The NOAO Data Laboratory: A conceptual overview

Michael J. Fitzpatrick*, Knut Olsen, Frossie Economou, Elizabeth B. Stobie,
T.C. Beers, Mark Dickinson, Patrick Norris, Abi Saha, Robert Seaman,
David R. Silva, Robert A. Swaters, Brian Thomas, Francisco Valdes
National Optical Astronomy Observatory, 950 N Cherry Ave, Tucson, AZ, USA 85719

ABSTRACT

The *NOAO Data Lab* will allow users to efficiently utilize catalogs of billions of objects, augment traditional telescope imaging and spectral data with external archive holdings, publish high level data products of their research, share custom results with collaborators and experiment with analysis toolkits. The goal of the Data Lab is to provide a common framework and workspace for science collaborations and individuals to use and disseminate data from large surveys.

In this paper we describe the motivations behind the NOAO Data Lab and present a conceptual overview of the activities we plan to support. Specific science cases will be used to develop a prototype framework and tools, allowing us to work directly with scientists from survey teams to ensure development will remain focused on scientifically productive tasks. This will additionally develop a pool of both scientific and technical experts who can provide ongoing advice and support for community users as the scope and capabilities of the Data Lab expand.

Keywords: data publication, virtual storage, virtualization, large database, collaborative science, virtual observatory

1 INTRODUCTION

Throughout much of the history of modern astronomy, growth in scientific ambition prompted the field to simply build bigger telescopes. While that tradition does continue, our collective ambition is also increasingly finding its outlet in the drive to construct ever larger and more complex datasets. As a result, the role of the astronomer is changing from a traditional approach of single investigators or small groups collecting, calibrating, and analyzing small sets of data, to one of working in big teams to run surveys that generate large collections of data used by broad swaths of the community. Those users start with pipelined data products, and then spend the bulk of their time writing software to understand the trends and secrets hidden within these large sets of data. This evolution in the culture of astronomy also brings several challenges, such as: 1) How do astronomers efficiently access, explore, and visualize datasets that are too big to fit on desktop computers? 2) How do large teams coordinate their data analysis activity of massive datasets? 3) How do astronomers effectively share large datasets that they have generated themselves with the broader community?

In this paper, we describe the science case and technological approach for an NOAO Data Lab that will be designed to help astronomers meet these challenges for current and future surveys. The goal of the Data Lab is to provide a collaborative computing environment for users, in which they can easily access and visualize large datasets, conduct experiments on subsets of the data, share analysis recipes and results with collaborators, and publish new large datasets for use by the broad community. For the first stages of development, the Data Lab will focus on enabling community scientific discovery with DECam-based surveys, including the Dark Energy Survey (DES). These surveys will deliver catalogs of such large size that a central database with select data services will be needed in order to allow users to find the subset of objects that are scientifically interesting to them. In supporting this science, the Data Lab will play an important role in preparing the broad user community for LSST. In the era of LSST operations, the Data Lab will continue to serve as a proving ground and incubator for LSST science, with the ability to conduct experiments on subsets of the LSST dataset.

*fitz@noao.edu; phone 1 520 318-8387; fax 1 520 318-8360; noao.edu

2 SCIENCE USE CASES

The first stage of Data Lab development will have as its primary goal to enable science with the first public release of the DES catalog, which will be available mid-way through the 5-year DES survey period. In order to build towards this goal, the Data Lab will initially develop around results from two smaller prototype programs with data collected from DECam. In this section, we outline the usage of DES that we aim to support, and describe the way in which the prototype programs anticipate that usage.

2.1 Community Use of DES

The Dark Energy Survey, which is spending 525 nights on the CTIO 4-meter (Blanco) telescope with DECam over five years to survey 5000 sq. deg. of sky around the South Galactic Cap, represents a tremendous scientific opportunity for the NOAO community. It will be an unprecedented deep, wide, multi-band and multi-epoch imaging survey, superseded only by LSST. As written in the document "Description of the Dark Energy Survey for Astronomers"[1], the DES is a tiered survey. The overall goal is to achieve 5-sigma point source depths of (g,r,i,z,Y)=(26.5,26.0,25.3,24.7,23.0) at the end of 5 years, broken into 10 visits per field and filter, with 2 visits each year. The first observing season, however, has focused on 4 patches of a few hundred square degrees observed to full depth, and hence faster cadence. In addition, DES contains a dedicated supernova survey, which will concentrate on 10 DECam fields visited ~100 times in each filter, thus providing a much more extensive cadence for a small area than the main wide-field survey. The final catalog will be of order ~25-40 TB in size. Given the large area of the survey, its depth, and the attention to precision for many of the measurements that the main DES science goals demand, the DES catalog and image products are going to be very useful for a number of other science projects led by the community. Supporting community use of the DES data is the first major goal of the Data Lab.

We have developed a number of potential community use cases of the DES catalog and associated images. These include the detection of substructure in the Galactic halo, which has the potential to answer a number of important questions regarding the accretion history of the Galaxy, the distribution of dark matter in the Galaxy, the presence and distribution of hot gas in the halo, and the history of interaction of the Milky Way's satellite dwarf galaxies. The key techniques are using deep color-magnitude diagrams (CMDs) to identify stellar populations of interest and then mapping these on the sky, as well as tracing structure three dimensionally through specific variable star populations. The dedicated supernova DES data will be a particularly rich resource for variable star populations. An additional science use case is thus to parse and classify all variable events, which involves analyzing the time series themselves as well as associating the DES catalog with external datasets. Finally, the DES catalog will be an excellent source for identifying candidate objects for further detailed study at the pixel level, such as looking for low surface brightness streams signifying accretion events around a well-defined sample of galaxies from the catalog.

Our range of use cases point to a set of basic needs for Data Lab users, including:

1. Database access to the DES catalog
2. Visualization tools for objects in the database, including an image postage stamp cutout service
3. Ability to apply complex filters to the database and to store subsets of the data in user-specific tables
4. Ability to attach custom analysis routines to the database output
5. Ability to correlate the DES catalog with other datasets

We will now describe how our two prototype programs will help develop these capabilities prior to the release of the public DES catalog.

2.2 Survey of the MAgellanic Stellar History (SMASH)

SMASH (Survey of the Magellanic Stellar History) is a 30-night NOAO Survey program (PI D. Nidever, U. Michigan) aiming to use DECam to identify the full extent of Magellanic Cloud stellar populations, derive the star formation histories of the Clouds out to large radius, and detect the debris from their past interactions. By identifying the stellar component of the gaseous Magellanic Stream and Leading Arm, SMASH data will also provide an important probe of

the Galactic halo, in particular its hot gas content. SMASH follows in the vein of the Outer Limits Survey[2], a now completed NOAO Survey program, but with a much larger delivered data set.

The main data products from SMASH will be images in *ugriz* in 154 fields to depths of ~24 mag per filter, from which will be derived a photometric catalog of ~250 million objects, with roughly half of these located in the crowded main bodies of the Magellanic Clouds. Additional data products will include the reduced images themselves, calibration data from standard star measurements, and artificial star tests in the most crowded areas of the survey. The measurements made on the survey data will include astrometric positions, multiband time-stamped photometry, and shape measurements. In all, the SMASH data volume will include ~20 TB of imaging data and ~100 GB of catalog data.

The key technique for SMASH is the same as for the Galactic substructure science case for DES, using deep CMDs to isolate potential Magellanic Cloud stellar populations, and then to map the spatial density distributions of those populations across the sky using a matched filter. Of critical importance is eliminating as many of the contaminating background galaxies as possible. The technique will make use of profile-fit magnitudes in *ugriz* and star-galaxy discrimination from shape measurements and from color distributions. Experiments on the SMASH data have shown that with contamination removal, we can detect Magellanic Cloud populations to very low surface density, equivalent to ~35 mags arcsec^{-2} in surface brightness.

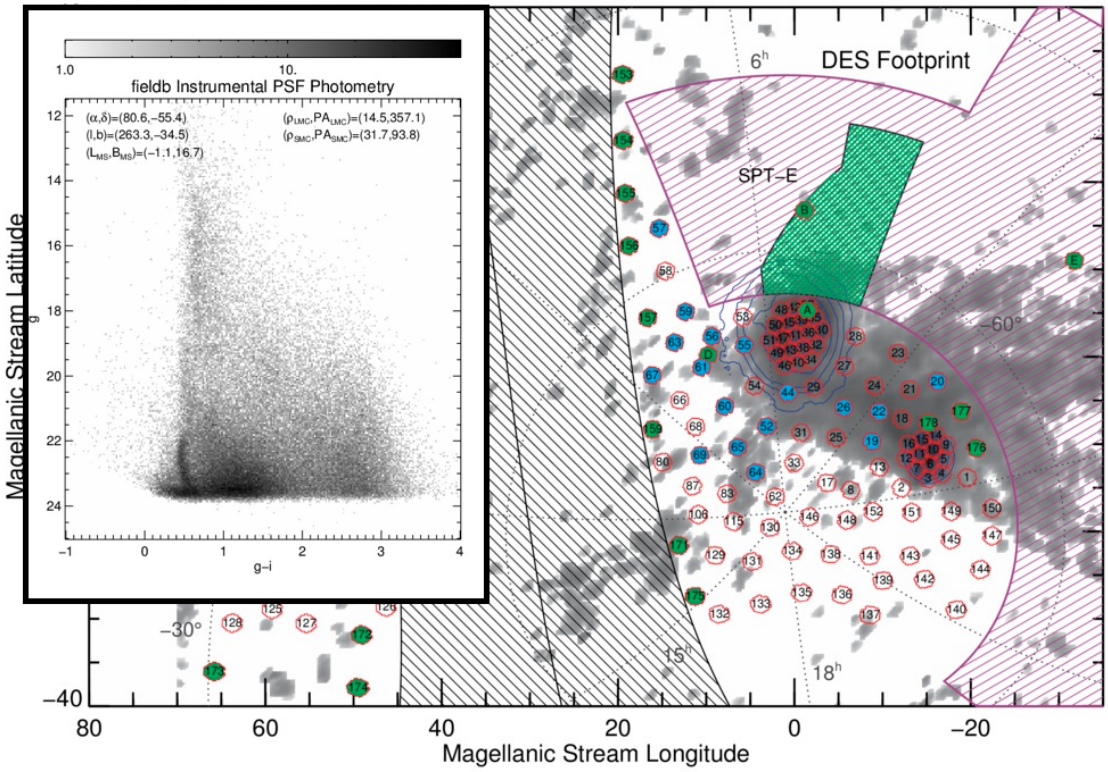

Figure 1. An interactive map allowing visualization of results from SMASH (courtesy of David Nidever). A user clicking on a field marker (here Field B) is presented with an image showing the g-i,g CMD of that field. In this figure, the map demonstrates the presence of an LMC population 14 degrees away from the LMC center. The tool accesses a static JPEG image when a point is clicked; a more advanced implementation in the Data Lab could access the SMASH database itself, and allow the user to apply additional constraints to the data before plotting.

What data analysis services hosted by the NOAO Data Laboratory are most needed for SMASH? The ultimate goal for these services should be to reduce the full catalog of objects to a subset of interesting objects small enough in size to be transferred to a local machine for further processing. The process of reducing the catalog to the most interesting objects will include:

1. Using all of the shape parameters present in the SMASH catalog to establish optimal star-galaxy separation based on morphological parameters
2. Applying color cuts on all of the *ugriz* measurements for additional background galaxy decontamination
3. Creating a working list of likely stellar sources that pass the morphological and color-based criteria
4. From that working list, applying a set of matched filters in color-magnitude space designed to isolate candidate stellar populations
5. Running software to turn the output of those matched filters into surface density maps of isolated stellar populations

One particular innovation of SMASH for sharing results has been the development of an interactive visualization tool for exploring the SMASH CMDs (see Figure 1). If linked to the photometric database and filtering tools themselves, this tool could form the basis of a powerful interactive data exploration capability, giving rapid feedback on the efficacy of the filters applied to the data.

2.3 DECam Survey of the Galactic Bulge

"A Deep Synoptic Study of the Galactic Bulge" (PI A. Saha, NOAO) is a DECam program that recently completed 10.5 nights of observing in five fields in the Galactic Bulge. The main scientific goal of the program is to derive the stellar population mix in the Bulge, which is thought to contain the imprints of the Galaxy's earliest formation history. In order to allow this analysis, however, one needs to account for the strong and variable extinction present in all Bulge fields.

The program hinges on deriving nearly extinction free CMDs of the Bulge fields, from which one can then derive the star-formation histories and look for unique objects, such as extremely metal-poor stars. To do this, the program will use the ~100 RR Lyrae variables that are expected in each field to construct reddening maps. RR Lyrae have well-defined intrinsic colors at minimum light, making them excellent probes of extinction along the line of sight.

The Bulge program data consist of images in five fields in *ugriz*, with ~40 epochs of observation in each field and each band. The data have been processed through the DECam Community Pipeline, and then passed to Saha's custom DoPHOT-based photometric pipeline for deriving photometry. Each field contains ~6 million objects, with measurements of photometry, object shapes, time of observation, etc. tabulated over ~20 columns. With five fields and 40 epochs per field, the complete final catalog will have ~1.2 billion rows in it, resulting in a database ~100 GB in size.

The DECam Bulge project hinges on the ability to identify and classify the variable stars in the catalog, which we expect to comprise ~1% of the total number of objects, or ~300,000 variable stars. The rough workflow will be to:

1. Construct a list of all of the unique objects in the Bulge database
2. Collect the time series of all measurements at every epoch for those objects
3. Calculate statistics, such as the variance in the magnitude measurements of each band compared to the expected variance from the photometric errors, to identify variable sources
4. Provide image postage stamp movies of variable sources, to aid in identifying data quality issues
5. From the list of good candidate variable sources, run analysis routines such as period finders and light curve classifiers to identify the bona fide RR Lyrae
6. Verify the output of these routines interactively

3 DATA LAB COMPONENTS

The Data Lab will be comprised of various components, both physical and logical, that interoperate to provide a complete system for use by astronomers. In this section we describe the data services that function as a starting point for some investigation, the data handling tools and services that can be combined with science-specific code to perform an analysis, and the infrastructure required to make it all work. Wherever possible we plan to adopt and extend existing implementations to meet a specific need, and will rely on standardized interfaces and protocols in developing new tools. The components as described here represent the fully realized vision of the Data Lab, however our implementation plan (see Sec. 5) is designed to deliver a working baseline system by the end of our prototyping phase.

3.1 Data Services

3.1.1 Large Catalog Support

As seen from the DECam-based science use cases, a critical capability needed in the Data Lab is access to large catalogs through databases. The final DES 'coadd object' catalog is expected to be 20 TB in size and contain ~2 billion objects (with the final individual exposure detection catalog being ~100 TB and ~20 billion rows) (Brian Yanny, personal communication, May 29, 2013). The future Dark Energy Spectroscopic Instrument (DESI, in development planning now) will require a targeting catalog on the order of the same size (estimated from survey area and depth), in addition to whatever final catalogs, reduced imaging data or extracted spectra are produced as part of the survey itself.

At this scale of database size, one cannot simply load into a standard single-machine database and remain performant for complex queries. The database must be split across multiple machines and queries submitted in parallel with results being aggregated before return to the client (see Figure 2). LSST examined the problem for their even larger data volume, concluded there is no viable turnkey solution available and began the *QServ* project[3] to handle their distributed database needs.

Based on preliminary analysis, our plan is to adopt a QServ solution since the LSST-driven requirements of a low-cost, scalable and reliable system are an excellent match to our needs for DES. Operationally, hosting a database of this the size will require dedicated hardware for the data to be served efficiently, and it is likely that we will need to extend the data interfaces developed by the main surveys to make the data more useful for other science objectives. Gaining familiarity with QServ as it develops gives the added benefit of preparing both the users and us for the LSST operations era.

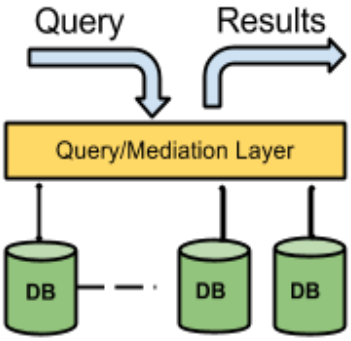

Figure 2. In a distributed DB the mediation layer sends queries to multiple DB instances to execute in parallel. Results are aggregated for the user.

In the context of the Data Lab, we would have one, or more, large catalogs covering up to 5,000 sq. deg. of the sky (for DES) to serve as a starting point for any number of inquiries. Surveys such as LSST and SDSS both document the types of queries supported by their data[*,†] and we plan to adapt those query lists to the research needs identified by our use-case analysis. Such queries would include the ability to do complex SQL queries and sub-setting of the data into personal databases (e.g. similar to the SDSS *CasJobs* and *MyDB* functionality in *SkyServer*[4]). The VO Table Access Protocol[5] (TAP) provides the complex query interface we need; the ability to save a query result to a new user's database (e.g. the familiar the '*MyDB*' capability in SDSS) will require a custom implementation since it will be a feature available to all data services (both local and remote, both basic and advanced) accessible from the Data Lab but is not specified in the TAP protocol. Although our first tools will use direct access to the data during development, TAP provides the needed job-control functionality. Combined with the collaborative storage and analysis services (see below) available within the Data Lab, the ability to import data into a MyDB-like service from other large data centers means we can reduce the number of user up/downloads while making it easier for collaborators to share data and analytical results.

As with other large survey programs (e.g. 2MASS, SDSS, etc) there will need to be multiple interfaces to the data, both programmatic and web-based, in order to satisfy different user scenarios. Using a standard interface to query and access the data means we have options for building web portals (e.g. using PHP or JavaScript frameworks) that are applicable to all the data services. The programmatic interfaces we can build directly into analysis services to fetch the data or in desktop applications for remote use. A multi-language approach[‡] ensures that we are not tied to a particular programming environment, we know from experience that NOAO users represent a broad spectrum of scripting/programming languages and analysis environments. Concentrating development on the data *interfaces* (at least

[*] https://dev.lsstcorp.org/trac/wiki/db/queries
[†] http://cas.sdss.org/dr4/en/help/docs/realquery.asp
[‡] By this we mean a single C/C++ implementation of core functionality with (largely) auto-generated bindings as opposed to separate implementations for each language. High-level tasks will likewise be accessible from many scripting environments; language-specific interfaces can be developed as needed and as resources permit.

initially) should make the transition to programming with the Data Lab much smoother for users. It is hoped that the tools developed by users will then the available through an open source model to provide an even richer scientific content to the data (see below for additional detail on this plan).

3.1.2 Image Access for Associated Catalogs

One of the basic needs identified by the science use cases is for Data Lab users to be able to access the image pixels associated with catalog sources, at least at the postage stamp level. The NOAO Science Archive (NSA) is the raw data archive of images obtained from a variety of NOAO telescopes and instruments nightly, as well as the archive of pipeline-reduced data products; it is the primary distribution mechanism for observer's data (raw or reduced) via a custom web portal. The NSA has its own infrastructure for standardizing data products coming from instruments spanning a range of acquisition systems, as well as an archive architecture optimized to support its web interface. This serves the intended purpose of principal investigator (PI) data distribution well, however the NSA itself is not easily extended to serve as a suitable platform for the data collections we would like to expose in the Data Lab since these would almost certainly not meet the requirements of the NSA for archive ingestion.

What makes the NSA a valuable Data Lab resource, however, is the availability of the pipeline-reduced images. For NOAO observations this covers only the Mosaic, NEWFIRM, and DECam instruments, for which pipelines exist. These instruments, however, accounts for a large fraction of the raw data in the archive, DES reduced data will also become available in the NSA as that survey progresses. With the NSA as a resource the Data Lab will thus provide a ready-made platform for analysis by NOAO observers as well as a source of reduced data products for other astronomers to use for any scientific purpose. Most importantly, DES catalog data can be connected to the original image data available in the NSA.

3.2 Analysis Services

A key capability identified by the science use cases is for users to attach custom analysis workflows to the large catalog data and, in some cases, the associated pixels. In this section, we describe our approach to handling the communication of such routines with the data as well as tools to aid in the construction of the workflows themselves.

3.2.1 Data Handling Services

In order to minimize movement of large amounts of data we want to enable as much processing as close to the data store as possible, either by web services that know how to access the data or with compute resources available to remote users through secure logins or by remote procedure calls. These services must be able to process multiple requests with a single call (e.g. lists of objects or queries) and should be able to run either asynchronously or synchronously to fit well in the model of both large and small workflows. To satisfy these requirements, interfaces to the virtual storage and private database tables are needed in addition to a basic job-control facility or capabilities such as table upload in TAP services.

As a basic example of the processing we would like to enable, consider the following fictional workflow, which emulates the needs identified by our science use cases:

1. A user submits a query of the DES catalog and gets back a table of 10,000 galaxy positions matching the color criteria; she saves this as a text file to her virtual storage space.
2. These positions are sent to a cutout service to create a 50x50 pixel cutout around each galaxy from parent DES images, again saving the cutouts to virtual storage.
3. A galaxy morphology application the user has uploaded to (or is already available in) the Data Lab is run on each image, and the resulting shape parameters are used to select according to user-defined parameters,
4. The sub-selected object list is used to query for additional image data in infrared, cutouts generated and saved,
5. The user downloads DES data, cutouts and shape information only for objects of interest for further analysis on her desktop

The final download in this case is many orders of magnitude smaller than the data that generated it. Moreover, this process can be repeated easily using different selection criteria or with alternative tools to analyze the cutouts. This allows an astronomer to manually execute the steps needed to refine a process before scripting it to build a more powerful tool for use on other data or by other users. What makes the Data Lab unique, however, is that (in the fully realized vision of the Data Lab) this workflow could be accomplished in the same way using the web, programmatic or desktop interfaces.

The above example also demonstrates the two types of processing services needed: *cutouts* are an example of generic image processing tools that can be used with any available image, similar tools for catalogs (e.g. cross match or table statistics) or spectra (e.g. dispersion rebinning) also fall into this category. These generic tools can be defined as standard methods of an *image, spectrum* or *table* object in a programmatic interface to hide the details of how it is implemented or whether the data are local or remote, allowing the scientist to focus on how the data are analyzed instead. In many cases the functionality needed is already available in legacy applications so adding this functionality to a programmatic interface is a matter of using a standard method to invoke the task.

The second type of service is something like the galaxy morphology tool used in the example, i.e. a specific science code to operate on the data. In this case the Data Lab can provide interfaces that simplify access to e.g. the virtual file storage or databases, however the astronomer must provide the needed analysis tools since the possibilities are so open-ended. It is for this reason the Data Lab is adopting a multi-language approach to its tools and interfaces, focusing first on support for popular languages such as *Python*[*] and *R*[†], but using high-level tasks or a single C-based library implementation when possible to provide the core functionality and minimize the language-specific interface development required. This approach also makes the system accessible to teams like SMASH/Bulge that might do the analysis of the data if a variety on languages without requiring direct support from the Data Lab itself, i.e. analysis could be done in IDL and use the command-line tools provided by the Data Lab for data access/movement long before we would have direct interfaces available in IDL itself. Similarly, favorite analysis environments routinely provide tools for photometry or source detection, however we also want to support user-developed algorithms in the long term.

Visualization will also be an important part of the processing of large datasets; for web-based tools we can provide basic image display and general plotting capabilities using existing tools (e.g. JavaScript versions of DS9[‡] or Aladin[§], SAMP[6] interaction with desktop applications) fairly quickly. In scripting environments such as *Python*, or *R* where specialized plot types are available, we will provide basic plotting tools where needed but the astronomer is free to use other preferred plotting tools if they so choose (e.g. by using alternate libraries in their application instead of the default ones supplied by the machine).

3.2.2 Advanced Services

The basic data access services described above are adequate for small collections where the image/spectral data in a result table are the correct format and the size is manageable by a user, or the catalog data is topic-specific and all columns are needed, or where a coarse positional query with a few common constraints would similarly produce little overhead to use. However, large survey catalogs contain information that makes it possible to form complex queries based on color characteristics, shape parameters, data quality flags, proximity to other objects and so on; these complex queries require an SQL-like syntax to be expressed properly or because multiple database tables are involved, something which is not supported by the parameter-based query mechanism of the basic service protocols. Likewise, cross matching a user catalog is best done on the large catalog machine and not against downloads on the user's local desktop. In order to satisfy the requirements of the science use cases we want to support in the Data Lab, more advanced data services will be required.

Table Access Protocol (TAP) -- TAP defines a service protocol for accessing database tables and tabular data remotely using a modified SQL syntax called ADQL (Astronomical Data Query Language)[7] that solves the above limitations nicely. TAP exposes the entire set of a database table schema to a client allowing access to survey programs with complex data products, the *large catalog* data that will no doubt be more than a single table, and database tables

[*] http://www.python.org/
[†] http://www.r-project.org/
[‡] http://ds9.si.edu/
[§] http://aladin.u-strasbg.fr/

used to operate the Data Lab services. A TAP data service not only permits complex queries of something like the DES catalog in a way which can mimic the CasJobs and SkyServer functionalities familiar to SDSS users, but enables simple queries against all data services with a single call as well. Requests may be either synchronous or asynchronous to support responsive applications even when using long-running queries. This generality however comes at the expense of a greater complexity to implement the TAP interface properly, both in collecting the required metadata to deploy services, and in creating client software and toolkits to manage the protocol when accessing remote services from within the Data Lab. A number of TAP server implementations exist which could serve as the basis for developing a TAP service. The available client-side interfaces are all highly language-specific however and thus more difficult to integrate into our plan fully. Nevertheless, we expect to be able to make use of existing TAP client code until a better solution is possible.

DataLink – The DataLink protocol[8] (again from the VO) links the primary data product in a query result table to other associated data or services that might operate on the data. For example, a query might discover a FITS image but what the client requires is a JPEG image for display in a web browser; large images (e.g. a stacked DECam image or data cube) might contain many more pixels that were requested by a positional search and require a cutout; or a reduced image might link back to the raw images used to create the stack. A DataLink service can be used to create all of these data products dynamically to give the user exactly what they need or allowing clients to drill-down to more refined data products.

Cross Match – Catalog cross identification is a fundamental tool in analysis workflows, but is also the most difficult to generalize because of widely varying requirements of specific problems. In the Data Lab we actually have two use-cases to support: the cross match of user-catalogs against Data Lab resources (e.g. DES), and matching tables created *in* the Data Lab with external resources (e.g. SDSS or 2MASS). In the first case we can use the TAP capability to upload a user table to perform a simple positional match, in the second case we are at the mercy of capabilities of the external service unless the data table is small enough (e.g. a table from a journal or a restricted region of the sky) to reasonably import it entirely as just another user table and then rely on the cross-match tools of the Data Lab to do the computation. These matching tools might implement different algorithms or produce different output tables and so we expect to support a range of cross-match capabilities using both local and remote data services.

Implementation of these advanced services will be delayed until the Data Lab development is well underway due to their complexity. However, clients can make use the basic services to achieve the same effect until then with some loss in efficiency. These services are, however, required in order to fulfill the long-term vision of the capabilities that will be provided by the platform.

3.3 Virtual Storage

There are two main motivations for providing a large virtual storage capability in the Data Lab: the analysis services that need to save intermediate results or stage data for download at a later time, and to support the sharing of the resultant data with other users in a collaboration. The data access protocols all respond immediately with a result table (in the case of a query) or direct access to the data as a response to the HTTP request (for data access). Dynamically generated data (e.g. image cutouts) are created as part of the request processing and may be staged to a service's working storage, but are otherwise not available to the user unless an identical request made that effectively then creates a new result. Similarly, results from traditional access-only data centers can only be shared by disseminating the URL to create the result, or more commonly by having one user utilize other mechanisms (e.g. Dropbox[*], upload to a project or personal web site, etc) to make the data available to their collaborators.

The Data Lab will solve these shortcomings by providing a large storage area as a basic capability of the system. Storage may be allocated in a variety of ways depending on the number of users being supported and the resources available, but roughly speaking these break down into the following categories:

Personal Space	A quota allocated to individual users for storage of whatever is deemed necessary. All Data Lab users will be allocated a personal space and may request an increased quota as needed.

[*] http://www.dropbox.com/

Transient Space	A working partition with a limited lifespan intended solely for the staging of data during a processing workflow. In some circumstances output may be put into this space assuming it will either be downloaded or transferred to a more permanent space before the data expire. Quotas are not generally imposed on transient spaces spaces due to their short-lived nature.
Collaborative Space	A long-term storage facility providing redundant backup meant to support specific science collaborations. Access to this space will be by request and subject to approval and resource availability.

Transient space is managed as a "`/staging`" directory of either personal or collaborative space and is not counted in the quota of space used; effectively, it is a symbolic link to the working storage and under its own resource management. Users may request that results be put into any of these spaces using a specific URI or with specialized methods in the programmatic/desktop interfaces, services may default to use one type of space over another unless otherwise requested. Note that these Data Lab storage spaces can exchange data with other machines using the VOSpace[9] protocol (e.g. a Data Lab virtual machine running at the user's institution), effectively creating a download by specifying a remote space as the output of a request.

System interfaces will allow users to move data between spaces as needed (see Figure 3). For example, a query result may be in a transient space unless/until the user moves it to a personal space for permanent storage. If that personal space is shared with a collaborator it might then be moved to the project's collaborative space and then downloaded by even more team members much later. Users will have full control over which other Data Lab users have access to their personal space, or which users are allowed access to a collaborative space.

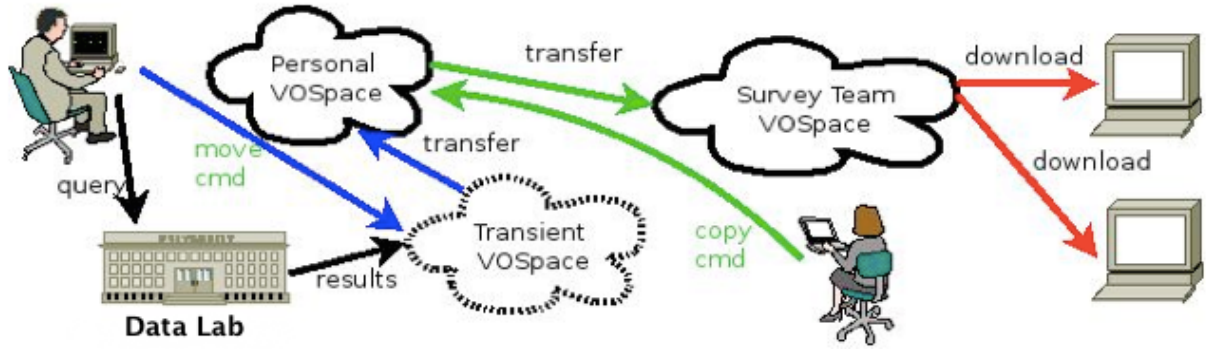

Figure 3. A Virtual Storage collaboration example. Results from a query are stored in the Transient space for a short period. The PI may move this to longer-term storage; collaborators may independently copy or download the data.

Existing implementations are available to build the virtual storage scheme we describe; these are based on the VOSpace protocol for describing how to use a distributed storage system without specifying the backend storage mechanisms directly. Additionally, the VOSpace protocol supports the concepts of *capabilities* on data nodes (e.g. to create an image access service that makes the contents of a directory searchable) and *views* to describe which formats are accepted for input or can be provided on output. The use of *views* would allow us, for instance, to make all of an observer's raw or reduced data visible in their personal space for a specific proposal without actually moving the data until it was accessed, i.e. the space contains a virtual directory listing of a request to the NSA for the data associated with a specific proposal-id. Transfer methods between storage nodes is also negotiable, allowing us to implement an optimal transport method for the size of the data. Capabilities such as creating a directory that can be synchronized automatically between storage nodes (e.g. a Dropbox-like interface), or direct mounting of the space as a user file system, can be layered on top of the base VOSpace protocols. Although the bulk of the storage available will be hosted on Data Lab hardware, we also expect to make storage nodes available as virtual machines (see below) to run on user's hardware; this will increase the available capacity by letting users host local storage for use in their workflows.

3.4 Data Publication

3.4.1 Publication Tools

A key goal for the Data Lab is to provide an environment for users to share their data and results easily with others. In this section we'll refer to *data publication* in the sense of making data available through network services to users. What distinguishes this definition from the NSA or large catalog services described above are the data collections themselves and how publication will be a central concept in the fully-realized Data Lab. NOAO Survey Program data has a separate archiving infrastructure that is no longer in development and so is difficult to update with new data, yet there is an ongoing commitment to make these data available to maximize their impact.

Initially our focus will be on developing *basic* services for catalog, image and spectral data collections. Later stages of the project will implement advanced query and access capabilities to these data (see below). Basic in this case refers to a simple (single) positional query with optional parameters to constrain results by bandpass, spatial resolution or time of observation (amongst others); queries for large numbers of objects are handled either by an advanced TAP service or within the calling application. The data themselves must also be relatively simple initially, e.g. a single flat table containing positional information or homogeneous collections of image or spectral data with a WCS. The advanced services developed later are better suited to publishing arbitrary tables (or indeed full schema) and to linking complex datasets (e.g. collections containing related catalogs, images and spectra).

The assumption being made is that all of these collections can in some way be mapped to the core set of metadata required by the service type; for image data taken with NOAO instruments, this is almost guaranteed. However, because the reduction process can be done in a variety of environments we must still rely on the astronomer publishing the data to provide the definitive information for the mapping tools (see the discussion of ingestion below), ultimately they are the ones responsible for the scientific curation of the data. The configuration mapping and the data files are then fed to the publishing tools to create the service metadata, populate a database to be queried, and then deploy the data service endpoint and a minimal project web interface (based on a template). The entire process has to be automated as much as possible (see Figure 4) to support many users with minimal staffing, and because over the lifetime of the project these services will need to be regenerated multiple times as the software evolves and new features become available, or with new releases of the data. To keep the project manageable we will have to impose constraints on the data regarding acceptable formats and required/optional keywords; in some cases this may require the data author to undertake a major reformatting of the data before it can be accepted.

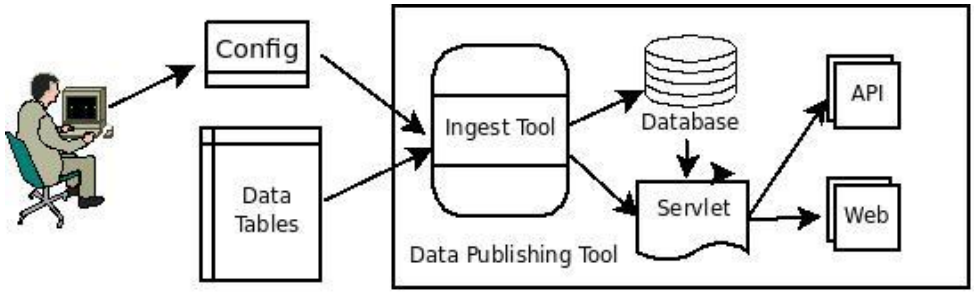

Figure 4. Straw man for the data-publishing tool, example showing catalog publication.

Automation also becomes important when considering the need to publish intermediate data via these interfaces during an analysis workflow. For example, an astronomer creates a filtered table of millions of interesting objects from the DES catalog that she wants to use as a temporary data resource for their analysis application. Although this filtered table can be saved to the MyDB personal table, the applications to be used require that the MyDB table be accessible as a data service. Similarly, a survey team may wish to share the data reduced midway through a multi-year program only with members of the team using a private service. Services that are publically visible will usually be created and maintained by Data Lab staff in partnership with the data author. However a model where astronomers (or tools, or indeed other services) can easily create temporary data resources will also be supported once development of the Data Lab is complete.

A number of data publishing toolkits already exist that each satisfy a different subset of all the requirements identified so far, but the most promising of these is the DALServer* project from the VAO. Although DALServer does not, yet, supply the advanced capabilities we would like, it has the compelling feature is that the service deployment is almost entirely data driven for basic services, and also matches our model of requiring only metadata descriptions and data files (or database table) while minimizing development costs. Features such as creating a resource web site will require additional descriptive text from the data providers, however the goal of automating the publishing process should be easily achieved with scripting.

3.4.2 Registry Services

As use of the Data Lab data publication system grows, Registry services will be needed in order to make user-published data discoverable by other users. We plan to use the VO Registry, and so must be able to provide at least the minimal metadata required for a Registry record to be viable. The full resource description is specified by the VO[10] and is based on the Dublin Core metadata standard†; this provides information such as curation and content metadata (e.g. author contact details, scientific subject and descriptive text), coverage metadata (e.g. footprint on the sky, spectral, temporal and resolution information) and capabilities of the resource (i.e. the data services provided). Extensions to the core resource description exist to further refine the information available to include test queries of a service and detailed information about table columns[11,12]. This level of metadata is something we only expect to generate for permanent datasets hosted in the Data Lab and for major resources such as the DES catalog.

There is no specific requirement that we implement a Registry service ourselves. Unless or until the Data Lab hosts a large number of data services, it will suffice to manually register any large catalog resources with an existing Registry service. If during operations we find that the number of new services being registered, or updates to existing records, becomes burdensome then implementing a local *publishing registry* (i.e. a small registry of local resources that is harvested for records by the wider VO) may become cost effective.

Populating the XML resource records however, cannot be avoided, although there is much that can be done to automate their creation. For collection metadata, a comprehensive web form can be used to gather the information from the astronomer and generate the record automatically, and the detailed tabular metadata needed to generate the service is sufficient to populate a detailed resource record as well. A simple script using XSLT[13] to extract information from the XML record is adequate to create the configuration files necessary to deploy the data service or create the project web page, however thought will need to be given to the best approach to take that permits easy editing of an existing record without resorting to implementing a complete resource database or recreating the record from scratch.

Temporary data services (e.g. those created to access a MyDB table) can be utilized without being fully registered since the access protocols require only a base URL and query parameters. A naming convention should suffice to form the base URL in many cases, allowing applications to construct a correct query URL. However we make no guarantee that these temporary services will be visible to all components of the Data Lab without a complete registry implementation.

3.4.3 Data Ingestion

The data ingestion process can be summarized as requiring multiple human- and machine-generated metadata files as input, tools to scrape information from data files, a common set of processed configuration files as output, and glue code to control the entire process. The publication of new (or updated) datasets will be relatively rare (a few times a year perhaps), however we expect that as the system evolves it will be necessary to re-ingest data much more often. For data authors, detailed documentation describing how to prepare the data files and what metadata is required will be needed to guide them through the ingestion process as well.

In terms of the data publication process, the *metadata* about the collection is even more important than the data itself. This metadata is used both for discovering individual images, spectra or rows in a table, as well as in describing the collection as a whole (e.g. the author's contact information, the footprint of the survey, descriptive text about the

* http://wiki.ivoa.net/internal/IVOA/InterOpMay2010Val/dalserver.pdf
† http://dublincore.org/documents/dcmi-terms/

scientific content, etc). As such, the process of collecting all the needed metadata requires information that can only be supplied by the astronomer/author, in addition to that which can be obtained automatically from the individual data files.

We will rely on an existing VO standard for a core observation data model to use, the ObsCore DM[14]. ObsCore was developed following an analysis of the data discovery use-cases defining the scope of the needed data model. Specifically (quoting from the ObsCore document):

- Support multi-wavelength as well as positional and temporal searches,
- Support any type of science data product (image, cube, spectrum, time series, instrumental data, etc),
- Directly support the sorts of file content typically found in archives (FITS, VOTable, compressed files, instrumental data, etc).

The ObsCore model is similar to the Common Archive Observation Model (CAOM)[15] but is much lighter in weight, consisting of only about twenty elements needed to describe most data sets. This is adequate for registering a service to discover the data, however the CAOM can be used more effectively with advanced services such as TAP as a means of querying across multiple data collections. A later implementation of CAOM fortunately does not deprecate use of the ObsCore model and so we will defer its use until later in the project.

For a wide array of different types of data, the list of format-specific issues to be addressed is quite lengthy and will make developing an ingestion tool (even one limited to a restricted set of data formats) challenging, however the output of such a tool will *always* be a set of metadata standard across all published datasets. A modular approach to building such a tool will provide the greatest flexibility in adapting it to unfamiliar datasets or to extending the capabilities of any one phase of the processing. The mapping of derivable metadata or each type of data (catalog or image or spectrum) to the ObsCore model will determine the amount of human-supplied information that is required, i.e. different web forms will be needed for different types of data. Documentation is also critical to understanding the process and reducing errors.

Finally, it is worth considering when designing the ingest procedures what additional metadata might be collected that will benefit advanced features of the Data Lab we would like to implement in the future. For example, Multi-order Coverage Maps (MOC)[16] specify arbitrary sky regions that could be used in a survey-completion visualization tool. The Hierarchical Progressive Surveys (HiPS)[17] mechanism from CDS is similar to Google KML[*] mapping technology, and allows a gradual progression of detail in catalog overlays on image displays. Although we can extend the ingestion tools and metadata scrapers in the future to create the information needed for these visualizations, we would like to avoid reprocessing all of the data so long as gathering this extended information does not unduly complicate the ingestion process.

3.5 Packaging as Virtual Machines

The hardware requirements of the Data Lab will depend on decisions made about how many users and datasets can be supported and the development timeline, however we expect to over-subscribe much of the hardware and deploy services using virtual machines (VM). This allows us to deploy heavily used resources to dedicated hardware, keep all other services online using a single server hosting multiple VMs where the expected load is much lower, and easily redeploy services as the system expands. Broadly speaking these VMs can be classified as being *storage*, *publication* or *compute* machines. Each type of VM will be configured with a base operating system suited for its purpose; the specific Data Lab software will then be added from a central repository allowing us to update the OS or software at any time to create a new machine.

These virtual machines can be created easily but are not limited to use on the Data Lab hardware; users can easily download the machine images to be run on their hardware. A storage machine, for example, could run as an instance at a user's institution and work in concert with the Data Lab (or other users) to synchronize files amongst collaborators, or a publication machine could be used to host a dataset locally. Most interesting is the idea that a compute machine can be deployed in the Data Lab and configured by an astronomer with tools needed for a particular type of

[*] https://developers.google.com/kml/

analysis, and once the system is refined, the entire VM can be moved to a commercial cloud such as Amazon for large-scale processing.

Not all of these VMs will be needed for the operational aspects of running the Data Lab, we plan to develop a number of machines configured with pre-installed analysis environments to serve as a starting point for development machines used by astronomers. This might include standard installations of *Python* or *R* as a programming environment or something as complex as a full LSST software installation that could be used to compare output against a trusted result without requiring users to configure the system themselves. Providing a library of pre-configured machine types makes it much easier for a user to login to their Data Lab account and start work, or else download the needed system for local use and interact with the Data Lab remotely. Registry services can be used to keep track of systems in a way that allows the system interfaces (see below) to make the best use of local and remote data or services.

4 SYSTEM INTERFACES AND ARCHITECTURE

The Data Lab will be accessible in a number of ways depending on the intentions and the skill level of the user. Web interfaces are convenient for browsing data, but algorithm development can require much lower-level interfaces, and both can quickly become a major effort within the project. Although we would like to support development at all levels eventually, our initial focus will be on developing command-line tools to provide a high-level capability. These tools can be invoked from a wide variety of environments easily and will provide the needed functionality until a more specific interface can be implemented.

Remote execution is a key concept in accessing the Data Lab and can be accomplished in a number of ways. The analysis services described above are not much different from the data access services in that they will also have RESTful interfaces[18] to invoke them and obtain a result, allowing us to build the web interface from underlying desktop tools, or likewise to build desktop tools that use network services. Existing interfaces to data services execute a query and return a result table or image/spectrum collection. However, this model doesn't scale well when the amount of data to be transferred in a reasonable time exceeds network capacity, or when a mix of local and remote data is required in the analysis. In this case what is required is the ability to remotely execute some method close to the data and return only a simple status or some equally small bit of information (e.g. the statistics of a table column).

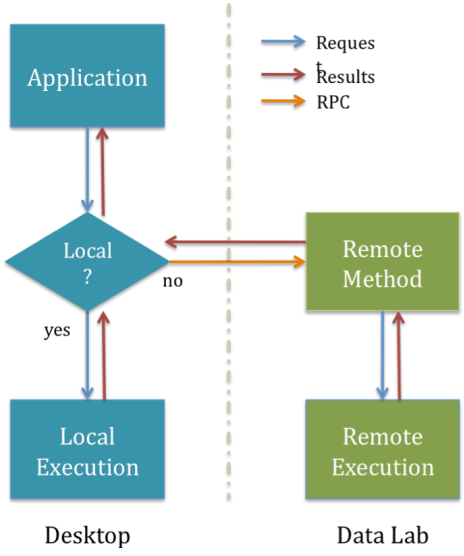

Figure 5: A schematic of the proposed API architecture showing the use of remote execution.

The programmatic interfaces being planned all include the idea of *location-agnostic data*, i.e. an application may use an image, table or spectrum object created from a local file or from a URI referencing some remote data (e.g. a FITS image in the VOSpace). Methods called on that object may be as simple as reading the values of a table row or performing some action such as computing the median of a pixel region, however the task is unaware of whether the method is executed on the local machine or remotely on the resource in the Data Lab (see Figure 5). Technologies such as Pyro* provide remote object execution frameworks that could be leveraged in some specific interfaces; the actual implementation will depend on the method being called, however. For example, the image interface might declare a *display()* method for displaying an image which is implemented using the SAMP messaging protocol to tell a third-party display program to load an image (send either a local or remote URI as needed); whereas a *readPixels()* method would require a Pyro or some other low-level RPC mechanism. In keeping with the goal that these methods should minimize the amount of data transferred there may not be a direct mapping between what the interface method returns and the more general web service doing the same operation. For example, an API *execQuery()* might create a new result object (that could be access remotely) in an interface where the web service might return then entire result table.

* https://pypi.python.org/pypi/Pyro4

In the Data Lab, this architecture will allow applications to be built that permit multiple users to run the same application on a single copy of the remote data (with appropriate precautions being taken for simultaneous access), minimizing the required data transfer. Because the shared access is done automatically by the interface the astronomer is able to concentrate on the science content of the code being developed.

5 IMPLEMENTATION PLAN

The Data Lab will be developed using a phased approach leading up to full-scale development that is expected to begin in FY16. Phase One (P-1) will begin with the work needed to support the SMASH and Bulge science cases and will be considered complete when we can support science workflows such as the example given in Sec 3.2.1 above. In this section we describe only the work on each of the Data Lab components that provides these needed basic science capabilities to be completed during this initial phase, all other component features are deferred and will be prioritized for implementation upon later review.

5.1 Large Catalogs

The DES catalog is the intermediate goal to supporting data on the scale of LSST, but will not be delivered as a first release until well after full development of the Data Lab begins. On the other hand, the SMASH and Bulge final survey data catalogs are sufficiently challenging to develop the needed tools, will be delivered much sooner, and will continually grow in size during our P-1 implementation as the surveys near completion. The survey teams are responsible for assembling and preparing the database tables to be used; the starting point for the Data Lab development is assumed to be a science-ready catalog.

For phase one, the focus will be on using these datasets for developing the needed large-database tools rather than as a testbed for the QServ database technology needed by catalogs the scale of DES. Example P-1 capabilities to be implemented include:

Query/Table Tools	• Command-line tools (scriptable from multiple environments) able to submit an SQL query and return a result in a user defined-format for download, save to virtual storage, or load into user MyDB. • Generic DB table manipulation tools, e.g. column sort, add/delete, append, crossmatch, etc • Basicic statistical analysis tools and toolkits capable of reading saved query result formats
Visualization	• General table (1-D and 2-D) plotting utilities • Interactive selection of data from plots • Catalog overlay on images
User Database	• Ability to save results to new user table (using an interface façade)
Parallel Databases	• Begin prototyping effort to deploy QServ-based catalogs and client services

5.2 Analysis Services

Analysis services during P-1 will be limited to the database and visualization tools above with the exception of a simple image cutout task and/or service to provide basic pixel-level access. Positional crossmatch of tables will be supported through use of stored procedures in the database to provide optimal performance; more sophisticated crossmatch tools are deferred unless implemented by the survey teams themselves.

5.3 Virtual Storage

For P-1 development a virtual storage capability is required for saving intermediate results and to support shared use of the data by both the SMASH and Bulge teams. We will implement only the 'collaborative space' described above with a preliminary authentication mechanism to avoid the overhead of account and resource management in the other storage types. A programmatic interface to the space will be needed to enable the catalog and analysis tools to store/retrieve data and the build the command-line tools needed by the users.

5.4 System Interfaces

The only significant interface development required for P-1 is to the virtual storage space to allow tasks (both in the Data Lab and on the user's desktop) to move data around. Client-side interfaces to access remote archives already exist and will be modified as needed to support VOSpace usage (i.e. saving results directly to the space). The primary interface used by the SMASH and Bulge teams will be command-line tools providing a high-level capability since these can be invoked from a variety of analysis environments. Low-level APIs used by the Data Lab developers can be made available to these teams if requested but are not planned for general release at this stage. Simple web interfaces to support the survey teams are possible and will be considered as needed.

For analysis it is assumed users will either be logged into a virtual machine running in the Data Lab and will thus have direct access to the data files, or the analysis is run on a local desktop and data are transferred into and out of the Data Lab storage as needed. Since the catalog and other command-line tools can be run over the network to query and save results to storage *in* the Data Lab, and *ssh* can be used to invoke a task remotely, a workflow (such as the example in Sec 3.2.1) can be executed either in the Data Lab or the user's desktop with no additional development effort.

5.5 Data Publication

The general publication capability of Survey and PI data products is not needed initially to support either SMASH or Bulge since these data will be accessible through standard database interfaces as the analysis proceeds. Once these surveys near completion of a final set of data products we will, however, use these data as the basis for developing the publishing capability to be generalized later.

For P-1 development some of the ingestion functionality will be required (e.g. metadata collection) to build tools for loading the databases, but the scope can be limited to the data formats used by the SMASH and Bulge teams. In-house Registry services (and metadata collection) will not be required for P-1 implementation; when needed, required services can manually be registered with an existing Registry.

5.6 Packaging as Virtual Machines

For P-1 development virtual machine images will be required for supporting interactive logins by the SMASH and Bulge teams and to create development and test systems. These images can be made available to the teams, as required, but there are no plans for scheduled distributions to users.

ACKNOWLEDGEMENTS

We wish to acknowledge the many useful conversations and comments received from the SMASH and Bulge survey teams in defining the requirements for the Data Lab. This project concept was developed in part with support from the Virtual Astronomical Observatory contract AST0834235.

REFERENCES

[1] Anonymous, "Description of the Dark Energy Survey for Astronomers", 1 May 2012, <https://www.darkenergysurvey.org/survey/des-description.pdf> (15 June 2014)

[2] Saha, A. et al. 2010, "First Results from the NOAO Survey of the Outer Limits of the Magellanic Clouds", *Astrophysical Journal*, **140**, 1719

[3] A. Szalay et al., "The SDSS SkyServer - Public Access to the Sloan Digital Sky Server Data," Proc. 2002 ACM SIGMOD Int'l Conf. Management of Data, ACM Press, 2002, pp. 570–581.

[4] Wang, D.L.; Monkewitz, S.M.; Kian-Tat Lim; Becla, J., "Qserv: A distributed shared-nothing database for the LSST catalog," *High Performance Computing, Networking, Storage and Analysis (SC), 2011 International Conference for* , vol., no., pp.1,11, 12-18 Nov. 2011

[5] Dowler, P., Rixon, G., Tody, D., "Table Access Protocol, Version 1.0", <http://ivoa.net/documents/TAP/>, IVOA Recommendation 27 March 2010

[6] Taylor, M., et al, "Simple Applications Messaging Protocol, Version 1.3", <http://ivoa.net/documents/SAMP/>, IVOA Recommendation 11 April 2012

[7] Oriz, I., et al, "IVOA Astronomical Data Query Language, Version 2.0", <http://ivoa.net/documents/ADQL/>, IVOA Recommendation 30 October 2008

[8] Dowler, P., et al, "DataLink, Version 1.0", <http://ivoa.net/documents/DataLink/>, IVOA Recommendation 05 May 2014

[9] Graham, M., et al, "VOSpace specification, Version 2.0", <http://ivoa.net/documents/VOSpace/>, IVOA Recommendation 29 March 2013

[10] Hanisch, R., "Resource Metadata for the Virtual Observatory, Version 1.12", < http://www.ivoa.net/documents/latest/RM.html>, IVOA Recommendation 02 March 2007

[11] Plante, R., et al, "Describing Simple Data Access Layer Servies, Version 1.0", <http://ivoa.net/documents/SimpleDALRegExt>, IVOA Recommendation 25 November 2013

[12] Plante, R., et al, "VOResource: an XML Encoding Schema for Resource Metadata, Version 1.03", <http://ivoa.net/documents/latest/VOResource.html>, IVOA Recommendation 22 February 2008

[13] "XSL Transformations (XSLT) Version 1.0: W3C Recommendation – Embedding Stylesheets". W3C. 16 November 1999. Retrieved 2009-01-06

[14] Louys, M., et al, "Observation Data Model Core Components and its Implementation in the Table Access Protocol, Version 1.0", <http://ivoa.net/documents/ObsCore/>, IVOA Recommendation 28 October 2011

[15] Dowler, P., Gaudet, S., Durand, D., Redman, R., Hill, N., & Goliath, S. 2008, in ADASS XVII,edited by R. W. Argyle, P. S. Bunclark, & J. R. Lewis, vol. 394 of ASP Conf. Ser., 426

[16] Bosch, T., et al, "MOC – HEALPix Multi-Order Coverage Maps, Version 1.0", <http://ivoa.net/documents/MOC/>, IVOA Proposed Recommendation 10 March 2014

[17] Fernique, P., "HiPS – Hierarchical Progressive Survey", <http://aladin.u-strasbg.fr/HiPS/HiPS%20technical%20doc.pdf> (retrieved 14 June 2014), May 2014

[18] Richardson, Leonard; Sam Ruby (2007), "RESTful web service", O'Reilly Media, ISBN 978-0-596-52926-0, retrieved 14 June 2014