Part 1: Table of Contents

Part 2: Identification and Significance of the Innovation

Summary of the Innovation

We propose to create a new storage, processing and retrieval database system for scientific data, with an emphasis on spacecraft remote sensing, that combines the power of cloud computing, relational databases, and fully integrated metadata to improve the efficiency, usability and scientific return on investment of spacecraft missions. The working title for this project, where spacecraft meet the cloud, is "Aerobrake." The core innovation of Aerobrake is the use of data definitions that correspond to the smallest meaningful units of raw observations -- in most cases, a single readout value of a solitary sensor pixel -- put into context by relational mappings to all available geometric, calibration, and state data. Structuring data in this way will vastly increase the flexibility of search and manipulation of remote sensing observations within a cloud-like storage and processing system. More finely granular data will permit greater parallel processing than currently possible. To minimize redundant processing, the database will permanently store intermediate and derived data (including calibrations, projections, model fits, etc.) and will preserve as additional metadata the relationships between all levels of abstraction of data processed within the system. This rich, metadata-driven relationality will effectively make all procedures lossless and reversible and empower users to perform quick and complex cuts, queries, and manipulations on data across multiple levels of processing and abstraction.

Summary of the State of the Art

Remote sensing data volumes have expanded dramatically in the last decade and will continue to grow even faster in the future. The volume of planetary spacecraft data stored in the Planetary Data System (PDS) is expected to surpass 1 petabyte sometime in the next few years [1]. Some individual instruments (like HiRISE on Mars Reconnaissance Orbiter [2]) now produce single images that are gigabytes in size; some highly active instruments (like Diviner on Lunar Reconnaissance Orbiter [3]) are expected to produce total archived data in the *hundreds* of terabytes.

This increase is matched or exceeded by terrestrial sensors. The next generation of Earth Observation (EO) satellites such as NASA's HyspIRI are going to gather "massively big hyperspectral data" (MBHD) -- the solicitation cites data rates of 800 Mbps -- that will revolutionize the richness and accuracy of information extraction and knowledge generation about our planet, but only if new techniques can be developed for managing the deluge. For example, to cover the entire planet Earth just once in 242 spectral bands, the EO-1 Hyperion imagers will require about 170 terabytes of storage [4].

Scientists who wish to use these data face a variety of barriers that make their research slow, difficult to budget, and sometimes entirely impossible. These barriers include both search costs (difficulties in simply figuring out whether the data needed exists at all, and if so, where in the corpus it

exists) and costs associated with additional post-processing and local storage.

In order to even determine which files might be relevant to their work, researchers must search on the files' metadata. However, while the metadata tags by which these files are organized are documented, they are inconsistently used, and new tags are often defined according to the exigencies of instrument operations or specific missions rather than in anticipation of subsequent science needs. Some geospatial information systems partly circumvent this problem by associating each supported mission's metadata with regions on planetary surfaces, allowing researchers to define latitude and longitude ranges and receive pointers to associated files. However, the researcher must still download these files locally to investigate them, and they might be terabytes in size. Thus, even when the data of interest are in an obvious location, simply downloading them can take hours or days and requires researchers to possess significant local storage capacity.

Furthermore, working with these data often requires proprietary software, which can be quite expensive or require recurring licenses that must be maintained through uncertain funding cycles. Such tools do not even exist for all data sets, and so researchers are often required to create hybrid solutions from available software, write their own tools, or do without. Even when the tools exist, many were created for prior generations of remote sensing instruments and make assumptions about end users and hardware that are no longer realistic -- for example, that the data set can be held in memory or contains a small enough number of dimensions to be meaningful to a human by visual inspection.

These problems all become worse when researchers wish to co-analyze datasets from separate instruments or missions. The likelihood of finding usable existing tools decreases greatly. It is not even unusual for a researcher to find that the data from each of the instruments they're examining in a combined study were differently calibrated and stored in mutually incompatible map projections. For instance, even though these instruments are on the same spacecraft, Compact Reconnaissance Imaging SpectroMeter (CRISM) [5] imaging data are usually map-projected using the 128-pixel-per-degree Mars Orbiter Laser Altimeter (MOLA) [6] shape model of Mars, while HiRISE 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 the data in order to get multiple datasets to line up in a region of interest. And since each distinct data set may be tens to hundreds of terabytes, even if software tools for processing the data *are* available, doing so requires access to a much larger computational infrastructure than would be required for a single-instrument study.

Not only are existing remote sensing data management tools inadequate for massive data volumes, they generally make expensive and wasteful assumptions for historical and optimization reasons that are no longer necessary with modern computing hardware. Most pervasively, almost all existing tools still treat remotely sensed *digital* data as *digitized photographs*: they assume that the data is continuous, lies on a regular grid, and possibly also orthonormalized. These assumptions almost never hold for real data. Pushbroom instruments, for instance, are very common and produce data that always violates these assumptions. Pushbroom pixels are distorted in the downtrack vs. cross track directions. While a full frame CCD imager, when stationary with respect to the scene, comes very close to collecting regularly gridded data, this feature is lost entirely when the camera is mounted on a moving spacecraft or when multiple observations from different instruments or different vantage points are combined. Thus, users are forced to rasterize their data as soon as they wish to start doing very much with it.

Rasterization, unfortunately, involves lossy and computationally expensive interpolations. This problem is well-known; the documentation for ArcGIS 9.2 states "There is also a loss of precision that accompanies restructuring data to a regularly spaced raster-cell boundary" **[7]** and many textbooks on remote sensing suggest that users choose their interpolation methods wisely -- between bilinear or bicubic and nearest-neighbor techniques -- in order to minimally impact required analyses **[8]**.

Historically, the information loss and computational expense due to rasterization has been relatively inconsequential for the most common classes of remote sensing investigations(e.g. morphological identification, quantification, and classification of *seen* objects) and the computational efficiency gains from array algebras were seen as a fair trade. However, interpolation destroys absolutely critical information (e.g., observation geometry and sub-pixel mixing) required as inputs to categories of investigation that are rapidly becoming crucial to instrument-based science, particularly hyperspectral or multi-instrument investigations. Note the following comment from *Hyperspectral Remote Sensing*:

> "From an information-processing perspective, one can simply extend the visualization and processing approaches for multispectral imaging to the point where the number of bands becomes very large and the width of the bands becomes very small. This viewpoint often leads to methods that simply extract subsets of the hyperspectral bands to support various applications. This approach, however, arguably overemphasizes the imaging aspect of hyperspectral sensing and fails to properly exploit its nature as a spectrometer. An alternate perspective is to emphasize the spectral sampling, resolution, and range required to capture intrinsic spectral characteristics of scene materials to support an information-processing application [...] Such information-processing problems become analogous to communications signal processing, where one typically wants to extract some sort of signal from interference and noise. [...] When hyperspectral sensing is viewed in the manner suggested, the importance of understanding all elements of the sensing chain -- from material spectral phenomenology through image data processing and analysis -- is clear." **[9]**

It is difficult to overemphasize the difficulties a raster-based computational paradigm presents to advances in the state of sensor-based science. **Deep analysis of multi-instrument and hyperspectral data *requires* a complete accounting of the sensing chain, but existing remote sensing tools almost universally *break* this chain.**

Of course, higher-order data products remain extremely valuable in multi-instrument and hyperspectral studies. Indeed, derivation and reduction of data is almost always required for meaningful analyses in this sort of science, as in any other. But all such operations, by definition, encode both the assumptions and intentions of the person designing them, and researchers working in these areas must (or should) start back from the least reduced possible data for new investigations.

Relevance of the Innovation

Here is the ultimate promise of Aerobrake: **If we're willing to deal with the added complexity and storage requirements associated with retention of sufficient metadata to describe data abstractions, we can make all such data abstractions lossless and reversible.** Some of the issues listed above arise from storage and retrieval limitations of hardware and software and so could be mitigated by leveraging cloud data storage and processing -- applying more resources in accordance with given paradigms. However, most of the issues are, fundamentally, due to insufficiencies and inconsistencies in available metadata.

Broadly, metadata is data that gives *context* to some other piece of data. Understanding whether specific data are of interest is a metadata search task. This task is impossible when the appropriate metadata are not available. Camera calibration procedures, standards, and software are all metadata. Camera *definitions* necessary to compute the geometry of a map projection from raw data are also a kind of metadata. It is a known problem, highlighted in a recommendation by the 2012 Planetary Data Workshop [1], that extensive and high-level metadata are not always made available by data providers, or are made available in nonstandard or difficult-to-use formats.

Another, less well-acknowledged, problem is that even if all of the previously mentioned types of metadata were available for every data set, we could still improve the usefulness and usability of the data set by improving metadata *handling*. This could be accomplished by directly linking metadata to their associated data in an easily-usable database architecture and by dramatically expanding the scope and volume of metadata associated with observations. A relational database is a kind of metadata management machine; by its very nature, it provides context for entries (as pointers to other objects). It is therefore the perfect tool with which to create a schema that will maximize the availability and usefulness of metadata associated with remote sensing data. These improvements would provide users with immediate access to the *total available context for every piece of stored data*. It would involve a major increase in the total amount of stored data. Historically, cost, size, and complexity considerations, rather than research considerations, have determined the quantity and quality of the relational metadata provided in databases of remote sensing data, but we are designing for cloud-based systems where shared infrastructure distributes the overhead among many parties, and makes access to extremely large-scale storage and processing systems feasible for most researchers.

In a remote sensing context, observational metadata have traditionally included only

1. Spacecraft state information (such as orientation, pointing, time, temperature),
2. Detector state information (such as exposure time and operating mode), and
3. Instrument / camera definitions (such as camera geometry models, filter labels, or hardware and interface specifications).

But metadata can also refer to derivational data like algorithms, modeled detector behavior, calibration products, and software. All of these provide crucial context for both raw and derived data. What's more, raw observational data provides context for derived data and derived data provides context for raw, so each is metadata for the other. Our current methods of storing and retrieving such data often "break" such relations and make it impossible to determine the true and complete context of every datum. **In other words, all data is metadata given context -- and that context can be defined quite naturally and efficiently by relational mappings in databases.**

Recording the relationships between raw and derived data can make all operations on raw data lossless and reversible. To give one very important example, to use remote sensing data within existing tools (like ArcGIS [10]), the first step is almost always to get it into some kind of array -- which is of course a type of data abstraction. This step, simply formatting the data, often accounts for a large fraction of the total research effort. It also often requires at least one rasterization or interpolation step (like mapping elongated "pushbroom" pixels onto a regular grid, co-registering data, or creating a Digital Elevation Model). Rasterization and interpolation are often performed at intermediate processing steps to reformat data as an array, even if end-users are not specifically interested in using the derived data from those intermediate processing steps. Reformatting carries computational overhead and information loss.

So to maximize the efficiency and downstream usability of data, *rasterization and interpolation should be performed as infrequently and as late in data processing as possible*. A database that retains complete metadata for every data abstraction makes all rasterization and interpolation lossless if all derived data contains metadata about how and from where its values arose. Moreover, if the database retains complete metadata for *all of the individual components of every observation and abstraction*, it becomes relatively easy to perform many common secondary operations on raw data, such as calibration and map projection, without having to rasterize the data at all.

Thus, we propose Aerobrake, a cloud-based, integrated storage and processing system for remote sensing data. By moving storage and processing of remote sensing data into the cloud and improving its overall intercompatibility, Aerobrake will make that data more *usable*: reducing difficulty and duplication of work will improve the return on research hours spent. Aerobrake will also make this data

much more *useful*: it will increase the total amount of science that researchers can produce from a given family of datasets. Furthermore, by serving as a centralized, integrated storage and processing system, Aerobrake will eliminate the need for each researcher to have his or her own large local system for downloading, storing and processing large amounts of data. This will reduce the upfront costs of remote sensing projects and increase the total return on research dollars spent.

The innovation of Aerobrake rests on the insight that raw remote sensing spacecraft data can be represented within a database by relational mappings and state descriptions for the smallest meaningful units of data rather than in monolithic raster-based file formats. For example, the Aerobrake schema would view a 1-megapixel camera as one million single pixel cameras with well-defined spatial relationships to each other. The treatment of each pixel individually along all stages of processing becomes important in, for example, the case of a pushbroom camera where the downtrack footprint and spacing of the pixels may change over the course of an observation, or in the case of LIDAR or RADAR measurements where the "pixels" not only are neither square nor gridded, but might actually have shapes and geometry defined most precisely in terms of *probabilities*. While this seems obvious when stated bluntly, non-raster treatments of remote sensing data are almost unheard of in modern remote sensing methodologies, and it is only the recent advent of high performance computing and massive low cost storage that make it practical to abandon the efficient array-based logic of raster data.

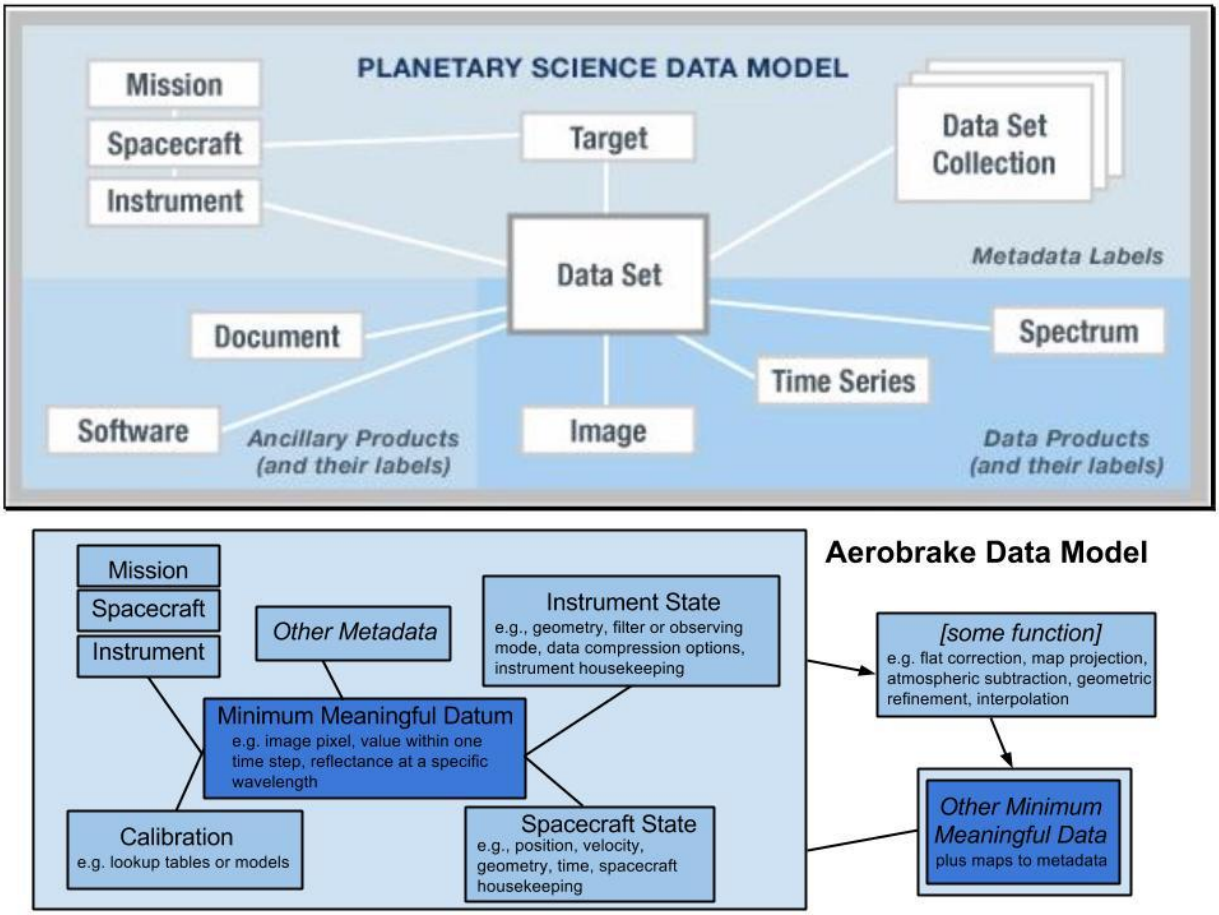

Figure1. Top: A representative model for the current paradigm of remote sensing data management, the Planetary Data System architecture [11]. Data and metadata are separated both physically and philosophically. Bottom: The Aerobrake data model, which establishes a direct link between data and metadata.

As a concrete example, we will explain how Aerobrake would approach one common form of remote sensing data: a CCD image. The smallest meaningful unit of a CCD image is a sensor pixel value, usually described as a Data Number (DN), which is a scale factor of the number of photons in the CCD well. Aerobrake would treat each well within the CCD as its own single-pixel detector. The state information of each of these pixels would be identical to the state information of the original CCD image as a whole (generally including time, spacecraft position and orientation, camera model, temperature, and operation mode) with a single additional value that defined the pixel extent and its relative position on the detector. The DN would be stored in the Aerobrake database as a single entry, relationally linked to its metadata -- in this case, its state information for the CCD image as a whole and a set of generic information about the instrument, such as focal length, sensitivity, and bandpass. It's simple to see that this representational data would be similar in total memory usage as the (uncompressed) raw data, and also that it would be possible to reconstruct the raw image from the individual pixel entries.

So no information is lost by storing the data in this way, but it makes the data much more *flexible.* Because Aerobrake instantiates data with rich context at the finest possible level, users will be able to define complex cuts that return only the specific data of relevance to their project or procedure. Similarly, it will be easy for users to map new relationships between all classes of data. Defining the data in terms of its smallest units homogenizes it and thus solves (at least in this narrow domain) a longstanding problem of data management -- that of incompatible formats. We can treat all datums as equal and only deal with their differences -- type, geometry, time, bandpass -- at the specific levels of abstraction that it is absolutely necessary to do so. We also automatically gain the ability to parallelize many data processing tasks at, e.g., the sensor-pixel level rather than at the *frame*, *scene*, or *observation* levels.

The ability of Aerobrake to generate new metadata relationalities in the act of data processing will also make many common secondary operations on remote sensing data easier and more efficient. For instance, consider the common case of projecting an image onto a Digital Elevation Model (DEM). This process always involves application-specific tradeoffs and decisions. The image, which is defined according to the camera projection, will not share the same "sample points" as the DEM, which is often defined as a regular grid on the surface. Such projections thus often require interpolation, and the appropriate interpolation algorithm differs between applications. Moreover, different applications prefer entirely different types of projection (conic, cylindrical, etc). Some methods of projection and interpolation are better at preserving volume information, some at preserving angular relationships between elements, and so on. In the projection process, a great deal of information about the origin and other characteristics is almost always lost.

However, if an Aerobrake user wishes to project an image stored in the Aerobrake database onto a DEM, they will not need to do so by immediately creating a lossy raster image. Since Aerobrake will define pixels independently and associate them with camera state (and thus camera pointing / projection information), a DEM projection from the image in the Aerobrake database can simply be defined as a *new set of entries*. Pixels in this new set will likely have irregular (and perhaps even multidimensional) shapes defined by metadata. This set will also have its own metadata related to the projection method, perhaps including a normal vector or a center of mass coordinate. And *each projected pixel will also retain a mapping to its source(s) in the raw data.*

If the user requires an interpolated projection, they can create one from this set of projected pixels. This is one of the major advantages of Aerobrake -- **because the system will allow data rasterization to be delayed until the final step in image processing, this computationally expensive and lossy step will only be done once.** Furthermore, since the interpolation is being performed within Aerobrake, it can *also* be defined as a new set of pixel entries with their own relational context. The set is lossless because each entry in the output carries information about how its value was obtained -- that is, which parts of the DEM-projected pixel set were used to obtain its value and how. At this point, the user

would have an interpolated, map-projected (and optionally orthonormalized) raster image in which each pixel could be mapped directly back to the original observed value *and* all associated state and transformation metadata.

Aerobrake will also simplify the synthesis of multiple remote sensing spacecraft data sets. Combining observations from multiple instruments is a common research practice that can enormously increase the science available from past missions by leveraging synergies between data sets. This work is often time-consuming, difficult, and imprecise. But Aerobrake will be able to ameliorate these problems. For example, assume a researcher wants to combine the projected CCD data mentioned above with data from a single-row, pushbroom-style multispectral imager like Landsat 8's Operational Land Imager (OLI) [12]. Such imagers maintain a fixed pointing while the spacecraft moves; they scan ("pushbroom") an area on the surface over a single exposure, and each element in the detector row may simultaneously observe in multiple wavelengths. The Aerobrake database would treat an observation from this pushbroom imager identically to the CCD observation. It would store observational data from each element of its detector row in separate entries. These entries would include multiple values associated with specific wavelengths and would be associated with relative position values, instrument definitions, and instrument and spacecraft state information. The researcher could then import the projection mappings from the projected CCD data and use them to perform the same projection of the hyperspectral imager data onto the same DEM -- and, should they choose, the same interpolation as well. By defining observations from both classes of instruments according to their smallest meaningful units, we no longer force researchers to generate heavily-derived, lossy data products merely to create a common format.

Part 3: Technical Objectives

Top Level Objectives

As noted in the solicitation, "[t]he purpose of Phase I is to determine the scientific, technical, commercial merit and feasibility of the proposed innovation, and the quality of the SBC's performance." To achieve this, we will focus on developing key internal technologies of Aerobrake and testing those in both a performance / cost context (in terms of hardware and overhead requirements) and a usability / utility context (with the help of our scientist consultants). The technical objectives of the Phase I Aerobrake project are to:

1. Test the scalability and extensibility of operations when the data is represented in the proposed schema,
2. Develop and test algorithms and strategies for tracking and establishing new relationalities as data is processed or manipulated,
3. Benchmark the performance for sample classes of such manipulations, and
4. Assess performance and capabilities in the context of user needs.

Representation

Cloud storage and processing are maturing but well explored technologies. We do not propose to make significant advancements in the field of cloud computing in general, but we do propose to advance the state of the art for use of spacecraft remote sensing data within a cloud computing environment. We have outlined how observational remote sensing data will be represented within Aerobrake; that is, as its smallest meaningful unit (a sensor-pixel DN or equivalent) given context by a large number of relational mappings to other data and metadata. We are confident that we can represent data in this way within a database, and have a strong intuition for what consequences it will have for data usability, versatility and performance (described above). However, the only way to know is to produce a prototype implementation that can be analyzed for those outcomes.

The initial representation of the raw data within Aerobrake will be very nearly equivalent in total

size to the uncompressed representation of the same data in any other format. While the use of data in the Aerobrake format is not conceptually difficult, it will amount in practice to searches on a massive many-to-many relational database, which can be quite complicated or slow and are heavily dependent on host system architecture. We will need to profile these operations across multiple combinations of database servers, frameworks, and application logics. It's likely that different operations will benefit from different optimizations or architectures. This will be important for informing our scope in Phase II.

It is also our intention that the result of each additional manipulation or step of processing -- for example, every calibration or map projection -- will likewise be stored permanently in the cloud along with its relational mappings back to the source data. There are two large engineering challenges underlying this capability. The first is *how* those metadata relationships can be defined automatically for essentially arbitrary manipulations -- how they are handled both algorithmically in code and organizationally within the database schema. Second, the proactive creation of new relations could easily result in an explosive increase in storage needs, and will certainly increase the complexity of the database. Concessions will need to be made in terms of what is stored. Determining specific rules for additional metadata generation and storage will be a component of our Phase I investigation. In particular, we will explore the tradeoff between time, complexity, and real costs of storing data permanently vs. recreating it as needed. We may find, for instance, that only the most computationally intensive outputs should be permanently stored, because the overhead associated with dynamically generating less computationally intensive outputs is much less than the overhead associated with simply storing and retrieving them.

Performance profile studies will allow us to codify specific rules for storage vs. on-the-fly processing as a function of the system on which Aerobrake is deployed. Future implementations of Aerobrake will be able to make automated decisions on whether to add the results of a given computation to the database *while the operation is taking place*, in a process that is transparent to end users.

Processing

As stated in the solicitation sub-topic text, adapting existing data processing and analysis tools for a cloud environment could represent a substantial effort. We are not proposing to solve this problem, although some rasterized outputs from processes within Aerobrake could certainly be translated into appropriate formats and fed into existing tools. We will, however, build analogous algorithms to very common operations on remote sensing data (e.g., detector calibration, map projection and orthorectification, multi-band or multi-sensor spectral reduction) in order to both explore and define the mechanisms for processing within Aerobrake and to make informative benchmark tests for parallel processing time-to-completion and processor and network load (all important factors in the *cost* to operate any eventual production system).

Costs of cloud computing scale as a function of both total static data volume and processor hours. It is standard practice in cloud computing applications to benchmark performance and estimate cost based upon representative samples of data and processing, and optimize operations against cost when possible. We will perform this analysis on the Aerobrake prototype implementation using existing planetary data sets.

User Experience

Our user experience goal is to make data quickly useful and available to researchers by providing them with in-language modules or Application Programmer Interfaces (API) that interpret in-language commands as queries on the cloud databases, returning in-language objects that are immediately usable by the researchers. We imagine that in its final implementation, most *builds* of Aerobrake will not be for the exclusive use of individual researchers but, to distribute costs, shared among large research groups, departments, or even whole fields. We would consider Aerobrake a phenomenal success if NASA or the

USGS eventually made public-facing Aerobrake servers available to the broad research community on their own systems (e.g. the NEX supercomputer [13]).

In heavily-trafficked environments of this type, read and write permissions, along with content curation, will become extremely important questions. These permissions and security issues should be explored early in the design rather than hastily solved after they become serious problems. At the very least, it would be unwise to allow anyone to modify the Aerobrake database at will. This could lead to rapidly-increasing data volume and complexity and a large fraction of the database consumed by "junk data" resulting from failed or nonsense investigations. We need to investigate and establish strategies for mitigating the problem while still allowing researchers to creatively draw on the power of cloud computing. One possibility is that individual researchers are given some usefully large amount of personal, private storage on the cloud system (perhaps 10 Tb) in which to test and develop MapReduce [14] and other data processing algorithms.In any case, the identification or development of a viable method to allow individual researchers substantial flexibility in utilizing the cloud environment while maintaining the integrity of the platform is important for establishing the feasibility of Aerobrake.

We expect that it will be easier to implement API commands that simply query the database compared to commands that modify the database. While the former may only require simple database search functions, the latter will require support for frameworks (like MapReduce). Determination of the level of effort required to support all desired functions is an important step in assessing feasibility for Phase II. The study of how researchers might interact with Aerobrake will take place almost automatically in the course of Phase I because *we* will need to interact with the system, and our solutions and experiences will serve as guidance when developing the full API in Phase II.

Part 4: Work Plan

Deliverable

In Phase I, we will create a series of rapid prototypes, simulations, and models and conduct performance and usability tests that address the outstanding technical questions noted above. It is not our intention that the rapid prototypes will, together or independently, compose any kind of fully featured prototype to serve as a deliverable to NASA; we expect to create a fully featured demonstration during Phase II. But the source code, procedures, tests, and outcomes of work of Phase I will all be retained and recorded for review upon request. The final contract deliverable will be a report that includes

1. An outline of the work performed;
2. The test results;
3. A revised assessment of the scientific, technical, and commercial merit and feasibility;
4. Its relevance and significance to NASA needs in light of those tests;
5. And a revised plan for adapting it into a product for NASA or other customers.

Software

We will use Savransky's server at Cornell University as the test and development platform in Phase I. This hardware consists of a 64-core server with 256 Gb of RAM and substantial hard disk space. It hosts multiple rapid prototyping software services for parallelization, including iPython Parallel [15] and a MATLAB Distributed Computing Server [16], and can also simulate large scale Hadoop clusters via local virtualization [17]. Savransky has committed to managing user access and providing support for team members.

Although we will eventually implement a multi-language API, we will prototype in Python. Not all specific features of the Python prototypes (such as object mutability and dynamic typing) will be adaptable to other languages, but broad principles of the software design will be. Prototype software that we will develop include, but are not necessarily limited to, tools and utilities that:

- Ingest raster-based image formats (e.g. PDS3 images) into the Aerobrake database;
- Make simple queries (e.g. extract the upper left quadrant of a CCD image);
- Make complex queries (e.g. find the uncalibrated sensor-pixel values of data which, when projected onto a geoid model, fall within specified latitude and longitude ranges);
- Perform a geometric translation on every pixel of an observation in parallel;
- Generate relational mappings between one input and one output of some procedure;
- Generate *complex* relational mappings between one input and many outputs or many inputs and one output of some procedure; and
- Profile and benchmark database operations (i.e. test scripts).

To be clear, there is no provision for a graphical user interface (GUI) in the Aerobrake design; interactions with Aerobrake will be through either command-line utilities or an API. Command line tools and APIs are still user interfaces, though, and their good design is important for encouraging utility and usability. While we have some initial ideas about how to best mediate user interactions between the database and the command line -- mimicking the indexing and class-method behavior of the popular Python library *pandas* [18] would be a good start -- we will develop our final interface design through extensive user testing.

We will also develop standard definitions for common categories of data. CCDs and similar detectors arranged in arrays are very important and will be easy to implement. They can be represented as we described above -- elements with intensity, position values, and links to state- and instrument-related metadata. Data from other classes of instruments, like laser altimeters or radar, will require modified or unique definitions. We will also begin to develop and maintain a manually-populated dictionary of metadata categories (which might eventually be heuristically definable on arbitrary data sets), including tuple length ranges, data type information, and links to commonly related metadata categories. Aerobrake will not necessarily be limited to storing observational data as such, and we will begin to implement storage for some common categories of functions, maps, and models -- projections and calibrations, for instance. Most of these may be easily storable as perfect hash functions, arrays representing systems of linear equations, or other common data structures.

Testing

While the Aerobrake system will be designed for eventual use with both planetary and terrestrial remote sensing data, our test data in Phase I will primarily be planetary because they are simpler in many ways (e.g. no clouds, very minor surface change over time), but share a fundamental set of technologies with terrestrial remote sensing. In addition, our team includes several members with significant expertise on these data sets, so we will not need to spend time acquainting ourselves with unfamiliar data products. In particular, we plan to use data from the Lunar Orbiter Laser Altimeter (LOLA) [19] and Diviner Lunar Radiometer Experiment (DLRE, or just "Diviner") [20]---instruments on the Lunar Reconnaissance Orbiter (LRO) [21].

Laser altimetry is an interesting use case because the "pixels" in question are not necessarily square or regularly spaced, but the functional form of the data is almost always interpolated and gridded. Diviner, likewise, is a good test case because, while the data volume is quite large, the raw data is stored in simple and well-documented ASCII tables, and the instrument itself contains a number of common but complexity-adding attributes -- it is hyperspectral, pushbroom-style with multiple boresights. Additionally, the Experiment Data Record (EDR, i.e. raw) and Reduced Data Record (RDR, i.e calibrated) for Diviner are co-aligned and losslessly derived, which gives us the opportunity to represent relational mappings between raw and derived data without the need to use, develop, or port some version of the actual calibration software. Diviner and LOLA data are frequently used in concert, so we will also be able to

explore a common case of synthesized data analysis. We will also explore paths for representing continuous data within the database, with the specific case of a Lunar geoid model. We may experiment with other data sets as time and need dictates. We may also generate and use fully synthetic test data.

We will experiment with projection and calibration operations using parallelization tools such as MapReduce. MapReduce [14] is a common programming model for distributed, highly parallelized processing of large datasets that first gained prominence as the infrastructure for Google's World Wide Web indexing system. Its techniques are predictably scalable using readily-available methods so we will be able to estimate performance requirements as a function of database size and complexity on representative fractions of the total data set. Though we will not make focused attempts at database optimizations in Phase I, we will try to *characterize potential problems*, identify potential performance bottlenecks and "sweet spots." Blandford will provide assistance and expertise in database design, optimization, and best practice.

We will also investigate means to track relationships between data across operations and manipulations. We must explore both how this will result in modifications to the database and how it will be handled algorithmically. We will start by working with simple mathematical operations (addition, subtraction, division) and eventually test toy problems of much more complex operations (like map projections, simple model fitting, or pixel unmixing).

Cost and Utility

Our initial prototype will allow us to test and/or model several aspects of Aerobrake's use costs and usability. Notably, we will be able to predict data I/O loads for common sample queries and test some methods for managing data permissions. Of course, it is not only important that candidate features are implementable, but also that they are *needed* -- that they are useful and usable for the conduct of actual scientific research. As such, our two scientific consultants -- Thenkabail and Horgan -- will review these prototypes and provide us with feedback from the perspective of actual target end-users.

Schedule and Management

The PI will fill the Project Management role and serve as the lead software developer on the project. St. Clair will be the second software developer. The PI (Million) is primarily employed by Million Concepts LLC, as required by the solicitation, and St. Clair will be an employee during the time of performance. Tasks will be split between the developers on an as-needed basis, and they will move between tasks as needed. In addition to making his computer systems available for this project, Savransky has particular expertise in the design of efficient numerical algorithms and will provide guidance on this topic, including limited amounts of software development. Subcontractor Blandford will provide assistance and expertise related to database and API design which will include programming and help with database setups and benchmarking. Consultants Horgan and Thenkabail will maintain email contact with the team throughout the project and provide feedback, advice, and guidance on prototypes and results to steer the final design towards maximum usefulness in their actual research workflows.

The PI and developer St. Clair will make two trips to Cornell University to meet with Savransky. The first trip will occur in the first month of the project and serve as a planning and organizational meeting as well as an opportunity for Million and St. Clair to familiarize themselves more closely with the hardware and plan the work to follow. The second trip will take place in the second to last or last month of the project (the 5th or 6th month) and serve as a final design and project review for Phase I. Both visits will feature "all hands" teleconferences with the other members of the project group. For planning purposes, these meetings have been labelled "Preliminary Design Review" and "Final Design Review."

The project term is six months. While the work is expected to proceed iteratively between successive prototypes and tests, there are specific technological capabilities that we want to be able to

demonstrated over the course of the project. The following timeline of effort makes an estimate of when in the work period we expect to be able to demonstrate each of these capabilities relative to others.

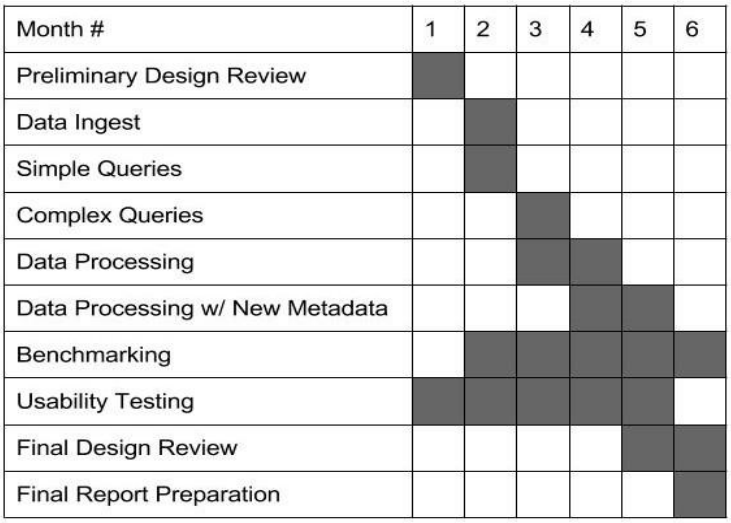

Figure 2: Project timeline by task.

The total level of effort, as budgeted, is ~1045 person-hours. The following figure estimates the breakdown of these hours per broad sub-task. Note that some of the categories align with specific team member roles, and some include multiple team members.

Task	**Hours**	**People**
User Feedback / Testing	80	Thenkabail, Horgan
Prototype Development	465	Million, St. Clair, Savransky, Blandford
Benchmarking	200	Million, St. Clair, Savransky, Blandford
Server Management	100	Million, Savransky, Blandford
Project Management	100	Million
Documentation	100	Million, Savransky
Total	**1045**	

Figure 3: The estimated Level of Effort (LoE) required per task.

The consultants and contractors will work remotely from the PI at their respective home offices and institutions. The PI has experience successfully managing remote, small-team projects of this kind and, in particular, has successfully worked on remote projects with St. Clair, Blandford, and Savransky in the past. The fact that the Phase I personnel are not co-located is not expected to be a problem. The PI will coordinate and supervise work by frequent contact with team members through email, instant messaging, phone, and video conferences. Development and documentation will also be facilitated by online collaboration and version control software (e.g. Google Docs, Dropbox and Github).

Part 5: Related R/R&D

By the Offeror

The PI, Chase Million, conceived, developed, and is in an ongoing contract with the Mikulski Archive at Space Telescope (MAST) at Space Telescope Science Institute (STScI) to create gPhoton, a large database and associated toolkit for the archival of photon level data from the Galaxy Evolution Explorer (GALEX) mission [22]. GALEX was a NASA ultraviolet all-sky survey mission which ended in 2013 [23]. The ultraviolet detectors on the GALEX spacecraft were non-integrating micro-channel plates that recorded individual photon events with a time resolution of five thousandths of a second. [24] Due to constraints on budget, storage, and time, the GALEX data was only released by the mission team as integrated images except by special request, making it difficult if not impossible to mine the GALEX data for short time domain variability. Mr. Million designed and is in the final stages of developing a standalone calibration pipeline and ~100 Tb database for creating and storing GALEX photon-level data. To facilitate use of this database, the PI has also created a set of Python based tools which query the database and generate calibrated light curves, images, and movies directly from the photon-level data on user request. While gPhoton is a wholly distinct, mission-specific project that does not draw on "cloud" technology, it shares many philosophical fundamentals with Aerobrake. It attempts to maximize the flexibility and total utility of a data set by describing it in terms of the smallest meaningful units, individual photon detections. It offloads some large storage and processing tasks to a remote server and database and provides end-user tools for utilizing that database. The PI will draw upon his experience in the development of gPhoton to inform the development of Aerobrake.

By Others

To our knowledge, Aerobrake is not competing directly with efforts by any other groups. Specifically, we're not aware of any enterprise-level Geospatial Information System (GIS) software that *does not* implicitly or explicitly demand a rasterized data schema. However, there are many, many technology solutions addressing similar broad problems as Aerobrake in both research data management and remote sensing.

Databases

There are more database technologies than we could ever hope to list. The most popular is undoubtedly SQL [25]. In fact, most other popular database systems are extensions of or use major concepts from SQL. The following data storage systems are of particular interest in relation to Aerobrake:

- *Rasdaman* [26] is an open source, raster-based database management system (DBMS). It is optimized for multi-dimensional array-based operations. It has *raster* right in the name ("rasda").
- The *Australian Geosciences Data Cube* (AGDC) [27] is an attempt to generate a common and useful format for multi-dimensional remote sensing data with tremendous support in recent years from the Australian government. Data within the cube, however, is required to be coregistered and orthonormalized, that is to say rasterized and interpolated.
- *SciDB* by Paradigm4 [28] is another open source, array-based DBMS for extremely large data sets in the cloud. While it does specifically optimize array-based operations, its general performance enhancements, as advertised, make it extremely promising. It will be one of the systems that we investigate in Phase I for use with Aerobrake.
- *Object Oriented Data Technology* (OODT) is an open-source data management system framework developed by JPL and now managed by the Apache Software Foundation [29]. It is widely used in planetary science, terrestrial remote sensing, and other disciplines for the discovery and query of distributed data resources.

Processing

- ArcGIS [10] by ESRI is undoubtedly the most popular single Geospatial Information System software suite. Another GIS of note is ERDAS IMAGINE [30]. But there are many software suites, but open and closed source, that mimic these capabilities.
- The Integrated Software for Imagers and Spectrometers (ISIS) [31] by the USGS Astrogeology Science Center is probably the most widely used calibration and data processing software suite in planetary science. A related, recent project, Projection on the Web (POW) [32], moves some key functions of ISIS onto a remote server so that researchers are only required to download the final, derived products of interest. There was also a separate trial project that runs ISIS3 in the Amazon Cloud [33].
- The Geospatial Data Abstraction Library (GDAL) [34] is an open source library for translating between various raster geospatial data formats.
- The Earth Science Data Systems section at the Jet Propulsion Laboratory has developed an Enterprise Data Management (EDM) Service for "data archiving, database hosting, application hosting, middleware, and metadata management." [35]
- The NASA Earth Exchange (NEX) at NASA Ames "combines state-of-the-art supercomputing, Earth system modeling, remote sensing data from NASA and other agencies, and a scientific social networking platform to deliver a complete work environment." [36,37]

Data Search and Retrieval

- Within planetary science, notable graphical data browsers include JMARS [38], Google Mars, PILOT [39], and PDS ODE [40]. Many instrument and mission teams have their own toolkits for locating and browsing data of interest.
- For terrestrial data, a number of graphical search tools exist including Landsat Look [41], USGS Global Visualization Viewer [42,43], EarthExplorer [44], and Google Earth Engine [45,46].
- A recent Python utility called *landsat-util* [47] enables search and processing tasks for Landsat data from a command line.

Part 6: Key Personnel and Bibliography of Directly Related Work

Chase Million (Principal Investigator) is the CEO and owner of Million Concepts LLC, a business which seeks to improve the quality and availability of software engineering in research science. He has a Bachelor of Arts in Physics from Cornell University (2007) and more than ten years of experience as a researcher and software developer working with spacecraft data in planetary science and astronomy. Prior to starting Million Concepts, Mr. Million was the lead software engineer for the ground calibration pipeline software for the GALEX mission at the California Institute of Technology. Prior to that, he was a member of the Mars Exploration Rover Pancam team at Cornell University. He also served as the team lead for the design and construction of an automated ground station for command and control of a CubeSat class satellite developed by Cornell University. He has direct experience conducting or supporting research with a number of datasets from NASA planetary missions which include the Mars Exploration Rovers (MER), Mars Reconnaissance Orbiter (MRO), Odyssey (ODY), and Mars Global Surveyor (MGS), as well as general expertise in spacecraft engineering, data management and calibration, and research support. He frequently fills a project management role that includes the responsibility of ensuring that work proceeds according to contract agreements.

Mr. Million has experience serving as the primary point of contact with contracting organizations and success leading small teams to ensure that projects proceed according to contract terms. As the PI, he will serve as project manager and be involved in or oversee all aspects of Phase I work. He will work from his home office in State College, PA. His total time commitment over Phase I will be approximately

40% (0.4 FTE) over six months, or approximately 64 hours per month for 384 hours (2.4 person-months) over Phase I. As owner and CEO of Million Concepts LLC, Mr. Million is primarily employed by the SBC and therefore eligible to be the PI of this SBIR proposal. Million Concepts LLC has an ongoing contract with Space Telescope Science Institute -- to develop a large database and associated tools to archive and provide access to the GALEX photon level data and described more thoroughly in Part 5 -- for 0.4 FTE (or 4.8 person-months) of his time through the end of 2015. He is a party on an ongoing NASA PGG grant which commits 0.25 months of his time yearly through 2018. He is the PI of a submitted NASA PDART proposal which, if funded, would commit him for 1.3 months in the first year and 0.65 months in the second and third years. He is also a collaborator on a submitted NASA LDAP proposal that, if funded, would commit him for 0.325 months in the first year and 1.3 months in the second and third years.

Dr. Michael St. Clair (Developer) is the Chief Technology Officer of Million Concepts LLC. He has a Bachelor of Science in Mathematics and a Bachelor of Arts in Theater from Case Western Reserve University (2005), as well as a Ph.D. from Stanford in Theater and Performance Studies (2013) for his research into the social and technological aspects of play and games. Dr. St. Clair is a technologist who has performed research and programming tasks for a wide variety of design, engineering, and scientific applications ranging from quantum computing to musical instrumentation. If this Phase I project is funded, St. Clair will work from his home office in Mountain View, CA but be employed by Million Concepts LLC during the project. His total time commitment over Phase I will be approximately 40% (0.40 FTE) over six months, or approximately 64 hours per month for 384 hours (2.4 person-months).

Part 7: Relationship with Phase II or Future R/R&D

The technical objectives and work plan for Phase I are intended to provide us with critical information about how to approach the design, development, and delivery of a complete Aerobrake system in Phase II. In addition to providing us with a foundation with which to assess technical and commercial feasibility of the Aerobrake concept generally, the Phase I results will give us information necessary to properly plan and scope the Phase II effort. Prior to and early in Phase II, we will be able to generate Concepts of Operations (ConOps) and early Software Interface Specification (SIS) documents which will reduce the risk and uncertainty of full scale development and also, as tools to communicate system features to potential stakeholders, allow us to seek early partners in NASA and elsewhere for integration and commercialization of the technology.

Part 8: Facilities/Equipment

Million Concepts LLC is operated from an office in State College, PA. Several computer systems are available with multiple operating systems and high-bandwidth internet. Savransky will make his research server at Cornell University available for prototyping and testing purposes during Phase I. This is a 64 core server with 256 Gb of RAM and an expandable RAID. It runs multiple rapid prototyping software services for parallelization, including iPython Parallel **[15]** and a MATLAB Distributed Computing Server **[16]** and can also simulate large scale hadoop clusters via local virtualization.

Part 9: Subcontracts and Consultants

Dr. Dmitry Savransky (subcontractor) is an Assistant Professor in the Sibley School of Mechanical and Aerospace Engineering at Cornell University. He received his Ph.D. in 2011 from Princeton University in Mechanical and Aerospace Engineering with work on the subjects of dynamics and control and machine learning, particularly as applied to the topic of space telescopes designed to detect extrasolar planets, followed by a two year postdoctoral position at Lawrence Livermore National Laboratory where he assisted in the integration, testing, and commissioning of the Gemini Planet Imager. He also previously

served in research and support positions at Cornell and JPL for Mars Odyssey and the Mars Exploration Rovers. He is an experienced developer of high performance image processing software and algorithms, and will advise and assist in the development of prototype software, algorithms, and tests as well as provide the Aerobrake team with access to his researcher server for testing and development purposes. He has committed 0.45 person-months of time.

Mike Blandford (subcontractor) has a Masters in Computer Science from Indiana University, Bloomington (2007) and 8 years of experience in commercial software development. He has worked on a wide range of software development projects including databases, server side algorithms, and User Interface (UI) and API development. Mr. Blandford will provide consultation and expertise related to database and API development and testing which will include production of code. He will commit 150 hours over the six month project.

Dr. Briony Horgan (consultant) is an Assistant Professor of Earth, Atmospheric, and Planetary Sciences at Purdue University. She obtained her PhD in Astronomy and Space Sciences from Cornell University in 2010 and was an Exploration Postdoctoral Fellow in the Mars Space Flight Facility at Arizona State University. Her expertise is near- and mid-infrared spectroscopy of planetary surfaces from orbital and landed platforms, and she has lead the creation of local, regional, and global-scale hyperspectral image mosaics of both Mars and the Moon, using data from the Mars Express OMEGA, Mars Reconnaissance Orbiter Compact Reconnaissance Imaging Spectrometer for Mars (MRO/CRISM), and Chandrayaan-1 Moon Mineralogy Mapper (M^3) imaging spectrometers. Because of their 3D nature, large size, and the high level of processing required for calibration and analysis, hyperspectral image cubes present unique challenges for data processing, management, and storage that could be successfully addressed by Aerobrake. Dr. Horgan will provide input and feedback on design documents and rapid prototypes during Phase I to ensure that Aerobrake appropriately addresses the needs of working research scientists in her field. Dr. Horgan has committed to 40 hours over the course of the project.

Dr. Prasad Thenkabail (consultant) has more than 25 years of experience as a well-recognized international expert in remote sensing and geographic information systems and their application to agriculture, wetlands, natural resource management, water resources, forests, sustainable development, and environmental studies. He is the editor in chief of the books Remote Sensing of Global Croplands for Food Security and Hyperspectral Remote Sensing of Vegetation. He is a member of the Landsat Science Team (2007-2011) and is on the editorial boards of two remote sensing journals, Remote Sensing of Environment and Journal of Remote Sensing. He led the global irrigated area mapping (GIAM) project and the global mapping of rainfed croplands (GMRCA) project, and has conducted pioneering work in hyperspectral remote sensing. Currently, he is a research geographer at the U.S. Geological Survey (USGS) where his ongoing, NASA funded project Global Food Security-Support Analysis Data at 30m (GFSAD30) seeks to provide high resolution global cropland and water use data derived through multi-sensor remote sensing (e.g. Landsat, MODIS, AVHRR) secondary, and field-plot data. Dr. Thenkabail will provide input and feedback on design documents and rapid prototypes during Phase I to ensure that Aerobrake appropriately addresses the needs of working research scientists in his field. Dr. Thenkabail has committed to 40 hours over the course of the project.

Part 10: Potential Post Applications

The US Government, including NASA and USGS, are primary *creators* and *users* of terrestrial and planetary remote sensing data for applications ranging from Mars exploration to irrigation monitoring. The next generation of Earth Observation (EO) satellites such as NASA's HyspIRI are going to gather "massively big hyperspectral data" (MBHD) that will revolutionize the richness and accuracy of information extraction and knowledge generation about our planet. However, MBHD also carry equally great challenges of turning the mountains of data into meaningful and actionable information and knowledge. These challenges include heavy computing, smart algorithms, data mining, establishing target Hyperspectral Narrow Bands (HNB) and Hyperspectral Vegetation Indices (HVI), establishing new methods to improve classification accuracies, modeling approaches, and techniques for quantifying biophysical and chemical quantities. The novel approaches developed by Aerobrake will ideally suit terrestrial hyperspectral data processing.

For example, understanding, modeling, mapping, and monitoring global croplands are critical to assessing global food security. Global croplands account for 70-90% of all human water use in most parts of the world. As the world population increases from 7.2 billion to nearly 10 billion by 2050, maintenance of food security will demand an improved understanding of crop productivity and water productivity. Recent research by our collaborator Thenkabail and others [50] has implied that the improved understanding will come from the use of hyperspectral remote sensing or imaging spectroscopy data. For Thenkabail's current work producing Landsat maps at 30 meter resolution [51], he must use a stepwise approach where 1 kilometer products are processed through the use of in-house code in javascript and python, ERDAS IMAGINE, the NEX supercomputer, and Google Earth Engine. Aerobrake holds the promise of dramatically streamlining this work and work like this.

Problems in planetary science data reduction (the domain of our collaborator Horgan), while perhaps not as critical in terms of the long term health of our planet, mimic the problems in terrestrial remote sensing very closely. Some of the issues are actually worse: whereas the consistent design of the successive Landsat spacecraft has made the nature of the data somewhat *predictable* over the decades, essentially every planetary science instrument has new and unique properties, with data formats defined according to the needs of the mission or instrument team (not subsequent science investigation), and software tools for even simple tasks limited to completely non-existent. With Aerobrake, we have the opportunity to build a tool that serves both communities, and increase the efficiency of scientific investigations and return on investment of NASA missions quite broadly.

Terrestrial remote sensing data is also incredibly valuable. The economic benefit of Landsat data in 2011 was estimated to be $2.19B worldwide, not including the contribution of value-added products based upon Landsat imagery [48]. The revenue of ESRI, a leading provider of geospatial products including the incredibly popular ArcGIS, is over $900M [49]. The value of hyperspectral data is expected to be much greater, potentially providing us with means to more precisely predict crop yields, locate natural resources, or better track pollution to name a few possibilities. But unlocking the value of hyperspectral remote sensing will require the development of new tools and new techniques. We believe that Aerobrake will be a key technology for managing the upcoming deluge of massively big hyperspectral data. With Aerobrake in our portfolio, Million Concepts will be positioned as a leader in this emerging market.

While Aerobrake is ultimately a commercial venture of the type that SBIR is intended to support, we also strongly believe that software used for scientific research should be open-source and free whenever possible. Free availability of software and source code is more attractive to be used by researchers and research support engineers because they can be assured of its availability over time. Open source is also an important factor in software sustainability and reproducible research and increasingly a requirement attached to funding or publication. We will therefore to release the fully

documented source code of the core Aerobrake system developed in Phase II under a GPL-compatible, BSD-style license.

Our revenue model for Aerobrake will follow that of open-source software companies like Redhat and Rasdaman. These companies release their tools for free but charge for support, targeted development, and some commercial uses. We further believe that having Aerobrake in the portfolio of Million Concepts will increase our visibility and attractiveness to academic, government, and commercial spacecraft data users as an outside vendor and contractor for custom research and analysis software, the services at the core of our business model. We anticipate future inclusion of Aerobrake in project data management plans from very early in the design process; we also expect to build support over time for many existing data sets. Aerobrake could be a service offered on hardware like the NEX supercomputing system or its successors. Long-term, Aerobrake could form the basis for new NASA data archives that fill an existing gap in services.

Our role as *developers* of the Aerobrake technology will also position us as its most qualified users. In the medium to long term, we would like to grow an internal pool of talented hyperspectral data processors and remote sensing technicians to generate value added data products for resale (e.g. heavily derived products targeted to yields of specific crops, natural resource abundance or utilization, climate or pollution monitoring, or military intelligence).

Finally, many of the techniques and concepts behind Aerobrake are extensible imaging data generally. That is, the only thing special about orbital data is that the instrument is pointing, roughly, "down." The same data format and retrieval paradigms could apply equally as well to data taken from instruments "on the ground" so long as the camera definition and state are well understood or can be derived. With recent advances in the reconstruction of camera position and orientation from multiple images of the same scene, it's possible that this includes a large fraction of *all* imaging data. We don't know what additional capabilities an Aerobrake-like system would add to the exploration of such data, and such uses are many years out, but possibilities range from instant 3D modeling of art installations derived from photographs in museum catalogs to maps of defoliation patterns derived from large public corpuses of geotagged photographs.

Part 11a: Essentially Equivalent and Duplicate Proposals and Awards

None.

Part 11b: Related Research and Development Proposals and Awards

gPhoton: a project to archive and make available the GALEX photon level data

The PI, Chase Million, developed and is in an ongoing contract by the Mikulski Archive at Space Telescope (MAST) at Space Telescope Science Institute (STScI) to create gPhoton, a large database and associated toolkit for the archival of photon level data from the Galaxy Evolution Explorer (GALEX) mission. The ultraviolet detectors on the GALEX spacecraft were non-integrating micro-channel plates that record individual photon events with a time resolution of five thousandths of a second. Due to constraints on budget, storage, and time, the GALEX data was only released by the mission team as integrated images except by special request, making it difficult if not impossible to mine the GALEX data for short time domain variability. Mr. Million designed and is in the final stages of developing a standalone calibration pipeline and ~100 Tb database for creating and storing GALEX photon-level data. To facilitate use of this database, the PI has also created a set of Python based tools which query the database and generate user-requested, calibrated light curves, images, and movies directly from the photon-level data.

Despite the similarities, the gPhoton project should not be considered equivalent work in whole or part. The gPhoton project is heavily mission specific to the GALEX photon level data in particular and is not directly extensible to any other data set. The gPhoton project also does not explicitly draw upon cloud computing technologies, only traditional SQL databases, and while gPhoton includes the creation of Python language tools for the access and processing data, these are not an API.

Works Cited

[1] Gaddis, L. R., et al. "Summary and Abstracts of the Planetary Data Workshop, June 2012." Working version online at astrogeology.usgs.gov/search/details/Docs/PlanetaryDataWorkshop/PlanetaryDataWorkshop_AbstractVol_Sept2013_Draft/pdf

[2] 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).

[3] Paige, D. A., et al. "The lunar reconnaissance orbiter diviner lunar radiometer experiment." Space Science Reviews 150.1-4 (2010): 125-160.

[4] Chien, Steve, et al. "Onboard instrument processing concepts for the HyspIRI mission." Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International. IEEE, 2010.

[5] 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).

[6] Zuber, M_T, et al. "The Mars Observer laser altimeter investigation." Journal of Geophysical Research: Planets (1991–2012) 97.E5 (1992): 7781-7797.

[7] "What is raster data?" ESRI. http://webhelp.esri.com/arcgisdesktop/9.2/index.cfm?TopicName=What_is_raster_data%3F

[8] a remote sensing textbook

[9] Eismann, Michael Theodore. "Hyperspectral remote sensing." SPIE, 2012.

[10] ArcGIS. ESRI. http://www.arcgis.com

[11] Hughes, J. Steven, et al. "The semantic planetary data system." (2005).

[12] Knight, Edward J., and Geir Kvaran. "Landsat-8 operational land imager design, characterization and performance." Remote Sensing 6.11 (2014): 10286-10305.

[13] Nemani, R., et al. "Collaborative supercomputing for global change science."Eos, Transactions American Geophysical Union 92.13 (2011): 109-110.

[14] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: Simplified Data Processing on Large Clusters." OSDI'04: Sixth Symposium on Operating System Design and Implementation,

[15] iPython Parallel. http://ipython.org/ipython-doc/2/parallel/parallel_intro.html

[16] MDCE. http://www.mathworks.com/help/mdce/index.html

[17] Shvachko, Konstantin, et al. "The hadoop distributed file system." Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on. IEEE, 2010.

[18] McKinney, Wes. "pandas: a python data analysis library." see http://pandas. pydata. org.

[19] Riris, Haris, et al. "The lunar orbiter laser altimeter (LOLA) on NASA's lunar reconnaissance orbiter (LRO) mission." Defense and Security Symposium. International Society for Optics and Photonics, 2007.

[20] Paige, D. A., et al. "The lunar reconnaissance orbiter diviner lunar radiometer experiment." Space Science Reviews 150.1-4 (2010): 125-160.

[21] Chin, Gordon, et al. "Lunar Reconnaissance Orbiter Overview: The Instrument Suite and Mission." Space Science Reviews 129.4 (2007): 391-419.

[22] Fleming, Scott W., et al. "New Kepler Data Products At MAST For Stellar Astrophysics." American Astronomical Society Meeting Abstracts. Vol. 223. 2014.

[23] Martin, D. C., et al. "The Galaxy Evolution Explorer: A Space Ultraviolet Survey Mission." Astrophysical Journal Letters 619.1 (2005).

[24] Siegmund, Oswald, et al. "High performance microchannel plate imaging photon counters for spaceborne sensing." Proc. SPIE 6220, Spaceborne Sensors III, 622004 (May 30, 2006).

[25] Chamberlin, Donald D., and Raymond F. Boyce. "SEQUEL: A structured English query language." Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) workshop on Data description, access and control. ACM, 1974.

[26] Baumann, Peter, et al. "The multidimensional database system RasDaMan."ACM SIGMOD Record. Vol. 27. No. 2. ACM, 1998.
[27] Australian Geoscience Data Cube. http://nci.org.au/virtual-laboratories/earth-observation/australian-geoscience-data-cube/
[28] Stonebraker, Michael, et al. "The architecture of SciDB." Scientific and Statistical Database Management. Springer Berlin Heidelberg, 2011.
[29] Mattmann, Chris A., et al. "Software architecture for large-scale, distributed, data-intensive systems." Software Architecture, 2004. (WICSA 2004). IEEE, 2004.
[30] ERDAS IMAGINE. http://www.hexagongeospatial.com/products/remote-sensing/erdas-imagine/overview
[31] Torson, J. M., and K. J. Becker. "ISIS-A software architecture for processing planetary images." Lunar and Planetary Institute Science Conference Abstracts 28 (1997).
[32] Hare, Trent. "Map Projection on the Web (POW): 'ISIS3 in the Cloud.'' LPSC 44 (March 2013).
[33] Laura, J. (2012). "ISIS3 in the Amazon Cloud." Poster Presentation at the 2nd Annual USGS Planetary Data Users Workshop. June 25-29, 2012.
[34] Warmerdam, Frank. "The geospatial data abstraction library." Open Source Approaches in Spatial Data Handling. Springer Berlin Heidelberg, 2008. 87-104.
[35] Hardman, Sean, et al. "Reusable software services for science data systems."5th International Symposium on Reducing the Cost of Spacecraft Ground Systems and Operations, Pasadena, CA, July. 2003.
[36] https://nex.nasa.gov/nex/
[37] Nemani, R., et al. "Collaborative supercomputing for global change science."Eos, Transactions American Geophysical Union 92.13 (2011): 109-110.
[38] Gorelick, N. S., et al. "JMARS: A multimission data fusion application." Lunar and Planetary Institute Science Conference Abstracts. Vol. 34. 2003.
[39] Wang, J., et al. "Planetary data access through the orbital data explorer from the PDS geosciences node." Lunar and Planetary Institute Science Conference Abstracts. Vol. 40. 2009.
[40] Bailen, M. S. "The PDS Planetary Image LOcator Tool (PILOT)." Summary and Abstracts of the Planetary Data Workshop, 2012.
[41] Landsat Look. USGS. http://landsatlook.usgs.gov/
[42] GloVis. USGS. http://glovis.usgs.gov/
[43] Campbell, James B. "GloVis as a resource for teaching geographic content and concepts." Journal of Geography 106.6 (2008): 239-251.
[44] Earth Explorer. USGS. http://earthexplorer.usgs.gov/
[45] Google Earth Engine. Google. https://earthengine.google.org/
[46] Gorelick, N. "Google Earth Engine." AGU Fall Meeting Abstracts. Vol. 1. 2012.
[47] https://github.com/developmentseed/landsat-util
[48] Miller, Holly M., et al. "The users, uses, and value of Landsat and other moderate-resolution satellite imagery in the United States—Executive report."US Geological Survey Open-File Report 1031 (2011): 42.
[49] Lagorio-Chafkin, C. "How We hit $192 Million in Sales." Inc. (August, 2013) http://www.inc.com/magazine/201307/christine-lagorio/jack-dangermond-how-he-started-esri.html
[50] Thenkabail, Prasad S., et al. "Hyperspectral remote sensing of vegetation and agricultural crops." *PHOTOGRAMMETRIC ENGINEERING AND REMOTE SENSING* 80.8 (2014): 697-709.
[51] Teleguntla, Pardhasaradhi, et al. "Global Cropland Area Database (GCAD) derived from Remote Sensing in Support of Food Security in the Twenty-first Century: Current Achievements and Future Possibilities." Remote Sensing Handbook: Land Resources: Monitoring, Modelling, and Mapping, Volume II, Chapter 7.