletter (PEN) and the Open Planetary blog.

F6 Task-based Workflow Tutorials

"We find that improved documentation for all programs would increase the utility of the software. Specifically, we find that more concrete, practical, modern workflow examples would be extremely useful for users.

While most ISIS applications have adequate documentation, the real power of using ISIS comes from chaining programs together into a workflow. While there are some tutorials that provide examples of these workflows, there are not enough to cover the broad applicability of ISIS. Additionally, more documentation about the general 'theory' of how and when to use certain options for a particular workflow or program would increase the utility of documentation over a simple, terse listing of program options."

ASC Response:

The ASC absolutely agrees with this finding. The ASC has developed tutorials and workshop materials that discuss how to use ISIS applications together in a workflow. These materials are currently available on the ISIS Wiki under *Training* and we are in the process of importing previous tips, discussions, and user questions to AstroDiscuss. In addition, as we are developing our code bases, we are continuing to focus on both the quality of new documentation and making improvements to existing documentation. Given the scope of work performed by ASC, this finding is also tractably addressed by engaging the existing community of users to collectively improve the documentation that we all depend upon. Therefore, we will also continue to reduce the burden of change submissions and actively engage all users to submit improvements to our documentation.

F7 ISIS on the Windows Subsystem for Linux

"We find that many users want a version of ISIS that is functional in a Windows environment, and that new technology available in Windows 10 (the Windows Subsystem for Linux) would allow it. Documentation for how to install ISIS on the Windows Subsystem for Linux is now referenced by the ASC.

It is important to note that there have been many requests over the years to provide a version of ISIS that runs on Windows. The historical problem with this request is that it would have required a substantial development effort to create a Windows-native version of ISIS that would have run on the Windows Command Prompt. Beyond that, creating a Windows program that could be doubleclicked upon and operated from a comprehensive graphical interface–which is what many users mean when they request a Windows version–would require even more effort. Such an effort would have diverted resources from developing new planetary-science-relevant functionality, addressing bugs, and working on critical functionalities within the core codebase. We find that the historical decision not to develop a Windows version was correct, given the limitation of resources.

The new Windows subsystem for Linux allows ISIS (and its future modular descendants) to run on a Windows machine in an analogous manner to the way that it currently does on Apple Macintosh machines; not because a special new development effort was made to run ISIS on Macs, but because macOS provided a Unix-like enough environment for ISIS to run on. The ASC is to be commended for publicly linking to a blog post which documents the process for getting ISIS installed and working in the Windows Subsystem for Linux within the ISIS documentation.

The ASC should incorporate the instructions from the external blog post to the ISIS documentation, though maintaining appropriate caveats about this installation method being not strongly supported is reasonable. As practicable, we encourage the ASC to test ISIS against and maintain at least installation support for ISIS on the Windows Subsystem for Linux going forward."

ASC Response:
The ASC agrees with this finding, and ISIS can now be installed and used within the Windows 10 environment. We thank the SAT for the commendation and for referring to the external blog post written by ASC employee T. Hare which documents the process for getting ISIS installed and working in the Windows Subsystem for Linux. We will also incorporate these instructions from an external blog post by T. Hare into the ISIS documentation and maintain appropriate caveats about this installation method being not strongly supported.

2.3 Core Functionality

F8 Code Modularity
"We find that the ASC has not divided the ISIS functionality into a core, stand-alone library and a set of applications and utilities, as specified in the ETRP Findings. We find that developing such a strategy is of high priority and should be vigorously pursued.

We agree with the ETRP finding that this definition of a clear and concise API will encourage and aid outside developers using and contributing to the software codebase. Discussions with ASC indicate that they have developed a preliminary strategy for accomplishing this modularization effort, and the intent is to identify separable, foundational functionality within the codebase (camera models, SPICE interactions, etc.), write stand-alone libraries with a clean API to provide that functionality, and then replace the current functionality with calls to those APIs, which would address this ETRP finding.

The ASC is encouraged to develop a detailed, 5-year plan for this modularization including a specified order for which functionalities will be modularized and implementation milestones for that plan. One year is a reasonable timeframe to develop such a plan."

ASC Response:
The ASC agrees with both this finding and that of the ETRP and will continue modularizing ISIS into a stand-alone library and a set of applications and utilities. We also agree that a 5-year plan describing a preliminary strategy for accomplishing this modularization would facilitate these efforts. We believe an effective plan will describe a series of milestones for accomplishing this effort, and details to achieve each milestone would be determined when that effort begins. We

believe it is also important to make these milestones and each detailed plan available for comment by the planetary community via a Request for Comment. The modularization plan in terms of milestones should be updated regularly to reflect lessons learned throughout this process. We will complete a 5-year plan for modularizing ISIS into a series of stand-alone libraries and make that plan publicly available by September 2019.

F9 Code Accessibility

"We find that the ASC has placed their main development trunk for ISIS and most of their non-proprietary software development on the public GitHub site, as the 2014 ETRP finding on Code Accessibility recommended.

This placement allows the outside community to more actively participate and collaborate with ASC developers on the code. This is a major shift from their previous development methodology of ISIS, and they are to be commended for it.

We encourage the ASC to continue to develop all major, non-proprietary software projects in the open in this manner."

ASC Response:

The ASC intends to continue to develop all major non-proprietary software projects through GitHub and within this Open Source community. Please see the response to finding F4 for related comments.

F10 Issue Tracking

"We find that the issue-tracking system that was in place for some time at the ASC was opaque to external developers and users, and that their recent switch to GitHub Issues for ISIS is an improved issue-tracking and resolution system that will greatly benefit that project. However, not all ASC software projects do so.

Since ASC is using GitHub for development, using the GitHub Issues system would likely be the best solution. Doing so would reduce effort for all parties, because known issues can be found by simple search, which saves contributors from duplicating effort. Also, new error cases for known issues could be easily facilitated by adding to existing findable GitHub Issues. Additionally, the already opened GitHub pull requests are usually directly linked to issues being tracked. The sequence of opening GitHub Issues and solving them via GitHub pull requests is a well known and highly accepted procedure for software provenance.

The ASC should consider using the GitHub Issues system for issue-tracking with all ASC software portfolio projects (not just ISIS). Six months is a reasonable timeline to accomplish this."

ASC Response:

The ASC agrees with this finding and believes that the actions described below fully satisfies this finding ahead of the 6-month timeline. All major software projects developed by the ASC are available in GitHub and GitHub Issues are used to track bugs and feature requests. The previous issue tracking system used for ISIS is called Redmine. We have evaluated each Redmine ticket, and all non-proprietary issues have been transferred to GitHub Issues. The ASC has been tracking all issues for new projects using the GitHub tool suite since October 1, 2018, and GitHub Issues will now be used for all bug fix and feature requests. All discussions on Redmine are in the process of being transferred to AstroDiscuss (an externally accessible discussion board for the ASC software portfolio), so that this helpful information can be encapsulated and preserved. See also the response to finding F5 for additional discussion regarding all communication platforms used by the ASC.

F11 Software Development Standards

"We find that the ASC does not have a set of consistent and logical software development standards for their software portfolio.

There are currently some loose standards and a style guide available for ISIS, but given the anticipated evolution of the ISIS codebase and other projects in the ASC software portfolio, a new comprehensive set of industry best-practice standards is needed. These standards are not just style guides, but also the guiding philosophies about how to write software for ASC projects. Once these standards are in place, all newly developed code should adhere to these standards. The existing large codebase is unlikely to conform to these new standards, and a massive 'conversion' project is not needed, but as older pieces of the codebase are modified and updated, they should be brought into conformance with the new standards. These standards must be followed by all external contributors, as well as ASC developers. The establishment of these standards will simplify maintenance, revision, and shared use of code, thereby supporting the move to an open source model.

Furthermore, a natural corollary of this Finding is an evaluation of how much of any particular codebase meets these established standards. The ASC should look into reporting a variety of code quality and coverage statistics for their software portfolio.

Establishing a set of such comprehensive software development standards and making them available to the community would support the move towards an open source model. A year is a reasonable timeframe to accomplish this."

ASC Response:

The ASC agrees with this finding and have made positive steps toward addressing this issue. Developing and publishing a comprehensive set of programming style guides and guiding philosophies about how to write software for ASC projects is necessary to facilitate the growth of an Open Source community surrounding the ASC software portfolio. The ASC has a published style-guide and we have developed and made available on the ASC GitHub repository guidelines for contributing to the ASC software portfolio and a code of conduct to encourage an open and welcoming environment with inclusive and diverse participation by the planetary community. We intend to add to and improve this documentation over time, with the help of our Open Source community. We believe this currently available documentation fully satisfies this portion of the above finding.

We acknowledge that some of our existing code does not conform to our current style guides and philosophies. The ASC is committed to ensuring that code that is currently under development, and is developed in the future, does conform to our style guides and philosophies. We recognize that we cannot meaningfully hold the community to a standard if we do not hold ourselves, and are committed to, that same standard. We feel the finding that the ASC should evaluate the existing software portfolio and report a variety of code quality and coverage statistics relating to it is not well justified. This evaluation is a significant effort that would instead divert limited resources from developing new planetary-science-relevant functionality, addressing bugs, and working on critical functionalities within the core codebase.

F12 Universal Binary

"We find that the ASC did not work to create a more universal binary under their old distribution system, but with the November 2018 move to their new conda-based distribution system, this goal is now realized.

The 2014 ETRP included a finding about creating a more universal binary distribution of ISIS. The concept was that such a 'universal' binary distribution would have fewer dependencies on the host system, at the cost of being larger to download, but with the benefit of being able to run on a wider variety of systems.

The new conda-based distribution system addresses the essential desire of the 2014 ETRP Finding to allow ISIS to be installed on a wider variety of systems. This is because much of the

system-dependent library installation issues are handled by the conda system which allows installation on a much broader spectrum of systems than allowed under the previous binary distributions."

ASC Response:
With the release of ISIS 3.6.0 and the use of a conda-based distribution system, we feel this finding to provide a more universal binary has been fully addressed. We will continue to make improvements to the ISIS build and distribution systems to ensure that installation dependencies on the host system are minimized.

F13 Binary Installation

"We find that the rsync installation process that had been in use for many years functioned well and was dependable for highly technical users, but that it had many pitfalls for less technical users. However, in November 2018, with ISIS 3.6.0, the ASC has removed it in favor of a conda-only installation process.

Under the rsync mechanism, warnings in the ISIS documentation about losing data when the installation would be performed incorrectly were frightening to potential new users. Incorrect use of the rsync mechanism caused accidental deletions or unintended data duplication for users. A Java-based graphical installer was supported in the past, but it is no longer functional and is not maintained. While that effort had the best of intentions, there are a number of practical reasons why it is not functional, and we are not suggesting that it be revived.

As of the ISIS 3.6.0 release, the ASC has discontinued the rsync mechanism for software download, and is using the conda package management system, making it a required installation in order to install ISIS. The new conda system does help users ensure that they get the right files in the right places, and that is ultimately a good thing. However, as it is a new installation system, more work needs to be done to streamline the user-experience. Such a mechanism could also enable installations of beta-versions or mission-specific versions for testing.

It is reasonable for the ASC to expect a larger number of user issues related to the new installation system. They should be especially vigilant for unforeseen issues that might cause difficulty with mission or instrument team installations and be ready to support them. If absolutely necessary, the ASC could fall back on the rsync mechanism."

ASC Response:
The ASC agrees with this finding and will continue to support the community and their adjustment to the new installation system. We believe the transition to a conda-only installation

process benefits not only our highly technical users, but allows ISIS to be more easily installed by all users. This change is in direct response to community feedback regarding the difficulty in building and/or installing ISIS. With every release we distribute using the new release system, we are improving our ability to seamlessly deploy our software. When issues have been brought to our attention, we have rapidly resolved the issues that have arisen from the new installation system and will continue to do so. For example, when we were made aware of an issue with 3.6.0 we replaced that version with 3.6.1 to resolve the problem within a few days. In addition, we have received positive feedback regarding this change. We have received no feedback requesting the previous system nor see a clear need to revert to the previous rsync mechanism. Any community feedback regarding the conda installation system can be provided to the ASC by submitting an Issue in the ISIS GitHub repository.

The ASC continues to provide mission-specific versions of ISIS, as needed. However, as our software release Roadmap is realized, we are finding that these special releases are becoming unnecessary for most mission teams. Rather than funding the release and maintenance of a custom version of ISIS, many missions are instead choosing to wait for 3-months or less for their requested software improvements to be included in the standard ISIS release. In cases where a mission-specific release is necessary, the conda distribution environment supports this action in an efficient and streamlined manner.

The ASC no longer maintains a "beta" release. Instead as detailed in our release Roadmap, we provide a "Release Candidate" distribution 1-month prior to the final ISIS release data and this release could be considered a beta release for testing. See the response to finding F14 for additional details about our improved release system.

F14 Development Builds

"We find that the availability of development builds of ISIS (referred to as prerelease, weekly, nightly, edge or dev builds, not just the release-candidate builds that are available) continues to be limited.

The 2014 ETRP found that the existing weekly builds being generated at ASC should be released to external users; however, these builds are still not available. Access to such development builds allows users to verify bug fixes, investigate new functionality, and generally close the loop faster with developers leading to increased software quality and less pipeline downtime. The ASC stated that due to occasional proprietary code in the source tree, weekly builds cannot be automatically released, and that work is ongoing via a new continuous integration platform to make development builds available to external users.

The ASC should consider how to make these development builds (at whatever cadence more frequent than official releases is practical) available in some form (rsync, conda, tarballs on GitHub, etc.). One year is a reasonable timeframe to accomplish this."

ASC Response:

The development builds that the SAT is referring to are no longer generated given the current release process. The ASC believes that our release schedule (see the release Roadmap) and build instructions (see the ISIS build wiki) fully address the needs identified in this finding. Our current release plan includes quarterly releases and up to three bug fix releases (on an as needed basis) between each quarterly release (up to 12 bug fix releases annually). One month prior to each release, we distribute a Release Candidate (RC), which provides an opportunity for users to investigate new functionality and identify any bugs in the RC. Bugs in the RC will either be fixed or the code backed out of the RC. ASC bug fix releases have a one-to-one bug to release correlation, allowing users immediate access to versions of ISIS with a given bug fix without having to wait for the next release. The one-to-one correlation is important because unintended consequences of a bug fix could exist, and we continue to make previous version of ISIS available should a user need to revert. Finally, when external customers (e.g., mission teams) for whom we are developing new code request and fund custom builds, theses custom builds are created, labeled, and pushed to the ASC public facing Anaconda page for any user to download.

Weekly builds are no longer created and nightly builds are no longer generated for public (non-development) distribution. Nightly builds are used for their intended purpose; to allow for overnight testing to run and for developers to assess the impact of the previous day's code changes (made through merged PRs). The ASC release strategy addresses the spirit of the request for higher velocity, public facing ISIS binary availability. We feel this release plan is a significant improvement over our previous schedule and addresses the community needs described by this finding.

F15 Software Compilation

"We find that there have been improvements to the ISIS build system, allowing external developers to compile the ISIS source code. However, it is still problematic for external developers to build and test the ISIS system.

The ASC has already opened up the development process for ISIS to be visible on GitHub and has stated that it wants to create an open-source community around the ASC software tools. For this to happen it is important that the general build process is stable and well documented, so that the community can participate, as indicated in a 2014 ETRP Finding.

There are many difficulties for compiling ISIS by external developers. The precise list of compiler flags and version requirements for different operating systems should be well documented. Additionally, availability of appropriate test data is needed, but seems to be partially addressed by the new ISIS 3.6.0 release which allows test data to be rsynced. For successful integration and participation of an open-source community a test system that can run on a public continuous integration system like TravisCI or similar would be advantageous.

The ASC is encouraged to continue to work to make the primary development branch of ISIS and other ASC software easy to compile by external developers."

ASC Response:
The newly deployed ISIS build system (beginning with version 3.6.0) abstracts operating system differences and dependency management. The use of a standard CMake build system and the cross-platform anaconda dependency management tooling significantly simplifies the compilation of ISIS. We continue to support an array of external ISIS builders using this toolchain and address issues via our GitHub repository. Compiling and maintaining a list of current compiler flags and version requirements for different operating systems is no longer necessary; for the interested party, the CMakeLists.txt files contain all relevant information. We believe the current conda-based distribution system fully satisfies the spirit of this finding, and that we have fully addressed this portion of the finding. See the related finding F12 for additional comment.

The ASC also agrees with the finding that the availability of appropriate test data is needed. We are actively and systematically addressing this concern by adopting a testing framework, working to refactor and improve tests while reducing overall test volumes, and exploring options for version controlled large file storage and transmission support for test data, where appropriate.

F16 Software Optimization

"We find that improving current application performance, where practical, is warranted.

Applications with performance issues (e.g. cam2map) should be investigated and identified. Efforts should focus on applications with the greatest opportunity for performance improvement, prioritizing those that can be easily or quickly fixed. Planning for growth and performance should be a consideration in all future development."

ASC Response:
The ASC agrees with this finding, as we also desire to provide applications that meet high standards of performance. As we work towards modularizing ISIS into a stand-alone library and

a set of applications and utilities (see finding F8), we are also identifying ways in which we can improve performance. In addition, as we address finding F2 by engaging our customers to determine how our core software base should be defined, we can also obtain information regarding the applications where optimization would be most beneficial to the community.

2.4 Methodology

F17 Support for Active Missions

"We find that the ASC's role in NASA mission support has been a cornerstone of the planetary science community, and it should continue and improve.

No other organization is in a position to provide live development and processing support to the wide variety of planetary missions being flown. This has long been one of ASC's most visible roles. The ability to continue to support mission and instrument teams depends on improving the responsiveness that ASC is able to offer those active teams. Early involvement and consistent communication with mission and instrument teams will help to maintain this support. Having a technical contact role between ASC and the team, rather than just purely scientific or administrative contacts, is crucial to this communication. We note that the open source framework toward which ISIS is headed would help to ease this tension, as mission and instrument teams would be able to develop and apply hot-fixes, or even additional features, themselves, with the understanding that those changes could be integrated back into the ISIS codebase. That framework requires clearly defined guidelines for code development, submission, and review to reduce friction for both ASC and mission developers as illustrated in other Findings in this document.

The ASC is encouraged to develop more flexible approaches to dealing with mission and instrument needs by clearly communicating to missions regarding scheduling and other resource constraints that allow both parties to plan and adequately budget for mission support.

NASA should consider ways to encourage and enforce more planning with respect to planetary spatial data infrastructure by mission and instrument teams early in their formulation, which could stimulate the formation of partnerships with the ASC (or other technical providers) earlier in their lifecycle."

ASC Response:

The ASC agrees with this finding and appreciates the commendations regarding our successful support of missions. Supporting missions is one of the primary means by which the ASC supports the planetary community, and we intend to continue and improve this service. We agree

with the finding that "the open source framework toward which ISIS is headed would help to ease this tension, as mission and instrument teams would be able to develop and apply hot-fixes, or even additional features, themselves, with the understanding that those changes could be integrated back into the ISIS codebase." This open source framework and the community we are fostering is intended to allow not only mission teams, but any contributor, to provide bug fixes, new features, etc. into ISIS. In addition, the transparency provided to customers through Git Projects and Gitter has already improved communication and responsiveness between the ASC and our customers, which includes mission teams. This communication framework is a more flexible approach to addressing mission and instrument needs. Our published Roadmap provides another means to clearly communicate to missions regarding scheduling and this release schedule has already been used to help plan and adequately budget for mission support. We believe our current communication platforms fully address this portion of this finding. See also the response to finding F5 for additional detail regarding our communication platforms.

We agree completely that having a technical contact role between the ASC and the mission team, rather than just purely scientific or administrative contacts, is crucial to this communication. To that end and beginning on October 1, 2018, we have strongly encouraged the ASC mission team contacts to designate a technical point of contact for all software development efforts. We also strive to maintain a consistent technical point of contact throughout the life of the mission, and when that is not feasible a clean transition period is maintained.

We also agree that NASA should consider ways to encourage and enforce more planning with respect to planetary spatial data infrastructure by mission and instrument teams early in their formulation. We will engage program managers at NASA HQ to determine means by which technical providers (including, but not limited to, the ASC) could collaborate more effectively with both NASA HQ and individual mission teams. Such collaborations could include planning and implementation with respect to planetary spatial data infrastructure by developing standards, policies, products, and software tools that would help support this effort.

F18 Support for small irregularly shaped bodies

"We find that support for small, irregularly shaped bodies in the ASC software suite is important to the planetary science community, given the increasing number of current and pending small bodies missions.

That work has successfully started with the implementation of support for the Navigation and Ancillary Information Facility 's (NAIF's) Digital Shape Kernel (DSK), as well as additional ray tracing engines (Embree, Bullet), fundamentally enabling active missions. However that support currently has a number of known issues, including efficiency, accuracy, occlusion, limb regions,

overhanging terrain, non-unique lat/lon points, additional projections for irregularly shaped bodies, and alternative shape model formats (e.g. .obj). We support the ASC's stated objective to expand their capabilities for small bodies. ASC is in a unique position, with its core set of expertise, to advance the community in these respects. We agree that coordination with missions, who may have active development of tools and techniques for processing images of small bodies, can help to make that effort more efficient. It is also important to note that there is expertise in the broader community for visualization of image data in 2D and 3D, e.g. geographic information systems (GISs), Java Mission-planning and Analysis for Remote Sensing (JMARS), Small Body Mapping Tool (SBMT), etc. The ASC does not need to reproduce that functionality. Given the variety of potential downstream users of small-body data that might be produced in ISIS, this is an area where export of data to standard 3D formats, as specified by a planetary spatial data infrastructure, is important."

ASC Response:

The ASC agrees with this finding and continues to leverage interactions with active flight missions to make targeted improvements to ISIS to support small bodies. These include, but are not limited to, support for bundle adjustment in cartesian coordinates, support for multi-part DSK files, and improved ray tracing for complex viewing conditions with occlusion. The ASC will continue to develop support for small, irregular bodies as a key component of our software package through a combination of internally funded software development and open source contribution by the community. Additionally, the planetary science community has tools for the visualization of small, irregular bodies in 2D and 3D and ASC will seek to collaborate with and support the efforts of other groups, rather than duplicating these efforts, with the goal of identifying the most cost-effective means to implement needed capabilities.

F19 Automated image matching and feature recognition

"We find that additional focus on improving the automated matching of datasets has clear value as datasets get larger.

ISIS currently has many powerful tools for building control networks manually and via area-based and feature-based image matching. The ASC has made investments in solutions and tools to improve control network creation and management with humans-in-the-loop (e.g. the Integrated Photogrammetric Control Environment, IPCE). We note that the optimal tool or workflow for automated matching in a given situation is often not evident to users at present. In addition, some of the tools that are presently released are difficult to use in practice (e.g. findfeatures). We recognize that the ASC already has some projects related to this finding underway and encourage developments of improved automated matching capabilities in ISIS to continue.

ASC Response:

The ASC agrees with this finding. We believe that matching capabilities that enable the generation of topographic and controlled products is a cornerstone of our contribution to the planetary community. We want to note that generating control networks and controlled products is a complex and data-set-specific effort. The "optimal tool or workflow for automated matching in a given situation" is indeed challenging to identify, different for every data set, and sometimes different based on the desired product. There are no simple solutions to the above expressed concerns. We will continue the development of improved automated matching capabilities within the ASC software portfolio and will seek to provide documentation that is helpful to the technical user.

F20 Broadening the SPICE web service

"We find that it would be desirable to update existing ISIS routines that do not fully support the SPICE web service to function in all cases.

Currently, there is a limitation preventing the use of the SPICE web service for a few missions/instruments (e.g. MESSENGER, CASSINI, HiRISE) because instrument calibration routines rely on local SPICE kernels. For multiple reasons (user convenience, improving consistency of behavior across the ISIS code base, centralizing SPICE management, etc.), it would be an improvement to update existing routines to make the SPICE web service functional."

ASC Response:

The ASC agrees that improved SPICE web service capability would benefit a subset of users, and particularly those working on missions that generate local SPICE kernels. However, we disagree with the above statement that the current SPICE web service is not functional. As indicated, calibration routines make direct access to the NAIF SPICE API. In hindsight, this implementation decision is non-optimal and impacts the perceived usability of the SPICE web service for a subset of higher profile missions because local SPICE kernels for one or more critical calibration processing steps are not available. In FY19, we continue to actively address SPICE modularization in the ISIS code base, but do not plan to make alterations to the calibration applications in the near term. This finding highlights an opportunity for community contribution to the ISIS code base, and the ASC is committed to supporting these community efforts.

F21 Python API

"We find that a Python API for ISIS would be of broad use by the community.

A Python API for ISIS was discussed during our on-site meeting at the ASC as one of the foreseen interfaces to the future ISIS core library. We encourage ASC to openly discuss the planning of this Python API soon, as the implementation of it will increase the ability of non-computer scientist application developers to create planetary science processing pipelines based on ISIS. Currently, developers rely on custom solutions, or third-party Python wrappers (e.g. pysis) that create system calls to ISIS applications, and a fully supported Python API would be welcome."

ASC Response:

The ASC agrees with this finding in principal, but we continue to grapple with whom such an API should support. For example, Python access to the 'low-level' C++ API is achievable at a moderate level of effort, but the burden for composing low-level routines into value adding capability is pushed to the end-user. The API would provide no direct capability to, for example, project an image to a map product, but would provide the pieces necessary to achieve this goal. The burden of understanding potentially complex processes (e.g., look vector to body intersection, potentially accounting for topography, followed by projection in to another coordinate system) is intractable to some subset of our user base. Alternatively, access at the higher, 'application' level is possible, but not before refactoring the architecture used to instantiate and execute applications, and after migration of logic from the applications into a base library has been performed. This work is also a major component of the test modernization work. In short, we agree that an API to the ISIS library written in a dynamic language would be invaluable, but caution that the immediate exposure of an API (the low-level option) would likely not satisfy the majority of ISIS users. We will engage our user community (see also response to finding F2) to determine if access to a Python API for ISIS is a high-priority need relative to other existing priorities and how such an API would be most often used by the community.

We note that ASC has demonstrated the technical competence necessary to provide wrappers to lower-level code bases via the ongoing CSM project. This technology demonstration, in conjunction with the ongoing testing modernization work, has provided ASC with the understanding necessary to develop APIs at both the lower- and application-levels.

2.5 Personnel

F22 Software Management Leadership

"We find that the ASC has not identified a lead architect for the ISIS software base as specified in the 2014 ETRP.

We agree with the ETRP Finding that such a position is critical for this software, and the broader software suite that ASC envisions. We believe that this role could likewise be fulfilled by a steering committee with decision-making authority that meets regularly to guide development. A key component of this leadership is to ensure responsiveness and traceability to ASC and community scientific use cases, funded projects, and mission support.

The ASC should consider establishing a lead architect or steering committee for all cartography and image processing software development with overall technical responsibility for strategic direction of cartography and spatial data infrastructure software development. One year is a reasonable timeframe to accomplish this."

ASC Response:

The ASC agrees with this finding. We have taken steps that fully address this finding and well in advance of the recommended time frame. In October 2018, the ASC put in place a new management structure that includes both a Software Development Lead and a Technical Operations Lead. These two individuals work together to develop and implement a strategic vision for software development in the ASC that addresses high-priority community needs. This new management structure largely addressed this finding.

F23 Workload

"We find that the ASC has followed the 2014 ETRP Finding on Workload by both hiring new developers and successfully outsourcing work to external contractors.

This mixed model of adding local talent and going to external groups has been a successful approach and should continue. It allows for expertise to be retained at the ASC. It also allows the ASC to be flexible if they either lack a particular expertise, or available on-site developer hours."

ASC Response:

The ASC agrees with this finding. We will continue to use our current model of hiring developers, both as contractors and as federal employees, to meet the software development needs of the ASC. We also encourage and will fully support contributions by our Open Source

contributors to help meet the software development and documentation needs of the planetary science community.

2.6 Oversight

F24 Continued Funding for Support and Maintenance

"We find that continued funding for software support and maintenance are critical parts of ASC's role as stewards of ISIS and their NASA-funded software portfolio.

The initial lack of funding for software support in the FY2019 Inter-Agency Agreement (IAA) budget curtailed ASC's ability to acknowledge and address bugs submitted by the community. The FY2019 budget under the IAA agreement allowed only for maintenance activities, which we understand to be mission-supported updates. This left the community at large, outside of funded missions, without support for a major piece of software. We understand that this oversight has been corrected, but it was a troubling omission.

We encourage NASA and the ASC to view the ASC software portfolio as fundamental infrastructure for planetary science that requires ongoing support and maintenance."

ASC Response:

The ASC agrees with this finding and has intentions to significantly fund software support and maintenance in FY20 in the following ways. We will have dedicated funding through our NASA-funded software portfolio to support bug fixes and feature enhancements requested by the community through GitHub Issues. We will prioritize these requests and then communicate to the broader community through GitHub which requests are going to be addressed by the ASC. Remaining Issues are then a prime means for other Open Source community members to contribute by making code changes and submitting these changes to the code base via a pull request. We strongly encourage and will support these contributions. In addition, we will also dedicate effort toward specific, larger efforts related to support and maintenance (e.g., improving our release system and making test data more easily available) that make code submission by external contributors less burdensome. If the planned funding models are successful, we will continue this level of support, with appropriate adjustments on an annual basis, in future years as well.

F25 External review of ASC annual funding requests

"We find that it would be valuable for NASA management of ASC software and development funding under the annual IAA to include some component of external review.

The review that we imagine would not require an extensive review process (i.e. an in-person panel). Instead, a process whereby one or two external reviewers examine the priorities and levels of effort proposed by the ASC would have potential to improve outcomes.

NASA should consider soliciting this type of external feedback on the same cadence as new work effort is proposed under the IAA."

ASC Response:

The ASC is inviting external feedback by the community at many different levels in the process and we agree completely that community input is extremely valuable. We have put in place mechanisms by which the strategic direction of the ASC will be established, for how this direction and our priorities will be made transparent to the community, and several avenues by which the community can provide critical feedback. The ASC is actively seeking to engage users and be more transparent in our communication regarding software direction through the following means:

1. Standing up an internal software management group who makes strategic decisions for what software is developed within the ASC and makes those strategic decisions publicly available in a long-term planning document (see also response to findings F1 and F22). We seek to stand up this group by September 2019.
2. Standing up a Technical Steering Committee (TSC) that would discuss and help inform implementation decisions within the ASC Open Source software portfolio. All TSC meeting notes and decisions are posted on GitHub, and membership on the TSC is continually solicited. In addition, all contributors to this Open Source community would be held to the decisions of the TSC, not only the ASC. This action has already been initiated.
3. Publishing on a public website the ASC's long-term plan for software development, product generation, and data access improvements. This long-term plan would be updated at least annually, so that it remains a living document that is in-line with the current state of software development. See response to finding F1 for additional discussion.
4. Initiating a Request for Comment (RFC) to describe proposed changes to the ASC code base to encourage broad community comment and engagement. This action has already been initiated.

Ultimately, the ASC is held accountable by NASA HQ and receives oversight by NASA program managers at regular intervals throughout the year. We will engage NASA HQ in a discussion about whether the mechanisms described above are adequate and determine if

additional oversight or review is needed. We feel these steps will meet the ultimate goal of oversight and external feedback, and with contribution from a broad user base and in a transparent and inclusive manner.

Table 1

Timeline for addressing findings. This table includes all actions where a timeline for completion was indicated in the SAT finding.

SAT Finding	SAT Timeline	ASC Action	ASC Timeline
F5 Community and User Support Forum	Summer 2019 (6 months post review)	The following communication platforms are available for engagement with users: 1) AstroDiscuss; 2) Git Issues; 3) Git Project; 4) Gitter; 5) Request for Comment	February 2019
F10 Issue Tracking	Summer 2019 (6 months post review)	All non-proprietary Redmine tickets have been transferred to the GitIssues. New Issues can be posted to request bug fixes or new features.	February 2019
F1 Strategic Plan	Winter 2019 (1-year post review)	*In Progress*	*TBD (Planned for September 2019)*
F8 Code Modularity plan	Winter 2019 (1-year post review)	*In Progress*	*TBD (Planned for September 2019)*
F11 Software Development Standards	Winter 2019 (1-year post review)	The ASC has a published style-guide, Guidelines for contributing to the ASC software portfolio, and a Code of Conduct.	November 2018
F14 Development Builds	Winter 2019 (1-year post review)	Our improved release schedule (i.e, Roadmap) addresses this finding.	November 2018

F22 Software Management Leadership	Winter 2019 (1-year post review)	ASC put in place a new management structure that includes both a Software Development Lead and a Technical Operations Lead.	October 2018