A Software Archiving Node for the Planetary Data System

PI: Adam Brazier[1]
Co-I: Alexander Hayes[1]
Co-I: Chase Million[1]
Co-I: Mahadev Satyanarayanan[2]

[1]Cornell University
412 Space Science Bldg.
Ithaca, NY 14853-6801

[2]Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213-3815

Table of Contents

Executive Summary

We propose to create a Software Node of the Planetary Data System (PDS) to archive and curate high-value software of scientific relevance to NASA's planetary missions. High-valued software includes that used to generate and analyze data archived elsewhere in the PDS, mission and instrument calibration and processing pipelines, and many analysis or processing tools written by other research groups or community members. In modern scientific workflows, the software is often nigh indistinguishable from the methodology itself, and therefore necessary for future researchers to understand, validate, reproduce, or build upon the earlier work. In this sense, such software *is* research data as stated in OMB Circular A110, and echoed in NASA Plan: Increasing Access to the Results of Scientific Research, where "*Research data* is [sic] defined as the recorded factual material commonly accepted in the scientific community as necessary to validate research findings [...]" [24][25].

Simply sharing, publishing or releasing source code or a software description (e.g. via Github, a personal or institutional website, or a Software Interface Specification) is insufficient. A Software Interface Specification (SIS) may be incomplete in unanticipated ways (e.g., not recording uncaught software errors), and personal and institutional memories degrade. Even assuming that the source code remains available outside of a formal archive like the PDS, all software becomes un-runnable over time through changes to the software and hardware operating environment in a process called "bit rot." This eventually results in complete obsolescence of the software—sometimes within just a few months—and any functionality is lost. To maintain the usability of software for decades or more requires a focused and intentional effort, as we propose, to create a record of the software, its operating environment, and the information necessary to recreate running instances of the software. The practice of software archiving—as distinct from archiving of other digital data objects or the release of source code—is *new*. Only a few researchers are exploring this area, and the PDS Software Node will be the first archive of its type outside of narrow contexts, test cases, or technology demonstrations. All of the required technology exists, however, and the PDS is particularly well suited to this effort.

Core products will be be reviewed by node, network, or institutional staff for PDS4 standards, ITAR, and licensing compliance and adherence to the specific requirements of the Software Node and also peer reviewed by qualified members of the community for completeness and merit. We will provide support to data preparers at all stages of work in service to the goal of a smooth and compliant archiving of data with maximum value to the community and public. Archived data products will be available and searchable through a public-facing web portal, and we will provide a range of services for the community to access and use archived software. Subject to licensing restrictions of the software's operating environment and dependencies, we will develop and provide scripts for the creation of running instances of the software either natively or in virtual machines (VM) on users' own systems [42], and host VM images of software running in its native environment for end users to run in a VM manager on their own systems or via "one-click," in-browser access via Olive Executable Archive technology which provides access to archived executable content running within VMs on remote servers.

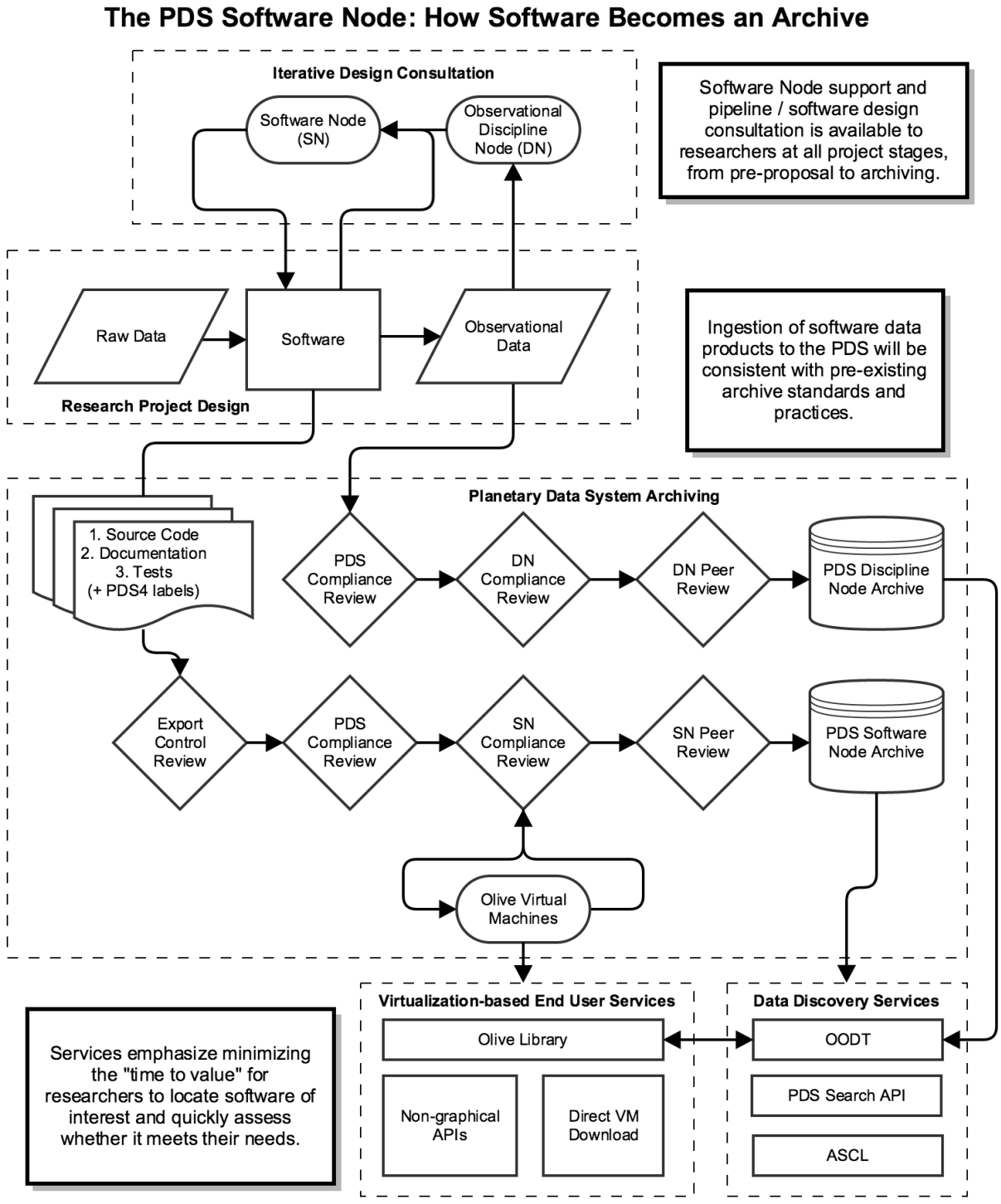


Figure A. How software becomes an archive. This flowchart describes the relationships and interdependencies of the Software Node ecosystem, from "raw data" to archive to virtualization-based end user services like Olive.

For non-graphical software (e.g. many pipelines), we will provide remote API access to archived software running on our own servers. We will host and moderate a "Software Node Forum," to provide assistance on use of archived software, general advice and feedback on preparing software for archiving, and a space for community members to discuss specific pieces of archived software. We will host at least once-yearly training workshops coincident with major planetary conferences on the topics of preparing or using Software Node data holdings, and at least one Node staff member will be available at each of three major planetary science conferences (AGU, LPSC, and DPS) for one-on-one consultations on the same. We will also work with and advise research, mission, and instrument teams at all project phases to develop software that is compliant with Node and Federation requirements, has maximum quality and utility, and can be smoothly ingested into the archive. We will also identify, locate, restore, and archive *legacy* software.

The PI will actively participate in the PDS Management Council (MC), attend monthly teleconferences, and travel with at least one Co-Investigator to each face-to-face meeting. We will work with other nodes and the network to improve the use and utility of all PDS holdings, including support for continued development of the IPDA Tools Registry, integration with the PDS metadata search API, and an effort to establish or strengthen a link between *observational data* held in other Discipline Nodes and *executable data* (software) held in the Software Node.

The proposed work is innovative and on the cutting edge of archiving practice. In addition to expertise in archiving, data management, and Planetary Science, it will require skill in software recovery, engineering, project management, and maintenance. Our team composition reflects this and contains all of the skills needed to meet the Node objectives. The Node Manager (PI Brazier) and Deputy Node Manager (Co-I Million) will perform the bulk of the technical, scientific, and managerial functions. A Node Scientist (Co-I Hayes) will contribute scientific expertise related to the archived software.

The Software Node will contain two consortium institutions. Cornell University will be the lead institution, and responsible for all scientific, technical, and archival domain needs. An Olive Subnode team, led locally by the Olive Subnode Manager (Co-I Satyanarayanan) at Carnegie Mellon but in close consultation with the Software Node Manager, will develop and provide technologies to maximize use and reuse of archived software products by the community and public by building upon expertise and capabilities already acquired through the Olive Executable Archive project and provide advice on issues related to the archiving of executable digital data. If awarded, all funds will flow from NASA directly to Cornell University pursuant to a Cooperative Agreement between NASA and Cornell University. The Subaward Division of the Cornell University Office of Sponsored Programs will provide a subcontract to Carnegie Mellon University consistent with Carnegie Mellon's proposed budget provided herein. Principal Investigator, Dr. Adam Brazier, will be responsible for ensuring that the services provided by Carnegie Mellon University are consistent with those promised in this proposal. Funds will be transferred to Carnegie Mellon University on a year-to-year basis. Dr. Brazier will be responsible for the Science and Technical reporting required per the terms of the Cooperative Agreement.

Technical, Implementation, and Management Plans

Introduction

The Planetary Data System contains a wealth of observational data related to human exploration of the solar system, but it does not currently provide capabilities for archiving the *software* used to create or analyze those data. In modern scientific workflows, the software is often nigh indistinguishable from the methodology and therefore necessary for future researchers to understand, validate, reproduce, or build upon the earlier work. Simply publishing source code without a simultaneous effort to describe or preserve its environment and functionality makes it effectively useless in the long term as personal, institutional, and digital memories fail. The end result is that software-based scientific capabilities are either *simply lost* or can be recovered or reproduced only at considerable effort and cost. A true *archive* is necessary to prevent this.

We propose to create a Software Node of the Planetary Data System (PDS) to archive and curate high-valued software of specific scientific relevance to NASA's planetary missions. High-valued software includes that used to generate and analyze data archived elsewhere in the PDS, mission and instrument calibration and processing pipelines, and many analysis or reduction tools written by other research groups or community members. This Software Node fills a gap in the current capabilities and services of the PDS and serves the interest of NASA in maximizing the usefulness of mission data. It is neither an active project management tool like Github [1] nor a directory like the IPDA Tools Registry [2], but a focused and intentional effort to create a record of the software, its operating environment, and the information necessary to recreate running instances of the same.

A. Identification of Data Sets

The core units of data (the "product" *in toto*) to be archived by the proposed node are: (1) source code, (2) documentation for installation, operation, and maintenance of the software (including detailed descriptions of its dependencies), and (3) regression tests. These units will be stored as plain text and are therefore compliant with PDS4 standards (as "parsable byte streams" per 12.1-2 of the "PDS4 Concepts" document [3]). Regression testing is a software engineering practice that confirms that software inputs continue to produce expected outputs; tests require input and output data, configurations, and running software. If "non-core" data is required by software or regression tests (e.g. calibration reference files), then those will also be archived as supplementary data, unless already archived in another Discipline Node.

PDS Discipline Nodes already hold some source code as supplements to specific data. PDS3 permitted archiving of software, and Section 3.5.4 of the PDS4 Proposer's Archive Guide [4] states that source code can be included as documentation (although we're not aware of any software archived under PDS4 as of this writing [5]). All such codes will be candidates for migration to the Software Node, subject to a Node and peer assessment of the value of such products for archiving as primary data products in their own right. Because such holdings are *not yet* primary data products in the PDS, there is presently no simple way to systematically list all of them without manually inspecting every collection. We provide a representative example below

in the form of calibration software for the Cassini Magnetometer, and will work with other Discipline Nodes to identify other such items within the corpus.

Generally, we do not consider PDS or other data access and use tools (e.g. image viewers, web sites, database management systems) to meet our definition of "high-valued" in the context of archiving within the PDS Software Node—especially where such software more appropriately falls under the purview of the Engineering Node—and *do not* propose to archive those. The difference is that while such tools—tools for viewing, but not working with, archived observational data or to validate data integrity, as, e.g., with the MarsView and MD5 checksum utilities archived with the THEMIS data [6,7]—are certainly *useful*, no fundamental reproducibility, functionality, or knowledge is lost if they become *unusable*.

Software that encodes the workflow of calibration, reduction, processing, and analysis of planetary science mission data sets are likely to contain information necessary for reproducibility that does not exist elsewhere. A Software Interface Specification (SIS) may be incomplete in unanticipated ways (like not recording uncaught software errors), and personal and institutional memories degrade; with the "passing" of the software, capabilities may be truly *lost*.

There are almost certainly unarchived codes of high value associated with every NASA planetary mission and instrument of the past ~30 years or more. Active or upcoming missions (such as those outlined in Appendices B and C of the CAN) certainly are producing or will produce codes of high value that can and should be archived. Currently, they may not have plans or budget to do so, and it is important that archiving software does not constitute a *new* burden on those missions or projects. In recognition of this point, we draw the following distinction between *legacy* and *future* software products:

Future Software: We define "future" software as all software conceived, budgeted, and designed with the goal or intention of eventually archiving it in the PDS. In practice, this will almost exclusively comprise software from projects that will be proposed after the Software Node is established. Because the investigators for such projects will have the benefit of planning to design their software to meet PDS and Node requirements and working with Node staff to design the software with the intention of a smooth, compliant, and maximally-valuable archiving process, we expect the associated software data products to be *fully compliant* with all Node requirements. As the software will be relatively "fresh" at the time of archiving, the PDS-side cost and effort associated with processing (including Node and peer review) and ingestion of such products will be consistent, predictable and relatively small (on order of 10 to 40 hours of effort per product, with efficiencies of scale and experience). The level of effort required for Node support of researchers at *all* stages of work—from pre-proposal through to actual submission—will be greater and will vary as a function of the size, complexity, and duration of the project.

Legacy Software: For ongoing and completed missions or projects, project funds and possibly the researchers themselves may not be available to support the archiving process, and the software products may not have been designed with any intention of release or reuse. We will seek out, recover, restore and archive these "legacy" products, and will work with the appropriate teams to do so to the extent possible given limitations in funding and time. For such legacy software, case-by-case exceptions to the Node's baseline archiving standards (described in Section C) will be considered, with the goal of archiving the products in as complete a manner as possible under any constraints and commensurate with their scientific importance. Exceptions to standards for peer review, described in E, *will not be made*. In the absence of additional support, the cost of preparing legacy products for archiving will fall to the PDS Software Node (under our mandate to "identify and restore datasets from past planetary missions").

By the nature of planetary mission timelines, we anticipate that the majority of software that the Software Node will acquire and archive in the five-year period of performance will be *legacy* by our definitions. We will actively identify, locate, acquire, and archive legacy software of high value. We also request the assistance of NASA in the identification, location, acquisition, and licensing for archiving of other planetary data processing and analysis software products, including encouraging and enabling past, current, and future NASA planetary researchers to submit their software to the PDS or provide the Node team access to software-related assets from past projects which reside on NASA systems. What follows is a *representative sample* of the types and classes of past, current, and future *legacy* data that we wish to archive, deriving from both inside and outside the current PDS holdings.

Examples (representative of software data within the scope of the Software Node):

- One class of PDS software of tremendous scientific importance and worth is the current and past versions of the SPICE [8] toolkit developed by the Navigation and Ancillary Information Facility (NAIF) Node of the PDS and used by planetary researchers at all project stages, from mission planning to data analysis. Note that we *do not propose* to reproduce the functions or services of NAIF (or any other software-producing project), in whole or part, but rather create an archive and record of software which lies at the heart of a tremendous fraction of modern planetary research.
- Cassini Magnetometer [9] Raw Data Archive held by the Planetary Plasma Interactions (PPI) Node has a "*software*" subdirectory which contains, as described by the text file therein, "software required for data processing and examination (provided as source code, executables and algorithms), and examples of raw data and the corresponding processed data, which demonstrate correct application of the processing software." [10] This software likely meets our definition of "high-valued" and should be archived in the Software Node. The associated sub-folders contain reference data, C++ (.cpp) files, compiled Windows executables (.exe), and PDS labels (.lbl). It appears that the source files were last edited circa 2005; whether or not the .exe runs on a present-day computer, it eventually *will not*. We will need to create and document a working operating

environment for the .exe or, more ideally (and perhaps easier), recompile the source code. The documentation is good. The format would need to be updated to PDS4, but the included examples would likely prove sufficient to construct "regression tests" to meet baseline Node requirements.

- "The Integrated System for Imagers and Spectrometers (ISIS) is a free, specialized, digital image processing software package developed by the USGS for NASA. ISIS key feature is the ability to place many types of data in the correct cartographic location, enabling disparate data to be co-analyzed. ISIS also includes standard image processing applications such as contrast, stretch, image algebra, filters, and statistical analysis. ISIS can process two-dimensional images as well as three-dimensional cubes derived from imaging spectrometers. The production of USGS topographic maps of extraterrestrial landing sites relies on ISIS software. ISIS is able to process data from NASA and International spacecraft missions including Lunar Orbiter, Apollo, Voyager, Mariner 10, Viking, Galileo, Magellan, Clementine, Mars Global Surveyor, Cassini, Mars Odyssey, Mars Reconnaissance Orbiter, MESSENGER, Lunar Reconnaissance Orbiter, Chandrayaan, Dawn, Kaguya, and New Horizons." [11] A significant amount of planetary science research relies upon this software package, and has for the 20+ years since its introduction. The current and actively maintained version is ISIS3, resulting from a rewrite if the previous version into C++ circa 2004. Not only might the previous version, ISIS2, be required to validate research prior to ~2004, many legacy workflows continue to rely on it, and that version is now sufficiently old that it does not run natively on modern hardware. There is also complex functionality that exists in ISIS2 but has not been migrated to ISIS3; for example, only ISIS2 has a set of programs, used by active mission teams, to deal with "single pixel" data like that produced by Mars Orbiter Laser Altimeter or the real aperture mode of the Cassini RADAR. Randy Kirk (USGS) also developed single-image photoclinometry capabilities for ISIS2 that, as far as we know, are unique within the planetary community. [12] The current project manager of ISIS development, Trent Hare, is listed as a Collaborator on this proposal and, with his guidance, ISIS2 and ISIS3 will be the first two pieces of software formally archived in the Software Node.
- The Microwave Radiometer (MWR) [13] on the Juno mission to Jupiter [14] which will arrive in July of 2016 and will collect microwave observations of Jupiter's deep atmosphere and radiation belt starting in November. These data are expected to serve as a valuable legacy data set, and a tremendous work effort is being put into development of calibration and visualization tools *necessary* for use of that data. Without long term preservation of the software, the capability to use the data (or, at minimum, reproduce the analyses of the instrument team) will simply be lost, and the instrument team has no budget for software archiving. The PI of MWR, Mike Janssen, is listed as a Collaborator on this proposal with a commitment to provide this software to the Node team for potential archiving.

- The ChemCam (Mars Science Laboratory) [15,16] and SuperCam (Mars 2020) [17] instruments combine laser induced breakdown spectroscopy (LIBS) and Remote Micro-Imaging (RMI) to acquire elemental composition information and high resolution imagery of targets on Mars. The data produced by these instruments require substantial post-processing to derive results, to which task the instrument teams have (or will have) developed new techniques encoded in software tools. For future researchers to understand, reproduce, or build upon the ChemCam and SuperCam results with confidence will require access to those tools. The PI of these instruments, Roger Wiens, is listed as a Collaborator on this proposal with a commitment to provide these ground data processing tools for possible archiving.
- The Arecibo Observatory has been and continues to be actively used for the study of solid bodies of solar system objects. The result has been valuable radar images of Venus, Mercury, Jovian and Saturnian satellites, and many asteroids and comets [18]. As with all radar data, the production of scientific images requires substantial digital post-processing and calibration. The PI of Arecibo Planetary Radar, Mike Nolan, is listed as a Collaborator on this proposal with a commitment to provide for possible archiving the processing and calibration scripts and programs used by this group.

Benefit of software reuse: Despite being on the same spacecraft, Compact Reconnaissance Imaging SpectroMeter (CRISM) [19] imaging data are usually map-projected using the 128-pixel-per-degree Mars Orbiter Laser Altimeter (MOLA) [20] shape model of Mars, while HiRISE [21] image projections typically use a Mars sphere model (which does not account for variations in local shape). A researcher may therefore need to exert a great deal of additional effort and accept large amounts of distortion in order to get images from these two data sets to align in the region of interest. This kind of problem is not unique to the Mars Reconnaissance Orbiter data sets. Access to the running pipelines used to generate the data in the first place—which might then, in principle, be modified by researchers to use the same model—would dramatically simplify such activities.

Capabilities may be lost: Mastrogiuseppe et al. 2014 [22] used the Cassini RADAR [23] as a sounder to probe the depth and composition of Titan's seas. The Cassini RADAR was not intended to operate as a sounder and the analysis required the application of custom tapering functions using the CPADS processing software currently only available within a subset of the Cassini RADAR Science Team. If Mastrogiuseppe were not a Cassini Co-I with access to the CPADS software, this important result would not have been possible.

B. Rationale and Justification

In modern scientific workflows, the software and the methodology are highly interdependent, and the software is therefore necessary for future researchers to understand, validate, reproduce, or build upon the earlier work. In this sense, such software *is* research data

as stated in OMB Circular A110, and echoed in *NASA Plan: Increasing Access to the Results of Scientific Research*, where "*Research data* is [sic] defined as the recorded factual material commonly accepted in the scientific community as necessary to validate research findings [...]" [24,25]. A Software Node fills a gap in the current capabilities and services of the PDS, which archives software only as "documentation", and serves the interest of NASA in maximizing the usefulness of mission data.

To maintain the usability of software for decades or more requires a focused and intentional effort, as we propose, to create a record of the software, its operating environment, and the information necessary to recreate running instances of the same. Simply sharing, publishing or releasing source code or a software description (e.g. via Github, a personal or institutional website, or a Software Interface Specification) is insufficient to ensure long term survival of the software. Even assuming that the source code remains available outside of a formal archive like the PDS, all software becomes un-runnable over time through changes to the software and hardware operating environment in a process called "bit rot." This eventually results in complete obsolescence of the software—sometimes within just a few months—and any functionality is lost.

Scientific and mission software is especially susceptible to bit rot because it is often developed as a bespoke solution by small groups of people with relatively little support provided for ongoing maintenance. The field of planetary science is rare among scientific disciplines in that we largely share a single, core set of observational data—much of it archived by the PDS—from which almost all results are derived. These data were difficult and expensive to collect and represent unique observations, generated by unique instruments and missions. Gaps of years or decades between missions to specific targets mean that both personal and institutional memories related to analysis *techniques* and *procedures* can and do fail, so that information necessary to use, understand, or synthesize older data becomes difficult, if not impossible, to recover and restore. The software we propose to archive records these techniques and procedures and have tremendous scientific and cultural value, consequently making their preservation particularly important; by preserving software, we give future researchers greater capacity to understand or modify earlier methodologies to extract new knowledge from the data.

The practice of software archiving—as distinct from archiving of other digital data objects or mere publication of source code—is relatively new. Only a small number of researchers are actively exploring this area. It's likely that the PDS Software Node will be the first archive of this type outside of test cases, demonstrations, or efforts targeted narrowly at one particular piece of software or multiple pieces of software on a particular piece of hardware (e.g. Atari games) [26,27].

An archive of software in the PDS, however, is both timely and appropriate. All of the required tools and techniques exist *right now*, as detailed in this proposal, and research software is being increasingly recognized as an important scientific work product critical to reproducibility and progress [28,29]. Planetary Science also is a particularly suitable field for this type of effort for the following reasons:

- The Planetary research community largely shares and derives all of their results from the same set of observational data, multiplying the benefit of making tools for understanding, working with, and extracting further value from that data.
- Similarly, there are unrealized efficiencies of code reuse by unrelated researchers working on the same or similar data sets or problems.
- There are a (relatively) small number of clearly identifiable pieces of software—mission and instrument pipelines in particular—with extremely high scientific and cultural value, making the effort to archive them justified.
- The most important "significant properties" of these software—the attributes which need to be preserved most faithfully—are well defined: identical inputs need to produce identical outputs (possibly within some defined margin).
- The PDS has a pre-existing and thorough archive of the very *observational* data that these software operate upon. Within this common archiving ecosystem, we can close the loop between data preservation and use.
- By the nature of the close relationship between the PDS and mission/instrument teams, we can work directly with the developers from early project phases to maximize the archivability of tool designs. We will established procedures to test thoroughly and document the properties of the software *moving forward* in a way that simplifies the creation and long-term maintenance of archival products over *post hoc* efforts at software archiving [30], while also restoring high-value software which predates the creation of the PDS Software Node.

C. PDS Archive Standards and Practice

We will work with the PDS Management Council (MC) and, in particular, the Data Design Working Group (DDWG) to define changes to the PDS4 Information Model as may be required for a maximally useful, consistent, and complete description of software as primary data products across the PDS and IPDA. In particular, we anticipate that an expansion of the current PDS4 definition for *Product_Software* (which is quite sparse as of Version 1.4.0.0 [31]) will be necessary. The Software Node PI will serve on and actively participate in the PDS Management Council. Software Node Staff will also serve on and actively participate in PDS Federation boards and working groups (including the DDWG and the Change Control Board) and in other advisory capacities as necessary and appropriate, within limitations of available time. The Software Node will comply with PDS policy decisions issued by the MC. Prior to peer review, core products will be reviewed by node, Federation, or institutional staff for PDS4 standards, ITAR, and licensing compliance and adherence to the specific requirements of the Software Node.

Node Requirements: Software Node-specific requirements will be (1) that the software source code and attendant documentation for installation, operation, and maintenance must be structured and released into the public domain according to PDS policy and standards, and (2) Node staff

can reproduce the functioning software in a virtual machine (VM) using information in the source code and documentation, and (3) the software as reproduced in the VM must pass all regression tests.

Virtual Machines are computers running in a software emulation of a computing hardware environment, with performance controlled by the allocations they receive on the host computer (particularly CPUs and RAM). They can be moved or copied to other hardware, or multiple copies of the same VM can run side-by-side, allowing scalability and isolated computing to maximize efficient use of available hardware, enhancing security and maintainability. As a VM is an emulation by one computer of another, it can be used to instantiate a precisely defined operating environment that matches the one under which the archived software was actually developed; in other words, not only does installation on a VM provide us with a means to verify that the software documentation, installation instructions, and regression tests are sufficiently detailed and accurate to permit reuse, but a VM also freezes the software's precise environment so that it can be executed even decades after creation. The VM image created by Node staff as part of this compliance check will be retained (but not "archived," because VM images are not current PDS4 compliant) and in most cases become a direct resources to the community either through direct download, access via Olive (as described in detail below), or as the executable heart of a non-graphical API (subject to licensing restrictions of the software dependencies, e.g. the Windows operating system). The PDS Software Node will produce documentation describing the production of the VM and its automation. As already noted, for *legacy* data products, case-by-case exceptions may be possible for Node-specific requirements (but not PDS or peer review requirements).

Completeness of tests: We intentionally do not specify any required *amount* of test coverage either as part of Node requirements or peer assessment (described in E) because such coverage has little meaning divorced from the properties of the code and quality of the tests. Node Staff will assess for the non-subjective, binary determination of whether or not the tests *simply pass*, whereas peer reviewers will assess for the more complex and judgment-based determination of whether those tests are *sufficient*, based on a range of criteria. The Node will also use the tests as an executable data adjunct to a checksum; that is, to test whether the nominally static product has changed, as a scheduled element of the Data Integrity tests. In many cases, functional end-to-end "black box" testing of the software system as a whole will be sufficient for the software to meet review standards.

Export Control: Software source code and attendant documentation related to the reduction and analysis of data from planetary spacecraft missions could potentially be export controlled as defined in the International Traffic in Arms Regulations ("ITAR"), "22 CFR (120-130)" and the Export Administration Regulations ("EAR"), "15 CFR (730-774)." [32,33] Consistent with current PDS practice, data providers will be required to audit any submitted data products for export controlled content and certify that they are free of such content that cannot, by definition,

be placed into the public domain via the PDS archive. Node or institutional staff will *additionally* review all submitted materials for potentially export-controlled content prior to any publication or dissemination of those by the Node or prior to sharing it with any non-US persons. Through its long history in NASA missions, Cornell University and the staff of the Center for Radiophysics and Space Research ("CRSR") have extensive experience understanding and navigating export regulations. CRSR recently organized a half day seminar during which an attorney from a Washington D.C. law firm, whose practice specializes in, among other areas, export control, provided training in connection with the recent changes made to the export regulations resulting from the President's Export Control Reform Initiative. In addition to other staff across the University, the training session was attended by the Associate Director of CRSR, who is a legal professional, and by Co-I Million. Cornell University also has on staff an export control officer who is available for consultation on an as-needed basis. As such, we both understand and are fully prepared to deal with any export control issues that might arise in the course of Node activities. We will also, as necessary, work with and call upon the experience, expertise, and advice of NASA, PDS Project Management, and the PDS Management Council to resolve issues of export compliance.

Licensing Requirements: The CAN states that "[a]s a matter of NASA policy, all data taken or products created in the performance of a NASA research award are considered to be public domain," and release into the public domain is a non-negotiable requirement for products to be archived in the PDS. Data providers will be required to self-report instances where submitted *core* products (being source code, documentation, and regression tests) contain non-original work which cannot be released into the public domain or otherwise properly licensed according to PDS and NASA policy, and those subsets of the data will be omitted from review and archiving. In the case of software, one must not be concerned only with the licensing of the plain text source code, but with the operating environment as well. That is, software that is otherwise in the public domain may depend on a licensed runtime environment like Matlab or an operating system like Windows or a proprietary library like the DivX Pro Codec. The licenses on these dependencies might carry many and varied restrictions against different classes of use including copying, running, sharing, reproduction, etc. Importantly, the Software Node will archive thorough descriptions of operating environments in the PDS but *not the environments themselves*. Node requirements will also not restrict the types or classes of dependencies that data providers might use when developing their software. Although we will encourage developers to use dependencies that are "open source" and licensed for free distribution and execution, where appropriate, we recognize that development of research software is results-driven, and so use of proprietary software may be necessary, efficient, or otherwise preferable to the use of an open-source solution. Although it is possible that extensive use of proprietary dependencies or use of such dependencies for core functionality of the submitted products might reduce the value of the software in the eyes of the review panel, we understand the constraints under which technological decisions are made in the field.

For many common licensed dependencies, Cornell University's institutional software licenses will suffice for Node staff to legally recreate the operating environments as part of the Node requirements check. There will also be cases where the data preparers can furnish the PDS Software Node with a license for this purpose and we may appeal directly to the licensing entities for an "archive exception," especially in cases of very old and obsolete software dependencies (like Windows 3.1). Our end users services (e.g. build scripts, Olive access, or non-graphical APIs) will include the provision of access to non-archived data which *do* include operating environments (e.g. VM images) and those will take into account and respect the restrictions levied by licensing of dependencies; different archived software will therefore inevitably have different types of associated end-user services.

D. Working with New Instruments and Teams

The Software Node will provide support to data preparers at all stages of work in service to the goal of a smooth and compliant archiving of data with maximum value to the community and public. The PDS already provides feedback and review of processing pipelines (per, e.g., section 6.5 of the Proposer's Archive Guide, v2.00 [4]) and advises data preparers in the design of pipelines to minimize the additional effort required for data archiving. For example, a common suggestion is that pipelines should reduce data into immediately archivable formats. The Software Node Staff will either step into or supplement this pre-existing PDS function for current and future mission, instrument, and research projects.

As explained in the Proposer's Archive Guide, "A new mission is usually assigned to a single discipline node, which serves as the 'lead node' for the remainder of the mission. Lead nodes, in turn, arrange for support of individual instrument teams (including potential data contributors without specific instruments, such as Interdisciplinary Scientists) through other discipline nodes if appropriate." and "In many cases, the choice of lead node will be obvious; proposers may have even specified a lead node in the proposal and/or CSR." Because it is not the primary scientific purpose of missions or most research projects to produce software, we do not expect the Software Node to be the obvious or appropriate Lead Node (LN) for mission or instrument projects or for the majority of data-archiving research projects. That is, in cases where *observational data* is being produced, we expect there to be a Discipline Node more closely aligned with the scientific objectives of the project. The Software Node might be the appropriate and obvious LN, however, for research projects where a software tool will be the *primary work product* (e.g., some responses to the Planetary Data Archiving, Restoration, and Tools program). In such cases, we will serve in that role and follow established PDS precedent and procedures (from the establishment of Memoranda of Understanding through to the delivery of PDS-compliant archive bundles).

For *future* projects (or for active *legacy* projects where the teams have expressed an interest in improving their software for the purpose of archiving), we will provide consultation and feedback on design and implementation of software tools and pipelines toward the purpose of producing more *sustainable* (and, by extension, easier to archive) software that meets both

Node and PDS standards and, where applicable, produces high-quality data suitable for archiving in the PDS. Brazier or Million will perform this consultation, with assistance from other Node or Federation staff or management as needed, and they will draw upon their decades of experience in pipeline creation, software engineering, and software project management. In many cases, the provided assistance will amount simply to suggesting ways in which basic software project management or "sustainability" best practices can be followed. These include: writing software in small and easily testable units, limiting features to those absolutely necessary, defining and refining necessarily software functionality prior to *coding* (i.e. creation of a Concept of Operations document [34]), limiting solutions to single languages when possible, not creating unnecessary GUIs, etc. Many of these advices will be common to most projects, and so we will publish a "Software Node Data Preparation Guide" or FAQs (analogous to guides for data preparation provided by other existing Nodes) with a high level overview of archiving requirements, the archive process, and suggestions like those listed above to generate products that will be both of maximum value and eminently suited for archiving. From the earliest stages of new project consultation, the Node Scientist will also provide data preparers with a non-binding assessment of whether their products are likely to be of sufficient scientific importance, by the PDS Software Node standards, to be eligible for archiving; we do not want someone to work hard to prepare a product for submission if there's a very good chance that it won't get past Software Node peer review.

We anticipate that Software Node archiving requirements will impose little additional expense on future missions over and above the normal expense of preparing for other data for PDS archiving, and we also expect that adherence to the requirements will result in long term *efficiencies* for both the teams and the Agency. The minimum Node requirements are generally congruent with basic software engineering best practices: document the system, its dependencies, and tests. Marginal costs for software archiving will be lowest when teams work with us from the early stages of product design and appropriately plan for submission of software to the archive. Modifications to existing or in-development software for archiving could be more time-consuming, difficult, and expensive, so the designation of a project as *legacy* or *future* has importance. For active or in-development NASA missions, that designation will be made by Node staff, in consultation with the mission teams and the Lead Node(s), based upon the status of design and development of the associated software and available resources, with the understanding that we will seek to not impose *unbudgeted* or *unplanned-for* additional effort or cost on work that is already funded and in-progress.

E. Peer Review

Prior to archiving, products must undergo evaluation by a panel of individuals with relevant professional expertise. Specifically, each product will be *peer reviewed* by at least 3 volunteer panelists who are both *qualified* to assess the product(s) under review and also *have no conflict of interest*. Volunteers will be solicited through multiple channels including announcements in community-wide mailing lists (e.g. Planetary Exploration Newsletter, DPS

Mailings) and direct requests by Node staff to specific individuals with relevant work or publication histories. The criteria for determining whether any potential panelist is "qualified" will vary depending on the context and nature of the product review, but we would seek to have a cross-section of the following on the panel: domain-specific scientific expertise, domain-relevant technical or engineering expertise, a working knowledge of the programming language(s) used in the product(s) under review, and an expressed interest in future *use* of the product(s) under review. Determinations for *qualifications* of reviewers will be at the discretion of Node staff while conflicts of interest will be determined subject to the criteria listed below.

A conflict of interest exists when a reviewer or member or someone in the member's immediate family has personal biases that may interfere or appear to interfere with his or her impartial judgment of the product. Such conflicts can arise from related financial interests, a supervisory relationship or collaboration with an employee or student, or a potentially competing research program. Each member will be informed that it is her/his responsibility to disclose any potential conflict of interest before conducting a review and to recuse him or herself from participating in a matter in which they are conflicted. In addition, the following specific conditions for participation will be enforced:

- The reviewers or members of their immediate family must not have active, very recent or near future significant financial interest with the Planetary Data System. (Previous PDS review panelists, for example, would be eligible.)
- The reviewers or members of their immediate family must not have active, very recent, or near future significant financial interest at the home institutions the project PI(s) or any of the scientists or engineers directly involved in the creation of the product(s) under review.
- The reviewers or members of their immediate family must not have *any* past, current or near future professional role (whether compensated or uncompensated) on the project(s) producing the product(s) under review.
- The reviewers or members of their immediate family must not have had *any* past role in the creation of the product under review, including but not limited to as the author(s) of components or sub-sections of the product(s) under review (esp. in situations of code reuse).
- The reviewers or members of their immediate family must not be current, very recent, or near future paid investigators on a grant-supported research project with the PI(s) of the project(s) or the scientists or engineers producing the product(s) under review.

For all of the above, "immediate family" comprises "spouse, same-sex partner, or dependent children." The phrase "active, very recent, or near future" is defined as "within the past 12 months or anticipated within the next 12 months." Per [35], significant financial interest (SFI) is defined as a paid appointment or serviced (such as consulting), compensation in cash or kind or equity interest, with an estimated value of $5,000 in a calendar year. SFI does not include reimbursement for reasonable expenses.

Reviewers will be instructed to assess for completeness and clarity of the source code, documentation, and tests with the objective that a future researcher would be able to reproduce the primary functionality by rewriting, porting, or recreating the environment for and directly compiling the source. Reviewers will assess whether the source code and attendant documentation are comprehensible and reasonable based on their technical and scientific domain expertise, as well as addressing the question of whether the software is of sufficient scientific importance ("high-valued") that it *should* be preserved. The emphasis is on the completeness, value, and utility of archived source code, not its *"quality,"* so reviewers will specifically not assess for that. Per PDS policy, review panels will "distinguish liens as either major or minor. Minor liens are those intended to improve the data set, but which are not considered critical to the understanding and use of the data. Major liens are those severe enough to render the data set (or increment, in the case of dynamic data sets) to be unsuitable for scientific research." [36] Major liens must be resolved prior to data becoming "certified" for archiving.

Results from a trial of scientific code review by Petre and Wilson, 2014 [37] state that "most reviewers were frustrated at not being able to run the code as the first step of review;" as such, we will provide access to virtual machine images instantiating running versions of submitted products to reviewers (via Olive, described in G) when this is practicable.

In the case of large and ongoing projects—such as missions—there may be multiple or ongoing reviews (e.g. Preliminary Design Review, Final Design Review, etc.) involving the same parties and the same product(s) at progressive stages of design and development; for smaller, shorter, or less complex projects, only a single review may be required, and review of multiple independently submitted data products may even be undertaken simultaneously by the same panel. These determinations will be made by Node staff subject to circumstance and in consultation with the PDS MC and PM when appropriate.

In situations where *observational data* are being archived in some other Node, and the Software Node is not the Lead Node, but the associated "pipeline" is also being submitted for archiving, it makes sense to perform a simultaneous, joint peer review of the data and pipeline. We will either work with the Lead Node to identify qualified reviewers of both software and data, or identify our own qualified software reviewers as adjuncts to those selected by the Lead Node, subject to coordination with the Lead Node.

F. User Friendly Web Interface

The Software Node will be an active resource for the research community. We will make archived data products available and searchable through a public-facing website (the Web Portal) and provide a range of means to access and use archived software, subject to the licensing and use restrictions of the archived software's dependencies, e.g. for software written for the Windows OS. Based upon archived information, we will develop and provide scripts for the creation of running instances of the software either natively or in VMs on users' own systems, and host virtual machine images of software running in its native environment for end users to download direction to their own machines or via "one-click," in-browser access using Olive

Executable Archive technology. For non-graphical software, which includes many pipelines, we will provide remote API access to archived software running on our own servers. We will host and moderate a "Software Node Forum," through which to provide assistance on use of archived software, general advice and feedback on preparing software for archiving, and a space for community members to discuss specific pieces of archived software. We will also work with other nodes and the Federation to improve the use and utility of all PDS holdings.

Many portal features and services will be usable without login, including accessing archived products, reading Node-created guides or the Users' Forum, or browsing Olive or API support for specific products. For some end services—e.g., downloading VMs or running them on RedCloud or locally using Olive, commenting on the Forums, and possibly using APIs—we will require users to register and login with a username, password, and valid email address, subject to the policies of the Security Plan, as a means for us to track and manage user permissions and resource allocation and usage across our services. Each archived product will have its own website page or pages with documentation of the product and instructions for use.

For Web Portal design and the presentation of the initial features and material, we will contract and direct assistance through Cornell University's web design team at the Cornell Computing and Communications Center (CCC). Programmatic development of the web portal and its ongoing maintenance will be performed by the staff of the Center for Advanced Computing (CAC). Both teams are very experienced and highly qualified to perform these tasks. We will work and consult with the Engineering Node and User-Centered Design team to maximize utility and interoperability of the portal for end users and the PDS, and to adapt or reuse pre-existing capabilities or technology where appropriate and efficient. Brazier has experience in software requirements elicitation and will lead the Web Portal specification process. Million has experience with User Experience (UX) design and project management for the development of web applications and will lead this effort as well as ongoing maintenance and improvement activities thereafter.

G. New and Noteworthy Tools

We do not simply want to archive software for the sake of preserving it. The software is important because it can be *used* and built upon by researchers to extract new and exciting meaning, results, and value from the corpus of *observational* data already stored in the PDS. This is the full promise of reproducibility: not just to confirm what was done before, but to meaningfully build upon it without starting over from scratch. A key piece of making that happen requires minimizing the time it takes a researcher to find archived software relevant to their interests and assess whether it meets their needs in whole or part. We propose a range of exciting but technologically-solid approaches for this, from making source code highly discoverable in the Astrophysics Source Code Library to developing the technology behind the Olive Executable Archive—a project Vint Cerf advocates as a candidate for "digital vellum," [38] by which digital information of today will be preserved as vellum preserved print information of the past—to the needs of the Planetary community.

ASCL registration: The Astrophysical Source Code Library (ASCL) [39] is a searchable registry of scientist-written software related to astronomical research, with a current manifest of over 1050 entries. ASCL has the primary purpose of protecting the integrity and supporting reproducibility of research by making research software source code more discoverable for examination. That is, the ASCL makes research more transparent, falsifiable, and reproducible by making it easier to find and analyze the computational methods that the research depended on. An important secondary goal is to improve the efficiency of scientific research by encouraging code reuse and reducing "reinvention-of-wheels." **Codes registered in the ASCL receive a permanent, unique data object identifier (the "ascl ID") which is registered with the SAO/NASA Astrophysics Data System (ADS), and so citations of registered codes are trackable via the ADS.** [40] We have confirmed with the ASCL Editor, Alice Allen, that source codes related to planetary science are eligible for inclusion. As a Collaborator, Allen will help us register archived software with the ASCL, subject to their editorial policies, in order to increase its discoverability and improve trackability of its use through citation. The ASCL editorial policy states that criteria for inclusion are that codes must have been used in research that has been peer-reviewed or submitted for peer review, or research published as an accepted thesis. In practice, software which passes Software Node criteria for "high valued" but *is not* eligible for inclusion in the ASCL will be rare; in such cases the software will simply not be registered in ASCL until such time as it is eligible. Note that though the ASCL and the Software Node have common practical and philosophical objectives, one is a registry and one is an archive; these are complementary functions.

Installation procedures and scripts: Node requirements and peer review metrics both assess for whether core products contain sufficient information that the running source code could be reproduced along with its operating environment. Where that environment is relatively *modern*, users may wish to install the archived software natively on their own systems. Based upon Node staff's experience building running instances of the software within VMs as part of the Node compliance check, we will write instructions for native installation of software and, when appropriate and practical, create build or installation scripts. These instructions are, themselves, expected to obsolesce with time, and so will not be continuously supported (although we may accept maintenance, updates and input volunteered by members of the community through, e.g., the Software Node Users' Forum).

Non-graphical APIs: In cases where the functionality of an archived tool is purely non-graphical (e.g. a command line data pipeline), we may create Application Programmer Interfaces (API) to running VMs of those software on our own servers. For some cases, this option will let us legally expose the functionality of archived software to end users without violating licensing on its dependencies (e.g. when we're legally using Windows as a *server*). While creating the API "wrappers" is not conceptually difficult, maintaining that interface and managing resource

utilization of the application *can* be. This option will be reserved only for extremely useful software with functionality that is not available through other means.

VM creation procedures and scripts: More promising and useful in the long term would be for users to be able to build their own VM images of running software in its appropriate execution environment. Following on the method of [41], and based on Node staff's experience building and running instances of the software within VMs as part of the Node compliance check, we will write instructions and *scripts* for the creation of virtual machine images of archived software running in its native environment. This solution avoids issues related to the licensing of software dependencies, making the end user is responsible for obtaining those licenses.

VM images: When licensing permits, we will make full virtual machines running software in appropriate operating environments (as created during the Node compliance check) available for direct download to user machines.

RedCloud: "Cloud" refers to a service allowing the creation of instances of Virtual Machines on-demand to provide a scalable computing solution, leveraging the strengths of Virtual Machines while managing the use of resources in an efficient way with security controls on a per-instance basis, which is invisible to the clients; the client simply asks for an instance with specified characteristics, and the Cloud provides it. RedCloud, a service of the Cornell Center for Advanced Computing, runs on Eucalyptus, an open-source project designed for compatibility with Amazon Web Services' Elastic Compute Cloud (EC2), the market leader in the provision of cloud-based processing. RedCloud, in operation since Fall 2011, was built by the Center for Advanced Computing and has been serving the needs of researchers since then, by providing computational resources when required and expanding as needed, only charging researchers for the resources they have used.

Olive: Olive freezes and precisely reproduces the environment necessary to execute software long after its creation. It uses virtual machine (VM) technology to encapsulate legacy software, complete with all its software dependencies. [42] The VM is incrementally transferred from a web server (much as video is streamed today) and executed at the edge of the Internet on a desktop, laptop, private cloud, or cloudlet [43]. The Olive team, under Co-I Satyanarayanan and with funding from the Sloan Foundation and the Institute for Museum and Library Services, has built Olive and demonstrated its ability to resurrect over a dozen closed-source applications that are over 20 years old (some built for now-obsolete hardware). Users can browse an online library of available software and quickly assess, by using it, whether any specific resource meets their needs without having to download, configure or install it locally. Because changes to individual VM images do not modify the master image, of which they are clones, users can quickly make changes to source code and recompile within the Olive VM without the need to locally reproduce hardware and software dependencies, port the software to modern environments, or recreate (i.e.

rewrite) the functionality entirely. Getting old, legacy software to work on modern hardware and operating systems can be a difficult and time-consuming process, possibly enough that the effort is abandoned and potential science lost. Olive dramatically decreases the "time to value" for users of archived software.

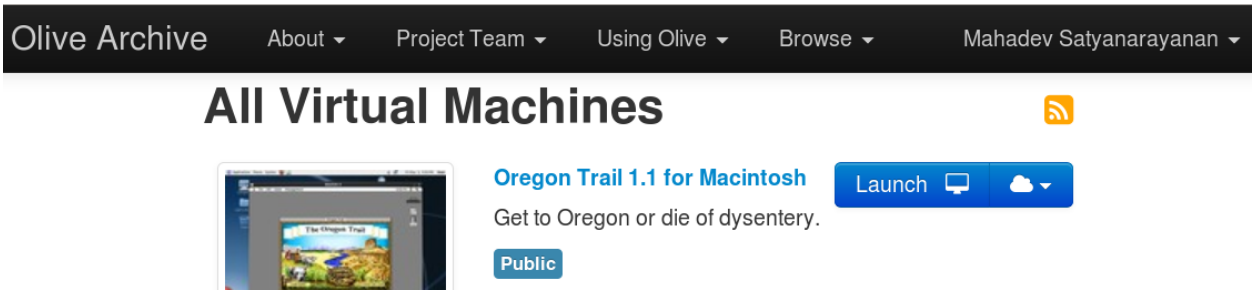


Figure 1. Olive technology provides one-click access to virtual machines from within a web browser. Researchers can quickly assess *by using it* whether the software meets their needs.

The Olive VM abstraction is implemented by the KVM/QEMU virtual machine monitor (VMM), which is a standard part of Linux distributions. Virtualization in Olive refers specifically to *hardware virtualization* of the Intel x86 architecture, and the term ``VM'' refers to a virtualized x86 machine. Today, this hardware architecture is dominant and is efficiently virtualized using Intel's VT extensions. Critically, however, Olive VMs are capable of archiving software written for other hardware architectures by using an additional layer of emulation nested within the x86 VM.

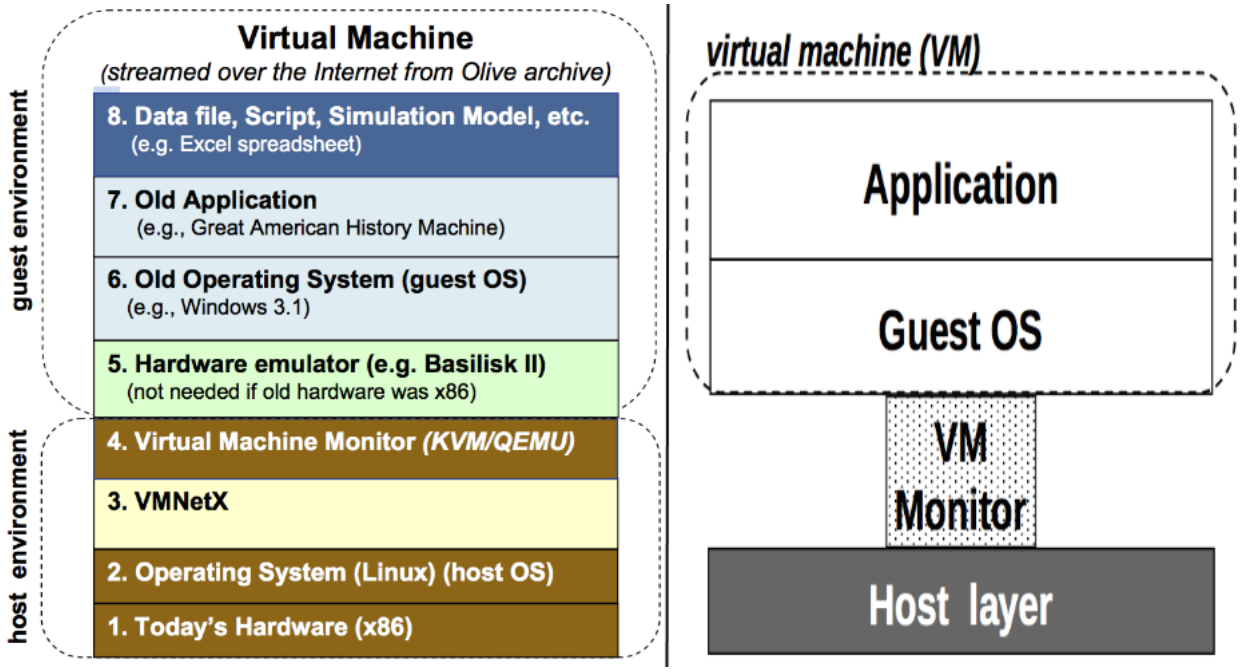


Figure 2: The conceptual structure of an Olive client.

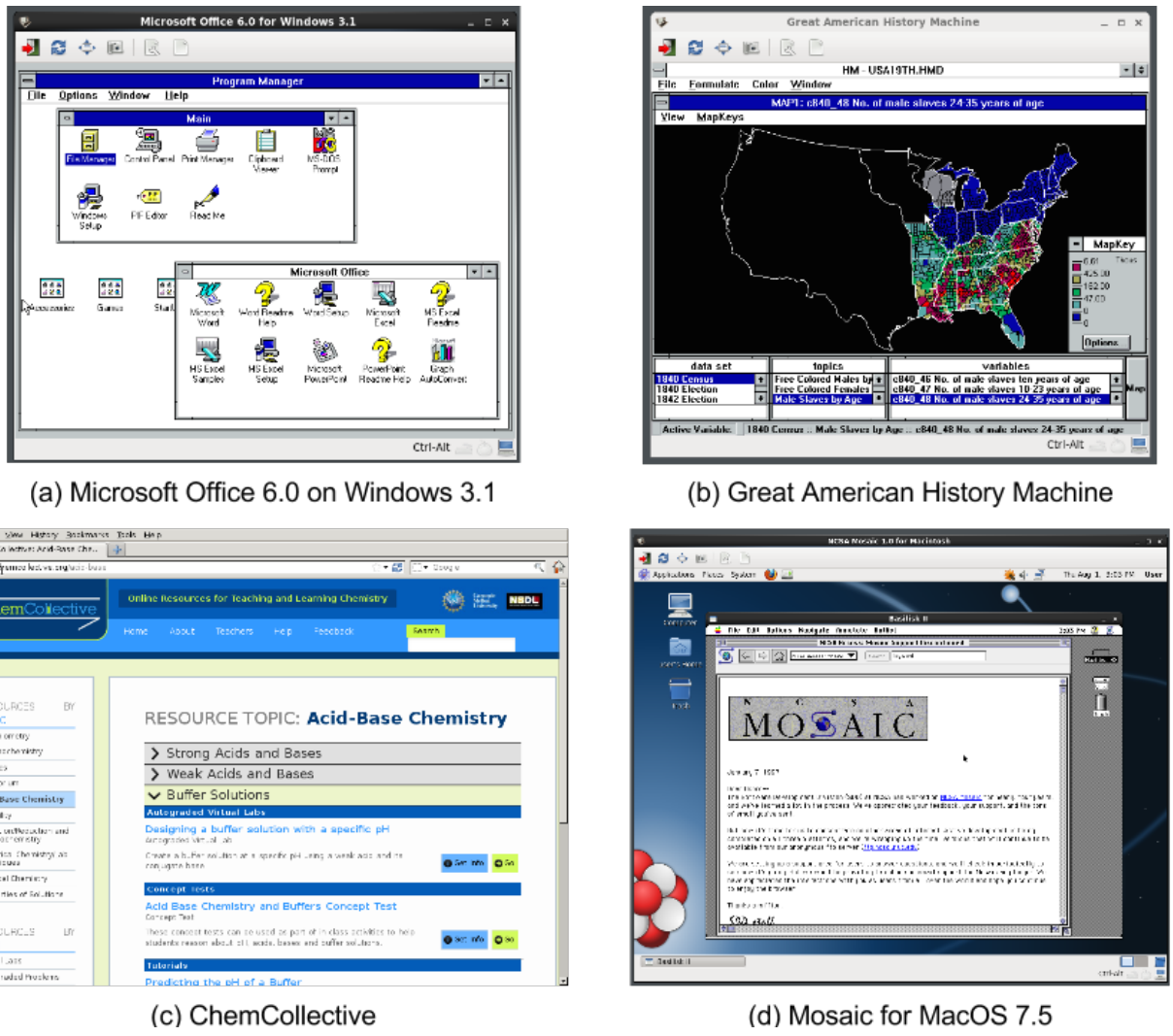
Since a typical VM image can be many gigabytes in size, Olive does not require it to be completely downloaded before it is launched. Instead, Olive adopts the approach of *streaming* used by video sites such as YouTube. Users launch and begin interaction with a VM as soon as a modest prefix has been transferred. Unlike a YouTube video, a VM instance is unlikely to access its VM image in simple linear order; it is the complex runtime behavior of a VM instance (including user interactions, data dependencies, temporal locality and spatial locality) that determines the reference pattern to its VM image. Therefore, Olive uses *demand-paged execution* of a VM instance over the Internet combined with *persistent caching* (i.e., caching on local disk or SSD) of VM state that has already been fetched. Using this approach, Olive achieves resume latencies of just a few seconds for typical VMs from well-connected Internet sites. Since demand paging and caching are completely transparent to the archived software within the VM, execution occurs without any awareness of the streaming process by the end-user.

Figure 2 illustrates the conceptual structure of an Olive client. At the bottom (Layers 1 and 2) is standard Intel x86 desktop or laptop hardware running Linux (generically called the "host operating system"). Layered above this (3) is an Olive component called VMNetX that implements caching and prefetching of VM images over the Internet. VMNetX presents the illusion of a fully assembled VM image to the VMM layer above, which virtualizes the x86 host hardware. Layers 5 through 8 in Figure 2 are encapsulated within the archival VM image that is streamed from Olive servers. The lowest of these layers (5) is a hardware emulator that presents the illusion of now-obsolete hardware (such as Motorola 68040). This layer can be omitted if the archived environment targets x86 hardware. Layer 6 is the archived operating system (generically called the "guest" operating system). The virtual disk of the VM is managed by the guest operating system, and appears as a local file system to higher layers. Layer 7, which represents the archived application, is the focal point of interest in archiving. It is to support execution of this application with high fidelity that the entire edifice shown in Figure 2 is necessary. Layer 8 represents input (such as a data file) that is provided to the archived application. For example, when examining an old archived engineering drawing, Layer 7 might be the ISIS pipeline and Layer 8 would be files input to ISIS such as PDS image data.

The VM does not *have* to be run on the user's local machine; using the Olive Thin Client Server, the VM can be run on a configured cloud, such as RedCloud, and consumed via a thin VMNetX client running on the user's local machine. Where the network connection between the thin client server and the cloud is fast and suffers low network latency, which will be the case for the PDS Software Node, the performance is very good.

Using a standard web browser, a user can explore a collection of VMs and launch them. Descriptions of the archived VMs are displayed on the Olive web site, along with links to documentation and provenance and an icon to launch the VM. [Figure 1] As validation and a demonstration of the general applicability of the technology, the Olive team has created a collection of about 15 VMs that include operating systems and applications dating back to the

late 1980s, including emulation of 25 year-old hardware. Launching a VM is as easy as clicking on a link to a pdf file and launching its reader.



(a) Microsoft Office 6.0 on Windows 3.1

(b) Great American History Machine

(c) ChemCollective

(d) Mosaic for MacOS 7.5

Figure 3. Four examples of software "resurrected" by Olive.

For brevity, we describe only four of these test VMs below.

- Microsoft Office 6.0 with Word, Excel and PowerPoint for Windows 3.1. [Figure 3(a)]
- Great American History Machine was originally created in the late 1980s by Professor David Miller of CMU to teach American history using census and election data from the 19th and 20th century. With no financial resources to port the software to newer Windows platforms, it fell into disuse, and no modern equivalent exists. [Figure 3(b)]
- The ChemCollective is a modern web-based service for teaching chemistry with a growing collection of virtual labs, scenario-based learning activities, tutorials, and concept tests. The VM shown in Figure 4(c) represents a snapshot of the web service at one specific point in time. DNS name resolution in the VM has been set to redirect all

references to http://chemcollective.org back to the VM, thereby creating a closed world that is unchanged by modifications to the live website. [Figure 3(c)]

- NCSA Mosaic was the world's first widely-used web browser, dating back to 1992-93. This VM is also interesting because the version of Mosaic that it encapsulates was written for the Apple MacOS 7.5 operating system on Motorola 68040 hardware. The VM also encapsulates Basilisk II, an open source hardware emulator for Motorola 68040 on modern Intel x86 hardware running Linux. In spite of two levels of virtualization, performance is acceptable because modern hardware is so much faster than the original Apple hardware. [Figure 3(d)]

Olive is and will remain an open-source project, available for use by both the Federation and others. As Olive matures, we anticipate that it will become a profoundly valuable resource to the *archiving community* (not just the Planetary community) in the same way that development of the OODT data management system for PDS use resulted in Apache OODT [44], a tool now adopted widely across many disciplines and use cases. To this end, we have allotted for part-time consultation with an *archive consultant* at the CMU Libraries—one of few organizations with expertise in the archiving of "executable digital objects"—to ensure that new tools and techniques of software archiving developed in service to PDS objectives are consistent with current archiving standards and, when possible, also of use to the broader archiving community. The functional extensions and performance enhancements to Olive listed below have been identified and will be implemented by the Olive Subnode team under direction of Satyanarayanan at his consortium partner institution, Carnegie Mellon University (CMU), over the period of performance; in all cases, plans for implementation exist, but cannot be fully described here due to space concerns. The order in which we will prioritize these items will depend on the requirements of the specific software being ingested into the archive, feedback from the user community, and guidance from the Node Scientist. New functionalities or enhancements may be defined according to those same exigencies. The Olive team will also work and consult with the Node staff at Cornell to install, run, and maintain Olive tools on Cornell servers. (No PDS resources will be directly hosted by CMU.)

- VM Access to External Data Sets: Olive does not yet support access to external data sources from applications within a VM. Data must be manually copied into the VM (e.g. by network file transfer) before it can be used. A high-priority improvement will be to simplify and streamline access to large data sets (especially PDS-held observational data) from within Olive VM instances so that users can immediately operate on the data for which software was built.
- Cloud Execution and Non-Linux End-Points: An Olive VM executes on the user's computer. We will also provide a thin client service for external users to run the VM on RedCloud and consume it via the VMNetX thin client on their own machines. There are situations, however, where it may be necessary to execute an Olive VM elsewhere. Examples include ARM-based Android or iOS tablets, or when an Olive VM requires

much more powerful hardware than an edge device to execute with acceptable performance. To handle such use cases, we will extend Olive to support remote execution of VMs.

- Multi-Core Parallelism: The existing Olive prototype can exploit multi-core parallelism within a single machine. At the time of VM image creation, cores can be statically allocated up to the maximum available to the guest OS. As a first step to better supporting parallel scientific codes, we will extend Olive to allow launch-time specification of available cores. Users could then execute Node-created VM images of parallelized code on ever more powerful hardware without recreating the image.
- Compute Cluster Parallelism: In the current Olive prototype, exploiting cluster-level parallelism involves extensive manual configuration. We will extend Olive to greatly simplify this workflow, working with the user community to identify the scientific applications to use as guides in the design and implementation. The Olive team will leverage their experience with rapid VM cloning in the SnowFlock system [45].
- Support for GPU Acceleration: SIMD parallelism of Graphical Processing Units (GPU) is being leveraged more frequently by the scientific community computationally intensive simulations and models. Virtualization of GPUs has proven difficult, however, as there is no standardized external interface, and many interfaces are proprietary and inconsistent. If archived software uses GPU capabilities, we will attempt to extend Olive to support these devices on a per-application basis.
- Prefetching for Last-mile Networks: There will be situations where Node users will want to use Olive on low-bandwidth, high-latency last mile networks. Examples include from a laptop with 3G wireless or on a heavily-loaded Wi-Fi network at a conference. The key to successful *video* streaming is aggressive prefetching of frames, but nonlinear fetching of VM state is much more complicated than linear fetching of video. Preliminary experiments by the Olive team with history-based prefetching of VM state over last-mile networks in an experimental system called *vTube* [46] suggest a qualitative user experience can be achieved comparable to viewing video over last mile networks.
- VM Synthesis: Olive uses demand paging to achieve rapid VM launch and only fetches VM state as needed during execution. An alternative approach called *dynamic VM synthesis* is based on the observation that a large part of a typical VM image is devoted to the dependencies, and each VM customization is typically derived from a small set of common base systems (such as a fresh install of Windows). The compressed binary difference between the base and launch VMs (the VM "overlay") can be downloaded to the host and applied to the base to derive the launch VM from which instances are created. Dynamic VM synthesis avoids licensing issues of proprietary dependencies because, by construction, an overlay does not contain any disk state that was already present in the base and therefore no bits of proprietary dependencies. The onus would be on the user to acquire a compatible and properly licensed base. This would allow us to serve a much wider variety of VMs of archived software [47].

Managing resource utilization: We have budgeted for reasonable but not unlimited processing capabilities on Cornell's RedCloud cluster from which these services will be hosted. Access to Olive and APIs will be controlled by a username / password signon so that we can enforce and modify case-by-case usage limits among users. In a case where a user wanted to do *a lot* of intensive processing with Olive, for example, it may prove necessary for the user to provision their own local or cloud-based processing resources. Migration should be relatively easy because RedCloud runs on Eucalyptus, which is fully compatible with Amazon's Elastic Compute Cloud and Olive has also been shown to work with the OpenStack cloud computing platform; we will provide consultation with users in this situation. We also request that NASA provide us with access to their cloud and high-performance computing systems in order to deliver the best possible service and experience to end users.

H. Funds Management and Transparent Accounting

Cornell University has extensive experience in the financial management of cooperative agreements in support of the nation's research infrastructure. Over the past four decades we have run many national research facilities supporting science and engineering and, as part of this activity, our underlying business systems have undergone review multiple times by a federal science agency. The accounting and business systems comply with the government standards set forth in the OMB circulars as well as the newer Uniform Guidance (2 CFR 200). Cornell's financial management systems were fully overhauled within the past five years, including incorporating features for collaboration between institutions and the specialized reporting needed for large infrastructure projects. Our most recent audit in accordance with the federal Single Audit Act may be found in the Federal Audit Clearinghouse.

Should this proposal be funded, the award will reside in Cornell's Center for Radiophysics and Space Research, ("CRSR"). CRSR has significant experience managing large, complex, NASA awards. Examples of some of the larger, more complex NASA awards that have been housed in CRSR are the Mars Exploration Rover Mission Science Team award, NASA's New York Space Grant, the Spitzer Science Team award and the Stardust NeXT Science Team award. These awards all contained multiple, multi-year subcontracts. In addition CRSR has managed hundreds of other NASA and National Science Foundation awards for many decades. If awarded, all funds will flow from NASA directly to Cornell University pursuant to a Cooperative Agreement between NASA and Cornell University. The Subaward Division of the Cornell University Office of Sponsored Programs will provide a subcontract to Carnegie Mellon University consistent with Carnegie Mellon's proposed budget provided herein. Principal Investigator, Dr. Adam Brazier, will be responsible for ensuring that the services provided by Carnegie Mellon University are consistent with those promised in this proposal. Funds will be transferred to Carnegie Mellon University on a year to year basis. Dr. Brazier will be responsible for the Science and Technical reporting required per the terms of the Cooperative Agreement.

Mary Mulvanerton, Esq, Associate Director of the Center for Radiophysics and Space Research, will be responsible for the financial reporting required by the Cooperative Agreement.

I. Frequent and Effective Communication

Brazier will attend the thrice-yearly PDS face-to-face (F2F) meetings of the PDS Management Council, accompanied each time by at least one other Node Co-Investigator, where they will report on the status of the Node and ongoing projects, including but not limited to legacy and future software product ingest, availability of user services, development and improvement of user services including Olive, and work with and support of mission, instrument, and research projects at all phases of development. Brazier will attend monthly Management Council teleconferences with Million serving as backup, and both will attend in situations of need. In between and external to such meetings, we will also communicate directly with Project Management (the PDS project office, Program Scientist, Program Executive, and Management Council) by email and telephone when necessary to provide them with information as may be needed to effectively serve their roles. And we will communicate by email or telephone with other members of the Federation as necessary.

Beginning with the March 2016 Lunar and Planetary Science Conference, approximately six months after the anticipated project start date, at least one of either Brazier or Million will attend each LPSC, American Geophysical Union (AGU) Fall Meeting, and Division of Planetary Sciences (DPS) of the American Astronomical Society conference during the five year period of performance. It will be advertised—through the Node website, community-wide mailing lists like PEN, and, when possible, the conference proceedings themselves—that Node staff are available at these conferences for limited, scheduled one-on-one consultation with community members who plan to prepare software products for the archive or make use of archived software. In addition, the Node will host a workshops at least once a year at one of those conferences—for which both Brazier and Million will attend—covering topics related to either software data product *preparation* or archived software data product *use* (e.g. tutorials in the use of the Olive environment to support scientific investigations). Workshops will alternate yearly between LPSC and AGU with additional workshops at DPS in YR3 and YR5 of the period of performance. These major conferences serve slightly different communities within planetary science, and many researchers are only able to attend one; we want reach as many people in the field as possible.

The front page of the Software Node portal will feature a public news section (i.e., a blog) where Node staff can and will post frequent updates about Node and archive status and services. For announcements of particular importance (e.g. the archiving of a major instrument pipeline), we will issue releases to community-wide mailing lists such as PEN and DPS Mailings. In extreme circumstances—e.g. a catastrophic event resulting in sustained downtime—we may send a bcc'd direct email alert to all registered users of the software node.

As an additional feature of the portal, we will create, administer, and moderate a public Software Node Users' Forum—analogous to the ISIS Support Forum maintained by USGS

Astrogeology [48] or the forum hosted by the current Geosciences Node at WUSTL [49]—where end users of the Software Node data and services can discuss uses and minutiae of specific archived software or request assistance from either Node staff or other community members, and to which staff can post extra-archival information related to specific archived data. Node staff will also share a common, publicly listed Node email contact to which users can send questions or requests, and a staff member will respond in a timely manner.

J. Discoverability Across the PDS

A primary scientific objective and use case of the Software Node is that researchers will be able to reprocess or re-analyze PDS archived observational data with archived pipelines and tools or modified versions of those pipelines and tools to verify and improve upon prior work or address entirely new scientific questions. As such, one of the major goals of the Software Node will be to build and encourage increasingly tight links between Software Node resources (archived software, services like Olive VMs) and related data held by other nodes. In addition to the high-priority work item (described in G) to improve access to external datasets from within Olive, we will coordinate with other members of the PDS Federation to *define* those links in a useful and transparent way. We will also work with Collaborator Christian Mattman, the architect of Apache OODT, the data management system that underlies the current version of the PDS, to use the data management system itself to make links between observation and software data tighter and more explicit.

The Engineering Node has developed a PDS-wide RESTful metadata search API [50] which appears, as yet, to have not been fully adopted across the current PDS. We support the development of a single-point-of-contact metadata search tool for the entire Federation will work with Engineering Node to make sure that our holdings are searchable with the system they have designed. When software archived in our Node operates on observational data archived by some other Discipline Node we will not only put *forward links* (from our software to their data) but provide, in collaboration with those Nodes, *backward* links (so that people looking at the data can quickly find the software).

There are obvious overlaps between the mission of the Software Node and what appears to be the *intent* of the nascent IPDA Tools Registry. We will work with the Engineering Node to develop the IPDA Tools Registry as a useful resource for the community; in addition to indexing software archived in the PDS Software Node, the registry can also index material that wouldn't or didn't pass PDS Software Node peer review, or which has not yet been submitted.

K. Seamless Transition

The Software Node is not being established on top of an existing Discipline Node, and so there is a necessary ramp-up period where we establish the basic infrastructure and functionality. We have designated the first 6 months of the period of performance to this task, leading up to our "unveiling" workshop at LPSC in March of 2016. [Figure 4] We will work with the Management Council, other Nodes, and new projects during this ramp up, as appropriate, but will not have the

full set of required functionality until the end of the ramp-up period. Because we are not specifically "taking over" data from any existing node, there will be no gap or risk of gap in *existing* availability of the PDS archive or end users services because of the time required for the Software Node infrastructure to be put into place and operations to commence.

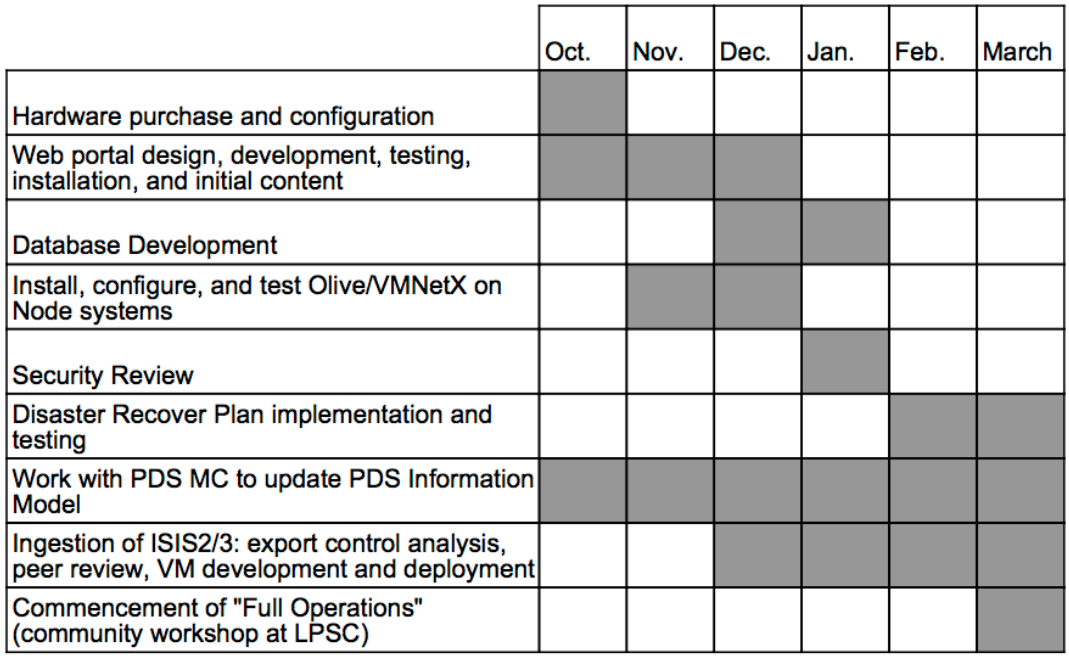


Figure 4. Initial six months of development of Software Node infrastructure.

We plan that, subject to export control analysis and peer review, ISIS3 and ISIS2 will be our first and second archived data products, respectively. These different versions of the Integrated Software for Imagers and Spectrometers created and provided by the USGS Astrogeology Research Program are excellent as first legacy products and based on our initial examinations we are confident that they will pass their peer reviews. The pipelines are of high scientific value, used for a large fraction of planetary research. The current version—ISIS3—is under active development and support whereas the previous version—ISIS2—no longer runs on modern hardware without substantial effort, although it has useful functionality that ISIS3 does not: for example, Co-I Hayes maintains a machine in his lab capable of running ISIS2 for specialized post-processing of Cassini RADAR images and Dr. Kirk (USGS) runs it within two layers of virtualization on his Mac to continue to use his photoclinometry tools (discussed in section A). Both software suites are well documented, and we will have the direct assistance of the ISIS project manager, Collaborator Trent Hare. Archiving these legacy codes ourselves as the first new Software Node products will give us an opportunity to run through all steps of software archiving—including the definition of an appropriate metadata template (which may require agreeing updates to the data dictionary or changes to the PDS4 Information Model), conducting the first software peer review, ingesting into the archive, and building the first public deployment

of an Olive VM. We anticipate presenting at the March 2016 Lunar and Planetary Science Conference (6 months after project start) a workshop introducing the Software Node, informing the community of our mission, soliciting community feedback, and "demonstrating" archived versions of ISIS2 and ISIS3. The workshop at LPSC will introduce the Software Node to the community and mark the point at which we can begin fully supporting *new investigations*.

As discussed, the current PDS Discipline Nodes already host some software as supplements to specific data, and those will be candidates for migration to the Software Node, subject to a Node and peer assessment of the value of such products for archiving as primary data products. In such cases of software already archived in the PDS—which was archived for use as supplementary information or documentation of other data—we will *not seek to remove it from the collection in which it already exists*, but we will copy and re-ingest it to the software node as a primary data product, subject to our policies, and possibly with additional supporting data, metadata, and information as may be available, and with appropriate references back to its collection of origin. Where other Discipline Nodes are interested in this information, we will supply for them pointers/links to our own archives of that software.

At the requested level of support, we estimate that we can archive ("restore") somewhere between 2 to 10 legacy data products per year (on top of other Node functions including administration and supporting *future* projects at all phases) with an estimated range of required effort between 2 person-weeks to 3 person-months per legacy product. The exact amount of time and effort required to archive any specific legacy data product will vary as a function of its size, complexity, age, the level of support available (i.e. availability of developers), documentation, etc. That is, simple, recent, actively supported, and well-documented legacy software will take much less time to archive than complex, old, unsupported, and poorly-documented legacy software. PI Brazier and Co-I Million are experienced with software estimation and will develop and document the process in order that estimates keep improving based on experience. Prioritization of legacy software for archiving ("triage") will be at the discretion of Node staff based upon available staff time and resources versus estimated time-to-archive, as well as perceived relative scientific value of the software as determined by Node staff (especially the Node Scientist) and in consultation with the PDS MC and members of the scientific community. Archiving of legacy products at a higher rate will require additional support. We will also gladly accept legacy data products that have been prepared for archiving by community members unaffiliated with the Node, and we anticipate that such activities could potentially be supported by the Planetary Data Archiving, Restoration, and Tools (PDART) program (for example).

Key Personnel

The proposed work is innovative and on the cutting edge of archiving practice. In addition to expertise in archiving, data management, and Planetary Science, it will require talent in software recovery, engineering, project management, and maintenance. Our team composition reflects this and contains all of the skills needed to meet the Node objectives.

- Dr. Adam Brazier (PI, Node Manager) is a Computational Scientist at the Center for Advanced Computing at Cornell University and has a decade of experience in scientific software engineering, scientific workflow design, database design and use, data management and archiving, software requirements elicitation and research in the physical sciences. His work includes leading the data management on the North American Nanohertz Observatory for Gravitational Waves (NANOGrav) [51] and the PALFA pulsar survey [52,53]. He also served as the CCAT Science Software Architect, for which his responsibilities included leading the observatory software requirements elicitation and design of the observatory data flow and archiving for the entire CCAT Telescope project [54].
- Chase Million (Co-I, Deputy Node Manager) is Founder of Million Concepts LLC, a software contracting and consulting company that develops scientific research software—including calibration and analysis tools—in the fields of spacecraft-based Planetary Science and Astronomy. He has broad experience with PDS-held data, and worked on the MER Pancam [55] and Odyssey THEMIS-VIS [56] calibration pipelines. He led the effort to recreate core functionality from the completely obsolesced Galaxy Evolution Explorer (GALEX) calibration pipeline (on which he was a developer) to calibrate and archive the approximately 1.1 trillion detected ultraviolet photons events at the Mikulski Archive for Space Telescopes (MAST) [57], making this data available for the first time. For the purpose of this proposal, his affiliation is with Cornell University and, if the Software Node is funded, he will become an employee of Cornell under this award.
- Dr. Alexander Hayes (Co-I, Node Scientist) is an Assistant Professor of Astronomy at Cornell University. He has broad domain expertise in Planetary Science that includes planetary surface processes, spectroscopy, instrument development, and extensive mission experience at all phases. He is a Co-I of the MastcamZ investigation on Mars2020, a participating scientist collaborator on the Mars Science Laboratory [58], a participating scientist of the Cassini-Huygens mission (associated with the RADAR and VIMS instrument teams) [23,59], and served many roles (including generation of PDS-compliant reduced data products) as a collaborator on the Mars Exploration Rovers [60,61]. He is also the Director of the Spacecraft Planetary Imaging Facility (SPIF) at Cornell University [62].
- Dr. Mahadev Satyanarayanan (Co-I, Olive Subnode Manager) is the Carnegie Group Professor of Computer Science at Carnegie Mellon University. He is a pioneer in distributed and cloud computing and was the principal architect of the Andrew File System (AFS) [63]. He is also the PI of the Olive Executable Archive project, which he will continue to develop and improve access for its applications in the Software Node. He will also provide support and guidance on problems related to the recovery and archiving of executable content.

Management and Feasibility

Brazier and Million will serve as Node Manager and Deputy Node Manager for the software node at 0.5 FTE and 0.55 FTE respectively. Brazier is responsible for the direction and success of the project, but Brazier and Million will work closely together on all aspect of Node management, administration, and policy and, in concert with CAC consultants (see below), many technical aspects of the project. Both have many years of experience in space science research, software development and engineering, and project management. Hayes will serve as the Node Scientist, supplementing the primarily engineering experience of Million and Brazier with his perspective as an active planetary research scientist and representing that key element of the PDS Software Node clientele; where questions of Node decisions, policies or priorities intersect with issues of *scientific value* or *utility*, we will first turn to Hayes. Hayes will participate actively in Node activities but less often with the larger Federation, and his commitment of 0.08 FTE (~1 month per year) is appropriate.

Olive Subnode: Technology like Olive is critical to the mission of providing *usable* legacy software running in a faithful emulation of its original environment. The team at Carnegie Mellon University are the world leaders, with the ability to improve and adapt the software to serve the needs of the PDS. Under direction from Brazier regarding requirements and priorities, Satyanarayanan (as Olive Subnode Manager) will lead his team to achieve those goals. There will be three broad categories of support provided by the Olive Subnode: (1) implementation, guidance and consultation in the creation of Olive-compliant virtual machines with newly submitted products, which may require modifications or enhancements to the Olive system itself (2) integration and ongoing maintenance and troubleshooting of the Olive system running on the Cornell Software Node servers, and (3) core development of new functionality in Olive, to be prioritized based on usage, need, and user feedback. The Olive subnode and the PDS Software Node at Cornell will be in frequent and regular communication regarding progress, goals and schedule, to ensure an integrated and user-friendly suite of products produced with an appropriate division of effort based on the relevant expertise.

CAC Consulting: We have budgeted for the equivalent of 0.5 FTE (1040 hours) of Center for Advanced Computing (CAC) consultant time. The CAC maintains a professional staff of highly skilled people with a range of expertise who can be contracted, as needed, by members of the Cornell community. They have experts in a range of programming languages, database and web development and hosting, information security, data management, data recovery, etc., etc. Because of the tremendous breadth of data products that we expect to encounter (written in different languages and from different *eras*), we don't know in advance what combination of these skills we might need. For purposes of the work plan, we are treating the CAC as an additional half time member of Node staff who has *precisely the skills needed* for the task at

hand. Many of the recurring activities associated with hardware and software maintenance will be performed by CAC staff, but we will also call upon them for work in legacy software restoration, testing Node requirements (like building VMs and running tests), and development of end-users services (like installation scripts or pipeline APIs). It would be difficult, if not impossible to find a single person with this amount of flexibility combined with the same high level of expertise. Brazier now works at the CAC and previously worked with CAC experts for nearly 10 years; these are his colleagues and coworkers, so there will be no management inefficiencies arising from misunderstandings or unfamiliarity with CAC personnel or their skills.

Feasibility of Effort: A Software Node is a new service, so there is some prediction uncertainty inherent to the timing of the project work plan related to which products are submitted, their state and the cadence of submissions. Additionally, the existence of a Software Node will likely change opinions and policies regarding software archiving. A confident assignment of specific hours to specific activities would, therefore, be inappropriate. There are common and recurring tasks for which we can *plan* (many of which are documented in the *IT Security Plan* and *Disaster Recovery Plan*) but a large amount of our time will be delegated according to circumstance and need. The team is *fully qualified* to manage this uncertainty.

To demonstrate that the budgeted amount of work effort is *reasonable* to achieve our objectives, we can perform the following rough calculation; as is typically the case in software project estimation, assumptions are necessary. All of the node administration, management, and non-Olive technical work will be performed by the combination of Brazier, Million, and CAC staff. Between the three, we have budgeted 1.55 FTE or the equivalent of 3224 hours per year of work effort. In one year, on average, mandatory travel by Brazier and Million for MC meetings (two people, three day trips thrice a year) and conferences (two six day trips for one person and, for a workshop, one six day trip for two) will take ~400 hours (~0.2 FTE). We estimate that ongoing maintenance of hardware and services (largely to be performed by CAC as detailed in the *IT Security Plan* and *Data Recovery Plan*) is another 260 hours (0.125 FTE). General "administration" which includes drafting requirements and procedures, other document preparation, communications with PDS MC and Management (including teleconferences), moderation of the forums, etc., is estimated for 0.5 FTE. This leaves ~0.725 FTE (~1500 hours) per year for us to work with new/future projects and seek out/restore legacy data. Estimating a mean 70 hours for work with future projects (start to finish) on average (missions will be much more work whereas small R&A projects will be less) and a little over 6 weeks on average (260 hours) for restoration of legacy software (and, again, with large dispersion), we estimate that we can support 8 to 10 future projects and 2 to 3 legacy projects or 1 to 2 legacy projects and < 15 future projects per year (or any other combination that respects those tradeoffs). The amount of extant software expected to meet the criteria for archiving in the PDS Software Node is sufficiently large that prioritization will be a tool both to ensure maximum return on expenditure

but also to ensure a manageable continuity of effort, when combined with the flexibility of the CAC work-effort allocation.

The above estimates implicitly assume that all work on future projects goes into a single year, which will rarely be the case in practice. For most types of projects, work effort will be spread over several years, so the number of future projects that we can simultaneously support is higher. We note that if archiving software in the PDS becomes *very popular* or common, such as if it is made a *requirement* across all NASA-funded planetary science missions *and* R&A projects, it is possible that the volume of products would be too large for the Software Node to handle without additional support.

Summary

The proposed Software Node is a timely and necessary addition to the Planetary Data System. By preserving software, this archive will ensure the survival of information and capabilities necessary to derive maximum possible value from the observational data already contained in the PDS. The work will be cutting-edge but built with existing, proven techniques of software requirements elicitation and documentation, testing, and virtualization. Archived data and metadata will be fully compliant with PDS4, and we will work with the DDWG to update the information model as necessary to fully describe software. We will also work with the Engineering Node and other Discipline Nodes to ensure that the metadata is *searchable* throughout the network and that a strong conceptual link is built and maintained by software and its corresponding observational data.

The Software Node will work with new mission, instrument or research teams at all project phases to ensure smooth and compliant archiving of valuable software. The scope of this node fits naturally into the existing PDS workflow. We will also seek out, acquire, restore, and archive *legacy* software from past and current missions. All products will undergo peer and compliance review prior to archiving in the PDS. In recognition that almost every member of the community both produces and consumes software, we are making ourselves very available to researchers through multiple forums as well as hosting regular training workshops at major conferences.

The team is composed of experts in scientific data archiving, software engineering, data and project management, planetary and computer science. Our expertise in planetary informatics is further reinforced by the inclusion of Collaborators Hare and Mattman in advisory roles. The Node data recovery and security plans, attached as appendices, are thorough and implement high uptime of user services with high stability of the archive.

Finally, we propose a range of new and exciting tools for software data discovery and use including ASCL registration of codes, creation of installation or VM build scripts, direct download of VM images, or one-click and immediate access to *running* examples of archived software via Olive Executable Archive technology.

Prior Experience

Dr. Adam Brazier (PI, Node Manager)

In 2012-2013, Brazier led the requirements elicitation for Scientific Software for the CCAT Telescope project, a 25m submm telescope to be built on Cerro Chajnantor in the high Atacama desert. In this effort he led a team made up of 17 Professors, postdoctoral researchers and software engineers, most of whom were initially unfamiliar with the practice of producing software requirements, to produce over 250 high-level, dressed Science Use Cases and a team of 8 engineers and software developers to produce over 70 Instrument dressed Software Use Cases. Following this Brazier produced the requirements for the end-to-end CCAT Observatory Data Management, broken into Observatory Data Management, Data Transport, Data Capture, Observatory Status Database and User Management, estimated data rates for the observing year and assessed the cost of the CCAT archiving policy.

Brazier leads the NANOGrav consortium's Data Management under the new NSF Physics Frontier Center award. In addition to the high-level system design and database implementation to store complex and iteratively processed pulsar timing products, in 2014-15 Brazier has been engaged with planning international data sharing with International Pulsar Timing Array partners and developing a Red Cloud-based solution for student workshops.

In his ongoing work running data management for the PALFA pulsar search consortium, archiving the data and data products from a very sensitive legacy survey using the Arecibo Radio Telescope, in 2014-15, Brazier has adapted the data management infrastructure for a new processing mode, specified new hardware, worked with external developers to build tools which access the databases he designed and managed, managed available space and streamlined data delivery from the telescope.

Since 2013, Brazier has been providing a backup for the Arecibo Radar dataset (22TB and growing), maintaining the backup and ensuring the hardware and software environment remains sound, and securely distributing the non-proprietary element of the dataset to a third site at UCLA.

In his work as a Cornell Center for Advanced Computing (CAC) Computational Scientist since August 2014, Brazier has designed and implemented: the database to serve as the backend for a complex social networks experimental tool; worked as part of a team to produce CAC's CloudLaunch product, allowing the creation of Virtual Machine clusters on RedCloud; and designed and executed the database-backed solution for ingesting and de-identifying IRB-controlled personal data from Cornell Massive Online Open Course (MOOC) offerings. Additionally, he has given workshops on Best Practices in Software Engineering, Workflows and Data Management, and Data Transfer.

Chase Million (Co-I, Deputy Node Manager)

As the Founder of Million Concepts LLC, a research consulting company that provides software engineering, design, and development on contract to research scientists primarily in the fields of planetary science and astronomy, Mr. Million has extensive experience at all phases of

software project development from requirements elicitation to estimates of effort and budgeting to implementation. He also has extensive managerial experience, both direct (as an employer-employee) and indirect, as project manager in charge of organization for teams of up to a dozen. He has experience with planetary and astronomy spacecraft missions and data management from pre-launch to post-project. He worked in the past on the development of the calibration pipelines for the MER Pancams and THEMIS-VIS instruments. He was the lead engineer of the Galaxy Evolution Explorer (GALEX) spacecraft ground calibration pipeline, a large and complicated software suite which he inherited in an utter state of disrepair. Shortly after the mission ended, the pipeline obsolesced completely and many attempts to create an environment in which it would run were unsuccessful. In the last three years he has led an effort, with support of the Mikulski Archive for Space Telescopes (MAST) at Space Telescope Science Institute (STScI) he has lead a project called gPhoton to migrate significant fractions of the functionality and also archive the ~1.1 trillion individual photons observed by this spacecraft to enable short time domain photometric studies in the UV. This valuable class of research was simultaneously not supported by the original mission calibration pipeline--data was released as integrated images--and *would not have been possible* without the ability to build upon the capabilities of the original pipeline. Though astronomical (instead of planetary), it's a perfect example of the need to preserve these software.

Alexander Hayes (Co-I, Node Scientist)

Co-I Hayes is an Assistant Professor in the Astronomy Department at Cornell University. His research focuses on comparative planetology and solar system exploration, specializing in the design, calibration, and operation of remote sensing instruments on unmanned planetary spacecraft. Currently, his group consists of fifteen researchers and staff who are studying a broad range of phenomena throughout the system, including Venus, Earth, Mars, Churyumov Garasimenko, Io, Europa, Enceladus, Titan, Saturn, and Saturn's Rings. Hayes has over 10 years of experience in utilizing spacecraft-based platforms to study the properties of planetary surfaces, including an engineering background in instrument design and calibration. His flight project experience includes MER, MSL, the Cassini-Huygens Mission to Saturn, and Mars2020. Hayes is also actively involved in mission and instrument proposals to the NASA Discovery, New Frontiers, Europa Clipper, and ESA M5 opportunities. In support of these flight projects, Hayes has participated in the design and implementation of multiple data calibration and reduction pipelines and is the lead for the Calibration Working Group of the Mars2020 Mastcam-Z investigation. Prior to joining Cornell, Hayes worked at MIT Lincoln Laboratory on the design, characterization, and deployment of infrared focal planes / telescopes for Ballistic Missile Defense and Tactical Air Defense programs. Hayes is also the director of the Spacecraft Planetary Image Facility, one of the 17 NASA-sponsored Regional Planetary Image Facilities that archive, distribute, and provide support for the use of NASA planetary mission data by researchers, students, and the general public.

Dr. Mahadev Satyanarayanan (Co-I, Olive Subnode Manager):

Co-I Satyanarayanan is the leader of the Olive ("open library of images for virtualized execution") project, which grew from his earlier Internet Suspend/Resume project and freezes environments and Virtual Machine (VM) state, and includes a "VM Player" which allows local execution of a remote VM, for recovery and preservation of legacy codes. The Olive library includes VMs running software dating back to the 1980s, software which ran on hardware obsoleted in the mid-1990s and runnable snapshots of evolving web applications. Through the VMNetX client software the VMs and applications running on them start in seconds, as compared to minutes for a cold reboot, with state preserved from the moment of their "freezing".

Dr. Satyanarayanan's Elijah project seeks to examine and enhance the convergence between mobile computing and cloud computing. Dr. Satyanarayanan first coined the term "cloudlet" to describe an entity which can integrate the interaction between a low-powered client and a high-powered computational resource to reduce operational latency and enhance user experience. In connection with this, Dr. Satyanarayanan has been exploring wearable cognitive assistance of devices such as Google Glass and Microsoft Hololens.

Dr. Satyanarayanan was an advisor to Maginatics, a cloud-based storage provider purchased by industry giant EMC in late 2014.

References

[1] Github Inc. Github. https://www.github.com, 2015. Online; accessed 12-April-2015.
[2] S. Hardman, J.S. Hughes, R. Joyner, D. Crichton, E. Law. "Deploying a Planetary Data Tool Registry." Second Planetary Data Workshop (2015).
[3] Data Design Working Group. "PDS4 Concepts." https://pds.nasa.gov/pds4/doc/concepts/PDS4_Concepts_2nd_Draft.pdf, 2nd draft (2012). Online; accessed 26-May-2015.
[4] "Planetary Data System Proposer's Archiving Guide (PAG)." https://pds.nasa.gov/pds4/propose/pds4-pag.pdf, v2.00 (2014). Online; accessed 26-May-2015.
[5] Personal communication with Dan Crichton (JPL).
[6] "Index of /pds/ODTGEO_v1/software/" http://static.mars.asu.edu/pds/ODTGEO_v1/software/, Online; accessed 26-May-2016.
[7] Christensen, Philip R., et al. "The thermal emission imaging system (THEMIS) for the Mars 2001 Odyssey Mission." *Space Science Reviews* 110.1-2 (2004): 85-130.
[8] "The Spice Concept." https://naif.jpl.nasa.gov/naif/spiceconcept.html, Online; accessed 26-May-2016.
[9] M. K. Dougherty et al., Space Sci. Rev. 114, 331 (2004).
[10] Cassini Magnetometer Raw Data Archive. http://ppi.pds.nasa.gov/search/view/?f=yes&id=pds://PPI/CO-E_SW_J_S-MAG-3-RDR-CALIB-SHM-V1.0/SOFTWARE/CAL02&o=1, Online; accessed 26-May-2016.
[11] Lee, E.M., Anderson, J. "Overview: Summary of ISIS." USGS. http://isis.astrogeology.usgs.gov/documents/Overview/Overview.html, Online; accessed 26-May-2016.
[12] Personal communication with Randy Kirk (USGS).
[13] Janssen, M.A., M.D. Hofstadter, S. Gulkis, A.P. Ingersoll, M.Allison, S.J. Bolton, and L.W. Kamp, "Microwave Remote Sensing of Jupiter's Atmosphere from an Orbiting Spacecraft", Icarus, 173, 447-453, 2005.
[14] Bolton, S. J. "The Juno mission." Proceedings of the International Astronomical Union 6.S269 (2010): 92-100.
[15] Maurice, S., et al. "The ChemCam instrument suite on the Mars Science Laboratory (MSL) rover: science objectives and mast unit description." *Space science reviews* 170.1-4 (2012): 95-166.
[16] Grotzinger, John P., et al. "Mars Science Laboratory mission and science investigation." *Space science reviews* 170.1-4 (2012): 5-56.
[17] Wiens R.C., et al. (2014) The SuperCam remote sensing suite for Mars 2020: Co-aligned LIBS, Raman, and Near-IR spectroscopies, and color micro-imaging. Abstract #1086, Second International Workshop on Instrumentation for Planetary Missions (IPM2014), Greenbelt, MD, November 4-7.
[18] Nolan, M. "Aricebo Planetary Radar." http://www.naic.edu/~nolan/radar/, Online; accessed 26-May-2016.

[19] Murchie, S., et al. "Compact reconnaissance imaging spectrometer for Mars (CRISM) on Mars reconnaissance orbiter (MRO)." *Journal of Geophysical Research: Planets (1991–2012)* 112.E5 (2007).
[20] Zuber, M. T., D. E. Smith, S. C. Solomon, D. O. Muhleman, J. W. Head, J. B. Garvin, J. B. Abshire, and J. L. Bufton 1992. The Mars Observer Laser Altimeter investigation. J. Geophys. Res. 97, 7781–7798.
[21] McEwen, Alfred S., et al. "Mars reconnaissance orbiter's high resolution imaging science experiment (HiRISE)." *Journal of Geophysical Research: Planets (1991–2012)* 112.E5 (2007).
[22] Marco Mastrogiuseppe, Valerio Poggiali, Alexander Hayes, Ralph Lorenz, Jonathan Lunine, Giovanni Picardi, Roberto Seu, Enrico Flamini, Giuseppe Mitri, Claudia Notar- nicola, et al. The bathymetry of a titan sea. Geophysical Research Letters, 41(5):1432– 1437, 2014.
[23] Elachi, Charles, et al. "Cassini radar views the surface of Titan." *Science*308.5724 (2005): 970-974.
[24] Office of Management and Budget. Circular a-110 revised 11/19/93 as further amended 9/30/99. https://www.whitehouse.gov/omb/circulars_a110/. Section 36.d.2.i. Online; accessed 12-April-2015.
[25] NASA. Nasa plan: Increasing access to the results of scientific research. http://science.nasa.gov/media/medialibrary/2014/12/05/NASA_Plan_ for_increasing_access_to_results_of_federally_funded_research.pdf. Online; accessed 12-April-2015.
[26] Matthews, B. M., et al. "An approach to software preservation." PV 2009 Ensuring Long-Term Preservation and Adding Value to Scientific and Technical Data (2009).
[27] Takhteyev, Yuri, and Quinn DuPont. "Retrocomputing as preservation and remix." Library Hi Tech 31.2 (2013): 355-370.
[28] Goodman, Alyssa, et al. "Ten simple rules for the care and feeding of scientific data." *PLoS computational biology* 10.4 (2014): e1003542.
[29] LeVeque, Randall J., Ian M. Mitchell, and Victoria Stodden. "Reproducible research for scientific computing: Tools and strategies for changing the culture."*Computing in Science and Engineering* 14.4 (2012): 13.
[30] M T Zuber, D E Smith, SC Solomon, DO Muhleman, JW Head, JB Garvin, JB Abshire, and JL Bufton. The mars observer laser altimeter investigation. Journal of Geophysical Research: Planets (1991–2012), 97(E5):7781–7797, 1992.
[31] PDS4 Information Model Specification Team. “PDS4 Information Model Specification.” v1.4.0.0 (2015).
[32] International Traffic in Arms Regulations (“ITAR”), “22 CFR (120-130)”
[33] Export Administration Regulations (“EAR”), “15 CFR (730-774)”
[34] IEEE Standard 1362-1998 - IEEE Guide for Information Technology - System Definition - Concept of Operations (ConOps) Document (2007).
[35] NIH Office of Extramural Research. “Financial Conflict of Interest.” http://grants.nih.gov/grants/policy/coi/index.htm, Online; accessed 26-May-2016.

[36] PDS Management Council. “Policy Statement on Implementing ‘Certified Data’ Procedures.” (2011)
[37] Petre, Marian, and Greg Wilson. "Code Review For and By Scientists." arXiv preprint arXiv:1407.5648 (2014).
[38] Rundle, M. “The Internet Is A ‘Black Hole’ Of History.” The Huffington Post. (2015). http://www.huffingtonpost.co.uk/2015/02/13/internet-history-vint-cerf_n_6676782.html, Online; accessed 26-May-2016.
[39] Allen, A., DuPrie, K., Berriman, B., et al., 2013, Astronomical Data Analysis Software and Systems XXII, 475, 387.
[40] Kurtz, Michael J., et al. "The NASA astrophysics data system: Overview."*Astronomy and Astrophysics Supplement Series* 143.1 (2000): 41-59.
[41] Kam Woods and Geoffrey Brown. Assisted emulation for legacy executables. International Journal of Digital Curation, 5(1):160–171, 2010.
[42] Olive. www.olivearchive.org. Online; accessed 13-April-2015.
[43] Satyanarayanan, M., Bahl, P., Caceres, R., Davies, N. ‘‘The Case for VM-based Cloudlets in Mobile Computing’’ IEEE Pervasive Computing, Vol. 8, No. 4, October-December 2009
[44] Mattmann, Chris A., et al. "A software architecture-based framework for highly distributed and data intensive scientific applications." *Proceedings of the 28th international conference on Software engineering*. ACM, 2006.
[45] LAGAR-CAVILLA, H., WHITNEY, J., SCANNELL, A., RUMBLE, S., PATCHIN, P., DE LARA, E., BRUDNO, M., AND SATYANARAYANAN, M. Snowflock: Rapid virtual machine cloning for cloud computing. In Proceedings of EuroSys 2009 (Nuremberg, Germany, March 2009).
[46] ABE, Y., GEAMBASU, R., JOSHI, K., LAGAR-CAVILLA, A., AND SATYANARAYANAN, M. vTube: Efficient Streaming of Virtual Appliances Over Last-Mile Networks. In Proceedings of the ACM Symposium on Cloud Computing (Santa Clara, CA, October 2013).
[47] HA, K., PILLAI, P., RICHTER, W., ABE, Y., AND SATYANARAYANAN, M. Just-in-Time Provisioning for Cyber Foraging. In Proceedings of MobSys 2013 (Taipei, Taiwan, June 2013).
[48] “ISIS Support. https://isis.astrogeology.usgs.gov/IsisSupport/, Online; accessed 26-May-2016.
[49] https://geoweb.rsl.wustl.edu/community/, Online; accessed 26-May-2016.
[50] Hardman, S. “PDS Search Protocol.” v1.2 (2014). Planetary Data System.
[51] MA McLaughlin. The north american nanohertz observatory for gravitational waves. Classical and Quantum Gravity, 30(22):224008, 2013.
[52] JM Cordes, PCC Freire, DR Lorimer, F Camilo, DJ Champion, DJ Nice, R Ramachandran, JWT Hessels, W Vlemmings, J van Leeuwen, et al. Arecibo pulsar survey using alfa. i. survey strategy and first discoveries. The Astrophysical Journal, 637(1):446, 2006.

[53] Benjamin Knispel, P Lazarus, B Allen, D Anderson, C Aulbert, NDR Bhat, O Bock, S Bogdanov, A Brazier, F Camilo, et al. Arecibo palfa survey and einstein@ home: binary pulsar discovery by volunteer computing. The Astrophysical Journal Letters, 732(1):L1, 2011.
[54] Thomas A Sebring, Riccardo Giovanelli, Simon Radford, and Jonas Zmuidzinas. Cornell caltech atacama telescope (ccat): a 25 m aperture telescope above 5000 m altitude. In Astronomical Telescopes and Instrumentation, pages 62672C–62672C. International Society for Optics and Photonics, 2006.
[55] JF Bell, SW Squyres, KE Herkenhoff, JN Maki, HM Arneson, D Brown, SA Collins, A Dingizian, ST Elliot, EC Hagerott, et al. Mars exploration rover athena panoramic camera (pancam) investigation. Journal of Geophysical Research: Planets (1991–2012), 108(E12), 2003.
[56] TH McConnochie, JF Bell, D Savransky, MJ Wolff, AD Toigo, H Wang, MI Richardson, and PR Christensen. Themis-vis observations of clouds in the martian mesosphere: Altitudes, wind speeds, and decameter-scale morphology. Icarus, 210(2):545–565, 2010.
[57] C Million, SW Fleming, and B Shiao. gphoton. https://www.github.com/cmillion/ gPhoton, 2014. Online; accessed 12-April-2015.
[58] John P Grotzinger, Joy Crisp, Ashwin R Vasavada, Robert C Anderson, Charles J Baker, Robert Barry, David F Blake, Pamela Conrad, Kenneth S Edgett, Bobak Fer- dowski, et al. Mars science laboratory mission and science investigation. Space science reviews, 170(1-4):5–56, 2012.
[59] Kevin H Baines, RH Brown, Dennis L Matson, RM Nelson, BJ Buratti, JP Bibring, Y Langevin, C Sotin, A Carusi, and Angioletta Coradini. Vims/cassini mission at titan: Scientific objectives and observational scenarios. In Symposium on Titan, volume 338, pages 215–219, 1992.
[60] Raymond E Arvidson, Steven W Squyres, Robert C Anderson, JF Bell, Diana Blaney, Johannes Brueckner, Nathalie A Cabrol, Wendy M Calvin, Michael H Carr, Philip R Christensen, et al. Overview of the spirit mars exploration rover mission to gusev crater: Landing site to backstay rock in the columbia hills. Journal of Geophysical Research: Planets (1991–2012), 111(E2), 2006.
[61] Steven W Squyres, Raymond E Arvidson, D Bollen, JF Bell, Johannes Brueckner, Nathalie A Cabrol, Wendy M Calvin, Michael H Carr, Philip R Christensen, Benton C Clark, et al. Overview of the opportunity mars exploration rover mission to meridiani planum: Eagle crater to purgatory ripple. Journal of Geophysical Research: Planets (1991–2012), 111(E12), 2006.
[62] JJ Hagerty et al. The regional planetary image facility network. In Lunar and Planetary Institute Science Conference Abstracts, volume 43, page 1548, 2012.
[63] James H Morris, Mahadev Satyanarayanan, Michael H Conner, John H Howard, David S Rosenthal, and F Donelson Smith. Andrew: A distributed personal computing environment. Communications of the ACM, 29(3):184–201, 1986.