NOAO Imaging Metadata Quality Improvement

A Case Study of the Evolution of a Service Oriented System

Sonya J. Lowry
National Optical Astronomy Observatory
lowry@noao.edu

Phillip B. Warner
National Optical Astronomy Observatory
pwarner@noao.edu

Evan Deaubl
National Optical Astronomy Observatory
deaubl@noao.edu

Abstract

Due to the natures of legacy astronomical imaging metadata acquisition and storage technologies and techniques at the National Optical Astronomy Observatory, the challenge of methodically improving the quality of such information has been insurmountable until now. The diversity of the sources and lack of cohesive effort along with technologies that enable silo style efforts have contributed to the current, disorganized state of the collection of astronomical imaging information. By meeting these issues with a solution that combines policy and technology support for implementing master data management, use of transformations that enable continued improvements and a history of changes made, and a modular architecture based upon services that are federated over a flexible, robust communications architecture, the NOAO Archive System can assure improvements in the quality of the imaging metadata now and into the future.

Categories and Subject Descriptors H.0 [*Information Systems*]: General; D.2.11 [*Software Engineering*]: Software Architectures—Patterns (e.g., client/server, pipeline, blackboard)

General Terms Design

Keywords Astronomy, Data Quality, Integration, Metadata, SOA, Service Platform, Service-Oriented Architecture, Transformation, Virtual Observatory

1. Introduction

Dealing with National Optical Astronomy Observatory's (NOAO) astronomical data poses some unique problems for an information system. Such a system must manage new and legacy data collected using multiple technologies and formats and deal with the varying quality and reliability of imaging metadata from the distributed data collection points in a way that supports access under evolving Virtual Observatory (VO) protocols [4].

Possibly the biggest challenge is finding ways to incrementally improve the quality of the imaging metadata[1] used by astronomers to differentiate the images of interest in their science from the terabytes of images stored within the archive system. This document describes how we evolved our system designed as a Service Oriented Architecture[11, 12] to solve this problem and thereby improved the way amateur and professional astronomers access NOAO data.

In general, the NOAO Archive System collects astronomical data via a data transport system at the mountain cache locations, persists that data redundantly across locations in both the Northern and Southern hemispheres, initiates file tracking behaviors, extracts and stores imaging metadata from header keyword values, and provides secure access to the stored metadata and data products via Virtual Observatory protocols.

Our data quality initiative extends the capabilities of the archive system to include features for validating, correcting, and regularizing the imaging metadata astronomers use to query against our astronomical data products.

2. Context and Requirements

Metadata with high fidelity enables the discovery of datasets throughout the VO in specific regions of interest, time-domain searches, filtering on relevant bandpasses, etc. Unfortunately, legacy NOAO data acquisition systems provide metadata that are quite heterogeneous across instruments, both in form and content. In addition, the metadata are frequently inaccurate or are missing altogether.

Data quality improvement is also orthogonally affected, and further amplified by, the limitations of the data format generally used in the astronomical community: the data and metadata are combined into a single file using the Flexible Image Transport System (FITS) Standard [18, 2]. Although the standard is continuously being improved (albeit at a glacial pace), improving the quality of metadata has not been

OOPSLA'08, October 19–23, 2008, Nashville, Tennessee, USA.
ACM 978-1-60558-220-7/08/10.

Figure 1. Four of the telescopes at Cerro-Tololo Inter-American Observatory (Credit: NOAO/AURA/NSF)

given much consideration (if at all). Hence, we are only able to use the standard for data transport. Further discussion of this format is provided in §2.3 below.

The focus of this effort is to bring critical metadata for all imaging instruments to a common standard for quality and completeness, and to establish a robust base upon which other value-adding processes (such as pipelines) can be built.

The goal is to evolve the NOAO Science Archive System to improve the development and execution of standardized operational processes and procedures that enable the validation of quality and completeness of essential imaging metadata, correct values when feasible, regularize the form and the format of the metadata, add new metadata where needed, tag all metadata with status information, and clearly document all quality improvement activities in a verifiable way.

The legacy approach to this issue has been for system operators to visually inspect collected FITS files for obvious errors in a few critical keyword values and to manually augment all copies of erroneous headers propagated through the system from the previous night of observing.

The laborious nature of this approach prohibits the validation and correction of more than a very few number of keywords as the system collects images each night from several sources in both the Northern and Southern hemispheres. This approach to correction cannot sufficiently provide the high reliability needed for effective scientific queries.

2.1 About NOAO and DPP

As a U.S. National Observatory operating a wide range of telescope facilities distributed throughout the world, NOAO is by definition one of the most significant data providers in the U.S. astronomical "system" which includes both ground-based and space-based facilities. This significance can be quantified both in the variety of instruments supported as well as the volume of data produced.

However, until recently, most of the data obtained at NOAO and affiliated facilities was only captured for backup purposes, e.g., in case principal investigators lost their own data, using a tape-based system. In 2000, NOAO created the Data Products Program (DPP) in order to establish a unified data management infrastructure for the distributed observatory facilities and their users. Our focus was on the general tasks of capturing, archiving, and processing data coming directly from NOAO telescopes and instruments [13].

2.2 Specific Challenges

The technical challenges that DPP faces include data capture from legacy instruments up to 20 years old, the wide geographical distribution of facilities over three mountaintops on two continents, and the non-uniform use of the facilities by visiting astronomers, each with their own data acquisition methodology[14].

One result of decades of such diversity is an enormous collection of legacy data products, each potentially representing a unique model for imaging metadata and in many cases, offering incorrect or missing values for scientifically critical metadata elements.

Additionally, new challenges are being introduced by the growth of the Virtual Observatory efforts and by new instrument projects with expectations of greatly increasing data volumes over what is currently experienced at NOAO facilities [13].

DPP must find a way to make this legacy data fit within the new models for data access with imaging metadata values that offer a high degree of reliability while scaling that same solution to accommodate the growing volumes of future data.

2.3 Key Functional Requirements

For the purposes of this document, astronomical information is composed of image files made up of two parts; the pixel data and the imaging metadata which describes the pixel data and is often represented as FITS file headers with keyword/value pairs representing each data element (see Figure 2).

This information is currently archived by the NOAO Science Archive system which redundantly propagates the FITS files and assimilates header content (imaging metadata) into a relational database. The primary goal of this project is to develop an automated system for performing transformations upon the imaging metadata in order to increase the quality of the metadata with a quantifiable improvement in reliability.

Due to the shear volume of legacy data and the daily volume growth in the (current) range of tens of gigabytes, the system must support data profiling efforts by automating the identification of newly encountered 'models' in the form of FITS files that may not display characteristics of previously identified and mapped sources.

In practice, the system should, when it fails to classify a file as a known type, place an unidentified or uncharacterized file into a holding location for manual review. An operator should then be able to define rules for uniquely classifying that file, validating its content, and configuring and executing the appropriate transformations that will result in improved quality for all imaging metadata of that type.

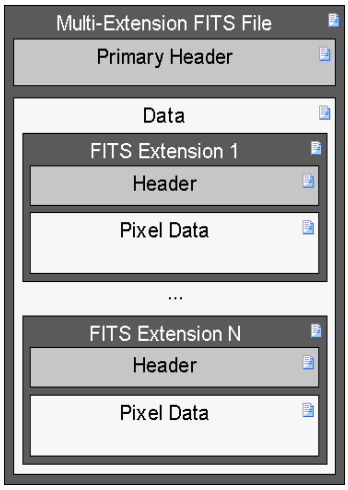

Figure 2. Logical representation of a Multi-Extension FITS (MEF) File. The MEF is basically a FITS file that contains, in the pixel data subset of the file, a set of FITS representations, each of which individually represent a single FITS file. A single FITS file contains a header unit and a data unit.

When a known type is encountered by the system, the appropriate transformation should be performed on the file content and validation of the result should follow. This process is potentially iterative and the solution should support such a scheme.

Additionally, provenance information should be kept in order to roll-back a transformation (if necessary), assign some reasonable level of confidence to new values, feed information back into the ongoing data profiling effort, and most simply for reporting purposes.

The most challenging aspect of this work may well be the distributed nature of potential transformation activities. Certainly, some algorithms for transforming the imaging metadata can be encapsulated in libraries or local services, but there are some that cannot fit into such a paradigm. Reasons for such specialization include political restrictions on deployment options, technical limitations of legacy systems, and dependencies between federated services.

A result of these restrictions and limitations is that some of the most scientifically significant transformations must be performed by sending FITS files to remote services (possibly hosted by other members of the astronomical community). This is complicated by the fact that much of the data collected by our system has a period during which it is proprietary and must be protected.

Further complications result from requirements to manage the state of a transformation over time, as there often are dependencies between steps, and execution of steps may introduce varying or unpredictable delays. An extreme example is a step which requires human interaction, hence a transformation can be held in a waiting state for long periods of time, e.g., weeks while a specific operator is on vacation.

2.4 Key Non-Functional Requirements

Of utmost importance is that the system must rigorously protect the data. This means not only protection against accidental deletion or loss of data and imaging metadata in transit, but also protection based upon a defined security model.

Current NOAO policy includes making imaging metadata publicly available; therefore, remote services which only require that information can be used for transformations on any imaging metadata.

However, when full FITS files are required, the service must conform to the NOAO policy to protect the data from unauthorized access. This can be accomplished by hosting within the NOAO protected network or by adoption of VO security services along with appropriate tunneling features or similar security measures.

Additionally, the system must robustly perform transformations on the volumes of data collected without falling behind. Not only must the system keep up with the existing data volumes, but also with the expected volume growth as new instruments come online and handle the correction of legacy imaging metadata as it is incrementally added to the system.

Furthermore, the data must be corrected and ready for retrieval within a limited timeframe, as astronomers often prefer to retrieve collected data the morning after their run.

3. Paradigms

The NOAO Science Archive system has adopted a number of complimentary paradigms each of which has proven appropriate to our goals. The Service Oriented Architecture (SOA)[11, 12] enables the robust distributed environment needed to execute our location and organizationally constrained processes. Component Based Software Engineering[8, 16] allows us to federate services within our system. Enterprise Application Integration[3] is the paradigm that guides our integration of legacy and third party components as well as our data and metadata. Finally, Metadata Master Data Management brings us the policies and technologies needed to understand what constitutes a true quality improvement.

3.1 Service Oriented Architecture

First and foremost, the system leverages the service oriented paradigm which provides a set of reasonable attributes that must be balanced in a distributed system. A successful Service Oriented Architecture balances these attributes in a way that best accomplished the goals of the system. It is critical to understand that there is no one solution that meets all needs.

Various lists have been created that attempt to summarize what constitutes the traits of a service oriented system. For the purpose of the NOAO Science Archive, the list would include: Reusability, Loose Coupling, Abstraction, Location

Transparency, Modularity, Interoperability, Relevance, Protocol Independence, Autonomy, Discoverability, and Composability.

3.2 Component Based Software Engineering

Component based software engineering has a long, successful history and should not be shucked simply because services look interesting. In fact, the service oriented paradigm, as a complementary concept, brings CBSE to a new level and conversely, CBSE applied to SOA enables greater reuse and improved flexibility in process definition.

The NOAO Science Archive capitalizes on both of these concepts by making use of service federation. Services are our components and federation gives us the ability to modify workflows at a granular level in order to tweak the behaviors of the higher level services understood by science/business users.

3.3 Enterprise Application Integration

Legacy NOAO distributed environments make extensive use of point to point connections and fail to integrate data effectively or at all in some cases. The NOAO Science Archive, while still suffering from an abundance of point to point connections, is attempting to provide a more integrated environment that will improve over time.

EAI, while not suitable for all aspects of our system, does provide some promise particularly in the data integration arena. The astronomical community chose, many years ago, to adopt the FITS file format which has led to a situation where each instrument team develops a new data model for the imaging metadata collected. The goal of our current effort is to improve the quality of this information and doing so depends upon an understanding of the data that can only come from transforming it properly to a single model. EAI gives us this ability.

3.4 Metadata Management and Master Data

In order to correctly validate and correct the imaging metadata we collect, we must organize an effort to reach some agreement on the quality of our master data[10]. This master data serves as metadata describing our system and so, once policy has placed appropriate authority, can be managed within our metadata management system[17].

The state of master data at NOAO has historically been tragic as most master data has never been electronically recorded and never assembled in any form. Some examples include some instruments that have been used for decades that have never been assigned names and lists of optical filters that may, or may not, have descriptions dispersed amongst web sites for various interested organization but never collected and managed by any single group. The collection and management of this master data is prerequisite to any successful attempt to improve the quality of the imaging metadata, as it makes up or serves as a dependency for most of that content.

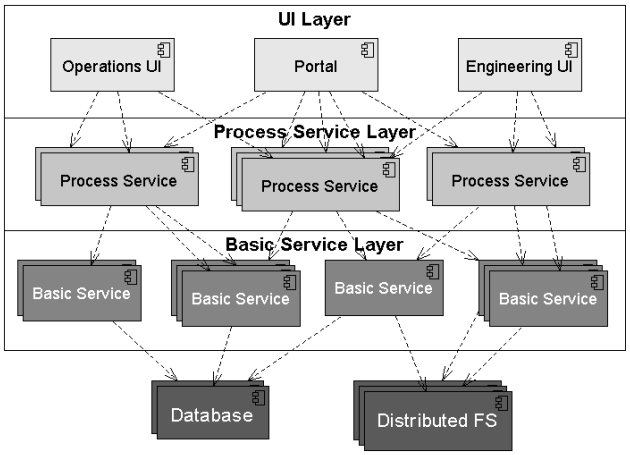

Figure 3. Layers of the Service Platform

4. Our Solution

The architectural approach currently used for our distributed archive system is a Service Oriented Architecture. Our system is an assembly of shared services that provide a platform [6] of components for building service federations [7], a shared metadata management system for managing 'system' metadata, and a communications infrastructure provided by an enterprise service bus. The new data quality addition is an extension to this system. We shall look individually at how each of these was adapted to provide the data quality subsystem.

4.1 Service Platform

Our service platform is made up of a set of variably reusable service components. These services, either *Basic Services* (logic- or data-centric) or *Process Services*, constitute the middle tiers of a layered architecture (see Figure 3).

Logic-centric Basic Services provide access to some bit of execution logic that may be used for classification, transformation execution, or validation. Often, the logic itself may be encapsulated within a library, our own or third-party provided. Nevertheless, a service is required in order to provide request interception, potentially adding additional behaviors.

Data-centric Basic Services provide CRUD access to data elements. Within our data quality system, these elements may include transformations, classification rules, or other 'system' metadata objects used, for example, in the configuration of both Basic and Process Services.

Process Services within our platform are responsible for workflow execution via orchestration or choreography. These services use Basic Services to accomplish a set of business tasks. As such, this type of service is not designed for high reuse.

Figure 4 presents a generalized view of service interactions. As shown in the figure, Process services may use

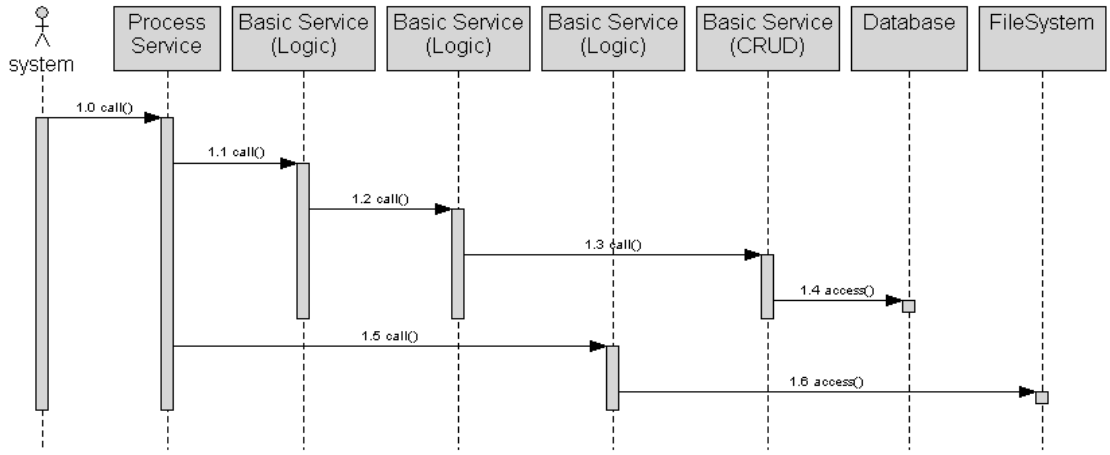

Figure 4. Sequence describing the generalized interaction between services.

any number of Basic services, while Basic services are constrained to interoperate with other Basic services or low-level system resources (e.g., a file system or database). A logical outcome of this constraint is that Basic services must inherently contain a well-defined set of responsibilities (or a single responsibility), thereby increasing their reusability within the system.

In order to solve our imaging metadata problems, we needed to define a set of new Basic and Process Services that will, when combined with some of our existing services, provide the new features requested by both the customer and the operations teams.

We identified a need for a Process Service that is responsible for orchestrating the high level workflow of classifying a file by data source, identifying and fetching the appropriate transformation, executing that transformation, validating the result, optionally iterating back to the beginning of the workflow, persisting the final result, and generating appropriate reports. The atomicity of each of these activities led us to identify the need for new Basic Services, each performing a given activity with the possible assistance of other services. All of these services are new to the system, but may still rely upon existing services as part of a federation.

All imaging metadata must flow through this Process Service. However, the metadata from each data source must be processed using a specific combination of transformations. These transformation workflows are executed at the level of Basic Services. It is this lower level workflow that introduces the potential for high distribution of services. The configuration and deployment of these services are described by 'system' metadata, which are stored in a database (accessible through a Data-centric Basic Service), and configuration files.

Finally, we identified a need for a service or set of services to provide the execution of these lower level transformations. Such services are designed to be configured by external information (as opposed to static configuration used at load time). Therefore, new transformations using existing services can be defined by operators at runtime.

4.2 Metadata Management

Metadata is an integral part of any software system, from the description of the structure of messages to the description of the structure of the database.

'System' metadata can include descriptions of data sources (e.g., the data acquisition system at the telescope), software components in the system, the structure of the FITS files (e.g., Multi-extension FITS, Single-image FITS, spectra, etc.) created by the various survey projects, image metadata, any transformations performed on data and image metadata, events in the system that together form an account of the history, and anything else in the system we find useful to manage.

For the purpose of imaging metadata quality improvement, we have extended our metadata management to include new types of objects for describing rules for classifying files, transforming the imaging metadata and mapping it to the data model, recording the transformation activities, and for validating the results.

We have also taken a step toward a more federated metadata management system by keeping this metadata physically close to the services that use it most, and providing access by the extended system via basic, data-centric services.

An important benefit of the introduction of managed metadata transformations is the ease with which an operator can define new transformations. As long as the system provides the low level components needed, an operator can create a new transformation that simply assembles these components in the appropriate order, and with dependencies that make sense for the intended data source. A graphical user interface that provides a visual way to assemble these

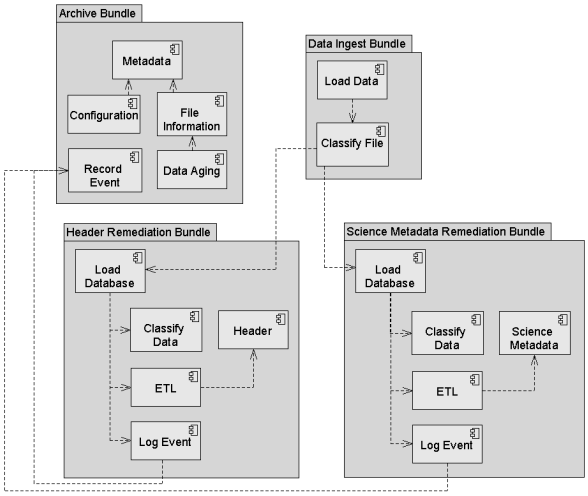

Figure 5. Component Interactions. Components are assembled functionally as bundles. Communications may be within the bundle or between bundles.

components further simplifies the operation of the subsystem.

4.3 Integration and Deployment

Linking the individual services together is an integral part of assembling a cohesive system. Our initial approach accentuated the flexibility of a service-oriented system, making each service individually installable, exposing every service on the network, and linked services using those network connections. We quickly found that this led to performance bottlenecks, increased integration complexity, and complicated deployment procedures. This led us to discover ways where we could greatly improve the simplicity of this process by intentionally reducing the flexibility that is exposed to the deployer.

The deployment architecture of our system includes the use of deployment bundles which are physical groupings of services with complementary functionality. This approach reduces the complexity of the system for the operators responsible for installation and daily operations activities by reducing the number of components they need to install and configure. It also allows us to reduce local communications overhead by using the simplest communications strategies where feasible.

Many of the services in our improved imaging metadata quality project can be co-locate. within a single bundle. The Spring Framework [15] is used to configure these intrabundle communications, avoiding the need to unnecessarily expose services on the network and subsequently keeping the network communication at a minimum.

However, some of the services cannot be deployed on the same node within the bundle, and others that are deployed within the bundle may need to provide remote access to other

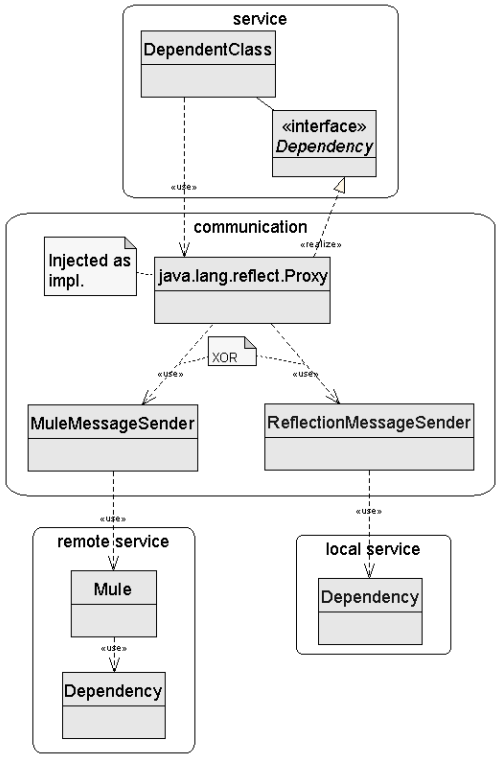

Figure 6. Communications Architecture.

system bundles, or even to external users. To provide these communications capabilities, we use the Mule [9] enterprise service bus (ESB), a middleware system that simplifies exposing and linking services on the network.

Using the imaging metadata quality improvement project as an example, the top level services identified may all be deployed within the same bundle, therefore orchestration at this level is managed by Spring configuration. However, transformation objects must be accessible to components outside the bundle, so a remote interface to those transformation services must be provided through the ESB. The lower level transformation workflows are executed using a combination of components, some living within the bundle and some external to the bundle, meaning that those services are integrated using a combination of both Spring and Mule, a mode of operation well-supported by both frameworks.

However, once we had settled on this design and had been using it for some time, we discovered that some system communications violated the decoupling constraint of the system architecture. This problem stemmed from: (1) the use of Java interfaces to define service interfaces, requiring a service to import the interface libraries of external services on which it depended, and (2) the use of a common message model based on Java objects that required services to link to the shared library containing these classes.

We chose to migrate away from this design, and towards a more generic, event-based approach to communications.

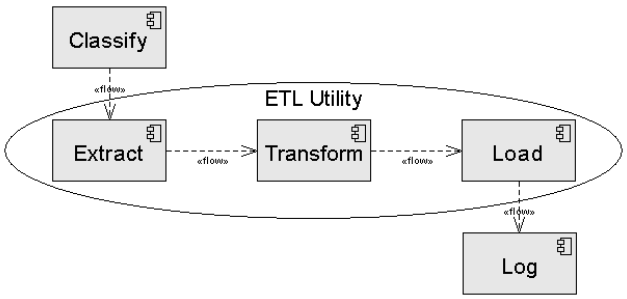

Figure 7. Logical representation of the Extract, Transform, and Load (ETL) Service.

This increased decoupling between services in the system, and placed responsibility for message translation and routing in the communications layer of the system, where it was a more natural fit.

Figure 6 represents one example of an implementation of this design. A service that depends on another service contains (owns) an interface that functions as an event to a remote service. A `java.lang.reflect.Proxy` instance is injected (using Spring) as the implementation of that interface, which use a MessageSender to connect to a remote service (using Mule) or a local service (by injection through Spring), transforming the message if necessary. This is a simple example; these events can be manipulated using the full capabilities of the ESB implementation in question. This has been successful for us in abstracting the service code as much as possible from messaging details such as message routing, format, and content.

4.4 ETL

Metadata enter the system in FITS files (in the FITS header; see §2.3) and are not consistent with the internal storage model. The obvious solution was to extract the information from the FITS headers and transform them into a format compatible with the internal model. However, several requirements serve as a barrier to simple conversion of the FITS header model into the internal model. Examples of such requirements include supporting queries on a single model, adding metadata that do not exist in the FITS headers, and repairing insufficiently formatted data or bad data.

One solution to this problem is to leverage an ETL[5] (extract, transform, and load; Figure 7) system. The concept of ETL has been used in data warehousing for decades. Very simply, it is the extraction of information from a source model, the transformation of that information in a prescribed fashion, and loading the transformed information into a target model. For the NOAO Science Archive effort toward improving imaging metadata quality, the ETL concept is leveraged to provide more advanced transformations.

The most challenging part of the ETL is the *Transform* service. Consideration has been given to devise an extensible system that enables dynamic configuration of the transformation workflow. We again chose Spring, which provides for both ease of use and the ability to independently extend the system, as well as its inherent capability to dynamically load the workflows.

The *Transform* workflow (Figure 8) involves three main components, namely the *Get Transform* Service, the *Execute Transform* Service, and the Process library (namely the *Process Manager*). *Get Transform* Service is a Data-centric Basic service that provides a Spring configuration file containing the declaration of the *Transform* implementation and its associated metadata. These *Transform Configurations* are dynamically generated by operational users, through the graphical user interface (previously mentioned).

Execute Transform Service enables synchronous interaction with the *Transform* after execution by the *Process Manager* (i.e., to obtain the result). The *Process Manager* provides asynchronous, thread-managed, unresponsive execution of runnable components.

Execute Transform is initialized through the Spring container, the configuration of which also contains abstract bean references that are the atomic transformations which are referenced and configured in the specific transformation workflows. In essence, the base Spring configuration contains definitions of generally usable components; these components are referenced by other Spring configuration files (specific to the data source); these 'other' configuration files are loaded dynamically into the container, and load the base components from the parent context.

5. Conclusions and Outlook

The NOAO Science Archive System design is known to be theoretically easily evolved as new features are identified and clarified. Fortunately, we've found that the service oriented nature of the system does enable, in practice, the evolution we had hoped for as we successfully incremented the system to add a new subsystem for the quality improvement of imaging metadata.

This new subsystem begins the federation of our 'system' metadata by providing a new node for managing the new metadata objects introduced along with adding new basic, data centric services for the management of these new metadata types.

Additionally, it adds new basic and process services that perform the actual classification, transformation, and validation of the imaging metadata along with services that provide support activities and orchestration.

This effort also helped us identify a process for introducing new features and illustrated some less than optimal choices we had previously made about communication that created too much coupling and so, inhibited system evolution.

Ultimately, the project is deemed a success in both providing the framework for the incremental quality improve-

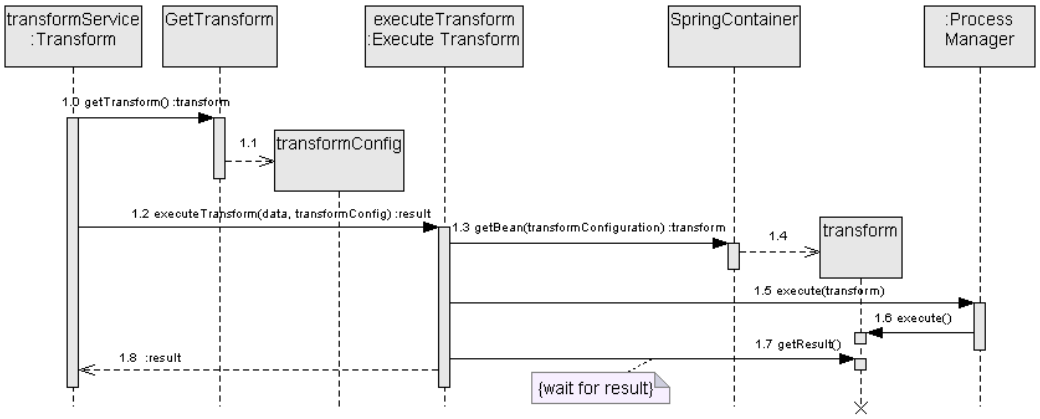

Figure 8. Sequence describing the execution of a transformation.

ment of the imaging metadata and also for helping us to establish an effective process for future, incremental feature additions to the NOAO Archive System.

As the system grows, concerns for discovery and service lifecycle management will become critical. The use of deployment bundles positions us to leverage OSGi implementations, such as Spring Dynamic Modules and Service Component Architecture, and address scalability and distributability of the components of the system as these issues move to the forefront. In support of these concerns, and to be prepared for leveraging new frameworks that are more appropriate for some components in our system (see next paragraph), we will continue to investigate improvements to migrate functionality that is more related to message routing and transformation to the communications layer of the system. This will allow our service code containing our business logic to be applied in ways we are only beginning to envision today.

Other future exploration includes consideration of grid-based data and information storage. While this will certainly impact the storage of the FITS files currently kept within a distributed file system, it likely will also have a great impact on the way the current database content is collected, stored, and retrieved and also on the way transformations are executed. Under consideration now is the Hadoop software platform. This platform will enable the movement of the remediation logic to the data itself and so, help improve the data consistency and the performance of the framework.

Acknowledgments

We are grateful to the entire NOAO Archive Team and also to the Operations and Science Stakeholders who provided their valuable expertise. We would also like to thank the ACM rehearsal staff for their assistance in our preparation. A very special thank you goes to our shepherd, Einar Landre, for his unwavering dedication and encouragement.

References

[1] D. C. A. Bultermann. Is It Time For a Moratorium on Metadata? *IEEE MultiMedia*, October-November 2004.

[2] FITS: The Astronomical Image and Table Format. *http://fits.gsfc.nasa.gov/*.

[3] J. Gable. "Enterprise application integration". *Information Management Journal*, 36(2), March,April 2002.

[4] International Virtual Observatory Alliance. *http://www.ivoa.net*.

[5] R. Kimball and et al. *"The Data Warehouse Lifecycle Toolkit"*. Wiley, 1998.

[6] S. Lowry and P. Warner. NOAO DMaSS Solutions Platform: An Integrated Approach to Services. In *Astronomical Data Analysis Software and Systems XVI*, volume 376, pages 135–138, 2007.

[7] J. McGovern, O. Sims, A. Jain, and M. Little. *Enterprise Service Oriented Architecture: Concepts, Challenges, Recommendations*. Springer, 2006.

[8] B. Meyer. *Object-Oriented Software Construction, 2nd ed.* Prentice Hall, 1997.

[9] Mule Open Source ESB and Integration Platform. *http://www.mulesource.org*.

[10] M. Oberhofer and A. Dreibelbis. *An Introduction to the Master Data Management Reference Architecture*. IBM Press, 2008.

[11] R. W. Schulte and Y. V. Natis. 'Service Oriented' Architectures, Part 1. *[Report] Gartner*.

[12] R. W. Schulte and Y. V. Natis. 'Service Oriented' Architectures, Part 2. *[Report] Gartner*.

[13] R. Smith, T. Boroson, and R. Seaman. The NOAO Data Products Program: Developing an End-to-End Data Management System in Support of the Virtual Obser-

vatory. In *Astronomical Data Analysis Software and Systems XVI*, volume 376, pages 707–710, 2007.

[14] R. C. Smith, M. Dickinson, S. J. Lowry, C. J. Miller, M. Trueblood, and F. Valdez. The NOAO End-to-End Data Management System: An Overview. In *Astronomical Data Analysis Software and Systems XVI*, volume 376, pages 615–618, 2007.

[15] Spring Framework. *http://www.springframework.org*.

[16] C. Szyperski. *Component Software: Beyond Object-Oriented Programming, 2nd ed.* Addison-Wesley Professional, Boston 2002, 2002.

[17] G. V. Tozer. *Metadata Management for Information Control and Business Success*. Artech House, 1999.

[18] D. C. Wells, E. W. Greisen, and R. H. Harten. FITS - a Flexible Image Transport System. *Astronomy & Astrophysics Supp. Ser.*, 44:363, 1981.