Publication Year	2015
Acceptance in OA@INAF	2020-03-07T11:10:08Z
Title	Learning from FITS: Limitations in use in modern astronomical research
Authors	Thomas, B.; Jenness, T.; Economou, F.; Greenfield, P.; Hirst, P.; et al.
DOI	10.1016/j.ascom.2015.01.009
Handle	http://hdl.handle.net/20.500.12386/23160
Journal	ASTRONOMY AND COMPUTING
Number	12

Learning from FITS: Limitations in use in modern astronomical research

Brian Thomas[a,*], Tim Jenness[b], Frossie Economou[a], Perry Greenfield[c], Paul Hirst[d], David S. Berry[e], Erik Bray[c], Norman Gray[f], Demitri Muna[g], James Turner[h], Miguel de Val-Borro[i], Juande Santander-Vela[j,k], David Shupe[l], John Good[l], G. Bruce Berriman[l], Slava Kitaeff[m], Jonathan Fay[n], Omar Laurino[o], Anastasia Alexov[c], Walter Landry[l], Joe Masters[p], Adam Brazier[b], Reinhold Schaaf[q], Kevin Edwards[r], Russell O. Redman[e], Thomas R. Marsh[s], Ole Streicher[t], Pat Norris[a], Sergio Pascual[u], Matthew Davie[v], Michael Droettboom[c], Thomas Robitaille[w], Riccardo Campana[x], Alex Hagen[y], Paul Hartogh[z], Dominik Klaes[q], Matthew W. Craig[aa], Derek Homeier[ab]

[a]*Science Data Management, National Optical Astronomy Observatory, 950 N Cherry Ave, Tucson, AZ 85719, USA*
[b]*Department of Astronomy, Cornell University, Ithaca, NY 14853, USA*
[c]*Space Telescope Science Institute, 3700 San Martin Drive, Baltimore, MD 21218, USA*
[d]*Gemini Observatory, 670 N. A'ohōkū Place, Hilo, HI 96720, USA*
[e]*Joint Astronomy Centre, 660 N. A'ohōkū Place, Hilo, HI 96720, USA*
[f]*SUPA School of Physics & Astronomy, University of Glasgow, Glasgow, G12 8QQ, UK*
[g]*Department of Astronomy, The Ohio State University, Columbus, OH 43210, USA*
[h]*Gemini Observatory, Casilla 603, La Serena, Chile*
[i]*Department of Astrophysical Sciences, Princeton University, Princeton, NJ 08544, USA*
[j]*Instituto de Astrofísica de Andalucía, Glorieta de la Astronomía s/n, E-18008, Granada, Spain*
[k]*Square Kilometre Array Organisation, Jodrell Bank Observatory, Lower Withington, Macclesfield SK11 9DL, UK*
[l]*Infrared Processing and Analysis Center, Caltech, Pasadena, CA 91125, USA*
[m]*International Centre for Radio Astronomy Research, M468, 35 Stirling Hwy, Crawley, Perth WA 6009, Australia*
[n]*Microsoft Research, 14820 NE 36th Street, Redmond, WA 98052, USA*
[o]*Smithsonian Astrophysical Observatory, 60 Garden Street, Cambridge, MA 02138, USA*
[p]*National Radio Astronomy Observatory, 520 Edgemont Road, Charlottesville, VA 22903, USA*
[q]*Argelander-Institut für Astronomie, Universität Bonn, Auf dem Hügel 71, 53121 Bonn, Germany*
[r]*Department of Physics, University of Waterloo, Waterloo, ON N2L 3G1, Canada*
[s]*Department of Physics, University of Warwick, Coventry CV4 7AL, UK*
[t]*Leibniz-Institut für Astrophysik Potsdam (AIP), An der Sternwarte 16, 14482 Potsdam, Germany*
[u]*Departamento de Astrofísica, Universidad Complutense de Madrid, 28040, Madrid, Spain*
[v]*Department of Astrophysics, School of Physics, University of New South Wales, Sydney, NSW 2052, Australia*
[w]*Max-Planck-Institut für Astronomie, Königstuhl 17, 69117 Heidelberg, Germany*
[x]*Institute for Space Astrophysics and Cosmic Physics, Via Piero Gobetti 101, Bologna, I-40129, Italy*
[y]*Dept. of Astronomy and Astrophysics, The Pennsylvania State University, 525 Davey Lab, University Park, PA 16802, USA*
[z]*Max-Planck-Institut für Sonnensystemforschung, Justus-von-Liebig-Weg 3, 37077 Göttingen, Germany*
[aa]*Department of Physics and Astronomy, Minnesota State University Moorhead, 1104 7th Ave. S., Moorhead, MN 56563, USA*
[ab]*Centre de Recherche Astrophysique de Lyon, UMR 5574, CNRS, Université de Lyon, ENS Lyon, 46 Allée d'Italie, 69364 Lyon Cedex 07, France*

arXiv:1502.00996v2 [astro-ph.IM] 10 Feb 2015

Abstract

The Flexible Image Transport System (FITS) standard has been a great boon to astronomy, allowing observatories, scientists and the public to exchange astronomical information easily. The FITS standard, however, is showing its age. Developed in the late 1970s, the FITS authors made a number of implementation choices that, while common at the time, are now seen to limit its utility with modern data. The authors of the FITS standard could not anticipate the challenges which we are facing today in astronomical computing. Difficulties we now face include, but are not limited to, addressing the need to handle an expanded range of specialized data product types (data models), being more conducive to the networked exchange and storage of data, handling very large datasets, and capturing significantly more complex metadata and data relationships.

There are members of the community today who find some or all of these limitations unworkable, and have decided to move ahead with storing data in other formats. If this fragmentation continues, we risk abandoning the advantages of broad interoperability, and ready archivability, that the FITS format provides for astronomy. In this paper we detail some selected important problems which exist within the FITS standard today. These problems may provide insight into deeper underlying issues which reside in the format and we provide a discussion of some lessons learned. It is not our intention here to prescribe specific remedies to these issues; rather, it is to call attention of the FITS and greater astronomical computing communities to these problems in the hope that it will spur action to address them.

Keywords:
FITS, File formats, Standards

1. Introduction

The Flexible Image Transport System standard (FITS; Wells and Greisen 1979; Greisen et al. 1980; Wells et al. 1981; Greisen and Harten 1981 and Hanisch et al. 2001; and more recently, the definition of the version 3.0 FITS standard by Pence et al. 2010) has been a fundamental part of astronomical computing for a significant part of the past four decades. The FITS format became the central means to store and exchange astronomical data, and because of hard work by the FITS community it has become a relatively easy exercise for application writers, archivists, and end user scientists to interchange data and work productively on many computational astronomy problems. The success of FITS is such that it has even spread to other domains such as medical imaging and digitizing manuscripts in the Vatican Library (West and Cameron, 2006; Allegrezza, 2012).

Although there have been some significant changes, the FITS standard has evolved very slowly since its genesis in the late 1970s. New types of metadata conventions such as World Coordinate System (WCS; Greisen and Calabretta, 2002; Calabretta and Greisen, 2002; Greisen et al., 2006) representation and data serializations such as variable length binary tables (Cotton et al., 1995) have been added. Nevertheless, these changes have not been sufficient to match the greater evolution in astronomical research over the same period of time.

Astronomical research now goes beyond the paradigm of a set of observational data being analyzed only by the scientific team who proposed or collected it. The community routinely combines original observations, theoretical calculations, observations from others, and data from archives on the internet in order to obtain better and wider ranging scientific results. A wide variety of research projects now involve many diverse datasets from a range of sources. Instruments in astronomy now produce several orders of magnitude larger datasets than were common at the time FITS was born, in some cases requiring parallelized, distributed storage systems to provide adequate data rates (Alexov et al., 2012).

Astronomers have increasingly come to rely on others to write software programs to help process and analyze their data. Common libraries, analysis environments, pipeline processed data, applications and services provided by third parties form a crucial foundation for many astronomers' toolboxes. All of this requires that the interchange of data between different tools needs to be as automated as possible, and that complex data models and metadata used in processing are maintained and understood through the interchange.

These changes in research practices pose new challenges for the 21st century. We must address the need to handle an expanded range of specialized data product types and models, be more conducive to the distributed exchange and storage of data, handle very large datasets and provide a means to capture significantly more complex metadata and data relationships.

A summary of these significant problems within the FITS standard was presented in Thomas et al. (2014). Already some of these limitations have caused members of the community to seek more capable storage formats, both in the past, such as the Starlink Hierarchical Data System (HDS; Disney and Wallace, 1982; Jenness, 2015), the eXtensible Data Format (XDF; Shaya et al., 2001), FITSML (Thomas et al., 2001) and HDX (Giaretta et al., 2003); and in the present and future (e.g., HDF5 (Anderson et al., 2011) and NDF (Jenness et al., 2015, ascl:1411.023)). There are other popular file formats among the radio and (sub-)millimeter astronomy community such as the Continuum and Line Analysis Single-dish Software (CLASS) data format associated with the Grenoble Image and Line Data Analysis Software (GILDAS) tools (ascl:1305.010). Although this file format does not have a public specification, there are open-source spectroscopic software packages like `PySpecKit` (ascl:1109.001) that support certain versions of the data format. Given the large amount of available storage formats, there is certainly a possibility that the use of FITS will fall in favor of other scientific data formats should it not adapt to these new challenges.

The strengths of FITS are well known and include an easily understood serialization, a plethora of stable supporting software, good documentation of the format and the simple fact that it remains to this day the *lingua franca* of astronomical data format exchange. What we feel has been missing is an attempt within the community to dispassionately discuss and understand FITS in terms of problems in its application to modern astronomical research. In this paper we hope to show that technologies and research techniques in astronomy have evolved but FITS has not kept pace. As a result, gaps between FITS utility and the needs of the research community have opened up and widened over time. It is our intended goal in this paper to highlight some selected, important, problems which exist in the FITS core standard today. We have deliberately avoided proposing solutions to the problems we discuss, and we remain agnostic (because the authors are divided) on whether replacing FITS is an obviously good or an obviously bad idea.

We present our argument in the following manner. The various issues have been grouped under the general topics "information interchange" (section 2), "data models" (section 3), "metadata and data representation" (section 4) and "large and/or distributed datasets" (section 5). We address each of these topics in turn below then try to provide an analysis of any deeper causes or "lessons learned" (section 6). A summary section ends the paper and provides an overview of our work and future direction.

2. Information Interchange

FITS originated as a delivery format for observatory data. It was the format of choice when transporting data between different data reduction environments such as IRAF (ascl:9911.002), Starlink (ascl:1110.012), AIPS (ascl:9911.003) and MIDAS (ascl:1302.017).

In principle, FITS promotes interchange through its simple and easily understood format which holds its information in various levels of groupings of metadata and data blocks. Metadata are captured via key-value pairs which are in turn grouped

*Corresponding author
Email address: `bthomas@noao.edu` (Brian Thomas)

into FITS headers. The first header is denoted as the 'primary' header and subsequent headers known as 'extensions'. Headers may or may not be then grouped with data blocks. An example primary FITS header appears in Fig. 1.

This simple arrangement of information can satisfy many use cases for transport, however, requirements for interchange have evolved. Effective interchange, as we shall illustrate, now includes things like the ability to declare models for use in higher level processing, validation of models within the file and, at the most basic level, the ability to declare which version of the serialization is being used.

These capabilities have been explored and implemented in several other data formats in astronomy. The Astronomical Data Center (ADC) XDF format, the Low-Frequency Array for Radio Astronomy (LOFAR) HDF5 data model (Alexov et al., 2012), CASA measurement sets (Petry and CASA Development Team, 2012), RPFITS[1] from the Australian Telescope National Facility and Starlink's NDF (Currie, 1988; Warren-Smith and Wallace, 1993; Economou et al., 2014) all serve as examples in this regard.

XDF was created primarily to support archiving, web-based use of published astronomical data and the development of FITSML – an XML version of the FITS data model which could use an XML schema for validation. NDF was developed in the late 1980s as a means of organizing the hierarchical structures that were available via the Starlink HDS format when it became apparent that arbitrary hierarchies could lead to chaos and lack of ability for applications to interoperate (Jenness et al., 2015). HDX (Giaretta et al., 2003) was developed around 2002 as a flexible way of layering high-level data structures, presented as a virtual XML Document Object Model (DOM), atop otherwise unstructured external data stores; this was in turn used to develop Starlink's NDX framework, which (among other things) allowed FITS files to be viewed and manipulated using the concepts of the NDF format. HDF5 (Alexov et al., 2012) was chosen to accommodate LOFAR's exceptional high data rates, 6-dimensional data complexity, distributed data processing and I/O parallelization needs.

2.1. *Format versioning*

There is no standard means for a FITS file to communicate the formatting version it conforms to. Consider the example primary header in Fig. 1: the only keyword which implies any type of format is SIMPLE which is set to 'T', or true. The comment indicates that the file conforms to "Standard FITS format", but what indeed is that 'Standard'?

The designers and maintainers of FITS have espoused the principle "once FITS, forever FITS" (see e.g., Grosbol et al., 1988; Hanisch et al., 1993). Certainly some in the community see this as a strength for the format as it appears to promote long term stability and "archivability" of FITS data (Allegrezza, 2012; Library of Congress, 2012). This is not, however, quite the same thing as saying that FITS is unversioned. There have been at least three named descriptions of FITS. These include the first, or 'basic FITS' document (Wells and Greisen, 1979; Wells et al., 1981), the NOST version of FITS (Hanisch et al., 2001), and the current version 3.0 (Pence et al., 2010). One can regard these as successive improvements of a document describing changing best practices for an unchanging format (compare "the value [of the putative FITS version keyword] is always 1.0 by default" in Wells (1997), which discusses this general point in some depth). However the fact remains that there are features in the most recent FITS description (such as 64-bit integers, negative `BITPIX` values, FITS extensions and tables) which were not present in the first FITS version and demonstrably FITS has evolved.

The "once FITS, forever FITS" doctrine may be taken to require backward and forward compatibility or, if you will, compatibility with all FITS files ever created in the case where there is only one version ever. Either way, backward compatibility means that it always *should* be feasible to use the most recent FITS reader. For forward compatibility, at minimum, reasonable expectation goes beyond requiring a FITS reader not crash when confronted with a newer FITS file; it should do more than this. Ideally, it should parse what parts of the file are still compliant with its understanding of the format and report on those parts/features of the file which it does not recognize. In either compatibility case, without unambiguous version metadata, readers have to rely on 'duck-typing' [2] and heuristics which are ultimately error prone because it requires the implementer of the parser to perfectly interpret the signature of any particular set of features present in the given FITS instance from among other possible features which are absent. Furthermore, as the format evolves beyond the date of its creation, the software cannot know how that signature may change and may incorrectly identify the version, a clear difficulty for forward compatibility. The reliance on heuristics also has impact beyond writing a FITS parser. Future archivists will certainly want to know what version of the format they are dealing with without having to guess from ancillary evidence such as the presence of certain keywords, date of the file creation and so on.

The lack of versioning also limits the ability of our community to move forward constructively with developing new FITS versions. The "once FITS, forever FITS" doctrine requires we accrete more and more "design rules" which may limit our ability to implement new and needed features and clutter reader code. Consider that three keywords have been deprecated (`BLOCKED`, `CROTA2` and `EPOCH`) by the latest version of FITS. Per the standard, these are "obsolete structures that should not be used in new FITS files but which shall remain valid indefinitely". As such, software writers must indefinitely be on guard for these metadata and writers of new conventions must avoid utilizing these specific keywords. As time passes and changes of this nature accumulate, it will be progressively harder to interpret FITS data correctly and write new conventions.

[1] `http://www.atnf.csiro.au/computing/software/rpfits.html` – RPFITS is an incompatible fork of FITS (see e.g., Barnes, 1998).

[2] see `http://en.wikipedia.org/wiki/Duck_typing`

```
SIMPLE  =                    T / Standard FITS format
BITPIX  =                  -32 / 32 bit IEEE floating point numbers
NAXIS   =                    3 / Number of axes
NAXIS1  =                  800 /
NAXIS2  =                  800 /
NAXIS3  =                    4 /
EXTEND  =                    T / There may be standard extensions
ATODGAIN=             7.000000 / Analog to Digital Gain (Electrons/DN)
RNOISE  =             1.010153 / Readout Noise (DN)
EPOCH   =       49740.82869315 / exposure average time (Modified Julian Date)
EXPTIME =          2500.000000 / exposure duration (seconds)--calculated
EXPO    =          1300.000000 / weighted average initial exposure time
RSDPFILL=                -250 / bad data fill value for calibrated images
SATURATE=               10237 / Data value at which saturation occurs
TEMP    =                   0 / Temperature (0=cold, 1=warm)
FILTNAM1= 'F555W              ' / first filter name
HSTPHOT =                    T / Preprocessed by HSTphot/mask
END
```

Figure 1: Representative simple primary header of a FITS file showing an assortment of FITS keywords and their associated values. This header from 1995 uses a definition of the, now deprecated, `EPOCH` keyword that is at odds with the standard usage of the period but the lack of parsable units for the field make it hard for a computer parser to understand this. Bytes which contain data may or may not follow the `END` keyword of the header.

Although the FITS format is apparently rather simple, on disk, the multiple versions of the format description, and the existence of numerous header conventions, mean that reading a FITS file in full generality is a complicated and messy business. As there is no versioning mechanism to effectively declare deprecated structures finally "illegal", these complications and costs will only increase.

2.2. *Declaration and validation of content meaning*

Related to, but separate from, the lack of versioning of the serialization, is the lack of ability to declare the presence of data models and their associated meaning. By 'data model' we mean:

> "a description of the objects represented by a computer system together with their properties and relationships; these are typically 'real world' objects such as products, suppliers, customers, and orders.[3] "

Of course, objects in astronomy are more likely to involve things like observations, instruments, celestial coordinates and actual astronomical objects such as stars. Likely properties one will encounter in a FITS file include things like observational parameters (start/end times), astronomical coordinates, name and properties of the observing instrumentation, and so forth. In FITS-speak, we can say that any FITS keyword outside those defined in the FITS standard is a data model parameter, and collections of related FITS keywords form a data model. Ideally a data model should be associated with a given, unique, "namespace" so that collisions in naming of the models and requisite parameters are avoided.

Data models can provide a standard by which information (data and metadata) in the file may be semantically and syntactically validated in software. Questions such as "are all of the required metadata/data structures present in the file?" (e.g., all of the needed keywords occur in the correct places in the file) and "are there any non-normative values in the file?" (all metadata/data values are within expected bounds) are both questions answered by syntactic validation, the conformance of information in the file to one or more declared data models. The question of "how do these data (inter)relate with other data" (e.g., can named structures in the file be associated in some manner with others in another file/extension?) is one of semantic validation. By confirming that the file is 'valid' in both senses, we may link the data model to the information in the file, and hence answer the fundamental question "what does this data you gave me represent?" (e.g., lists of stars, tables of galaxies, images of dust clouds, etc). It is important to note that all of these questions are critical to consumers of the file.

There is already evidence that the FITS community values and needs shared data models. There are many examples. WCS and some other FITS conventions such as OIFITS (Thureau et al., 2006), MBFITS (Muders et al., 2006), PSRFITS (Hotan et al., 2004), SDFITS (Garwood, 2000) and FITS-IDI (Greisen, 2011) are data models. The declaration of keyword dictionaries[4] is also essentially an act of declaring one or more data model(s).

Let us also note that it is not unreasonable to expect more than one model to appear within a file. Consider data distributed by the Palomar Transient Factory. For these data to permit the

[3] Definition adopted from Wikipedia, see http://en.wikipedia.org/wiki/Data_model

[4] Some collected data dictionaries with FITS keywords may be seen at the GSFC FITS site, see http://fits.gsfc.nasa.gov/fits_dictionary.html

widest variety of software tools to understand the astrometric distortion in these images, keywords from both the "SIP" and "TPV" conventions are included (Shupe et al., 2012). One convention expresses distortion polynomials in pixel space and the other in intermediate longitude and latitude, yet it is not immediately obvious which data model should be applied.

All of these data models imply an associated "namespace" which is a means of declaring the origin of the data model so that we may disambiguate and/or associate declared properties between models. For example, separate namespaces should exist for the two aforementioned astrometric distortion models in the example above. There are common problems which namespaced models help to solve and even the 'simple' metadata in Fig. 1 illustrates this.

Consider the `TEMP` keyword in the example. Without reading the comment associated with it, we cannot know if this is this a temperature or perhaps some type of temporary file or resource or something else. If it is a temperature then what is this the temperature of? What do the values '0' and '1' mean? Are these the only valid values for this keyword? `TEMP` is a likely keyword string to appear in other files, how do we know if the `TEMP` in the other files is the same one we see in the example?

Clearly, it is a non-trivial matter for the machine to determine whether these are the same properties and to know other important details for using this information. This problem is not isolated to a solitary bit of rogue metadata. We can ask similar questions about most of the keywords in the example header. Namespaced data models help address these issues. With an appropriate namespace mechanism in place, it is possible to create a machine-readable mapping between the data models so that any software program can determine whether `model1:TEMP` is the same (or different) property as `model2:TEMP`. Namespacing mechanisms can both provide humans with documentation, and provide software with the means to look up model definitions (perhaps from remote locations), and thus apply syntactic or semantic validation rules for the information at hand. This will allow the program to answer the remainder of our posed questions above.

These arguments indicate there is a pressing need for namespaced data models, yet, the only way in which we can currently implement them is for a human to inspect the file, or to write special purpose software programs targeted to particular data models. Given the data volumes that we have in astronomy, the latter choice is in the direction we should go, but is not practical in the general case.

The writing of generalized software programs to detect any data models present in a given FITS file is currently a difficult task for many reasons. First and foremost, we must recognize that there are constantly new data models being created and modified. Some of these are documented in a human readable fashion but there are many more models which do not even meet this standard. Worse, due in part to the lack of good validation tools, the community has accepted many informal variants of existing models. These variants may both be documented or not but are a result of either accidental or intentional stretching of the original metadata usage. The header in Fig. 1, for example, is an informal variant because of its non-standard use of the `EPOCH` keyword. Finally, there is the possible complication of more than one data model being fully, or partially, present within a file. Without explicit signposts for the software to use, it is likely impossible to determine which data models are present and map information to appropriate meaning.

3. Data models

One of FITS strengths is that it includes some common data structures which are important in astronomy data. The FITS standard includes such things like "table" and "n-dimensional array" the latter which is used to model both images and data cubes. These items are really simple representations of the data at a primitive level, and are certainly needed for basic access to the information within the file. Even so these structures, by themselves, do not contain much in the way of necessary detail and semantic information which tells the consumer exactly what it is they are actually consuming. For this reason, they cannot be considered to be data models.

The FITS standard does supply a data model, for example the aforementioned WCS may be considered to be part of it, and these standard semantics are generally regarded as another strong point of FITS. The other data formats we have previously mentioned vary in how extensive their core data model is. The range goes from HDF5, which does not supply any data model *per se*, to that of NDF which has very rich metadata in its data model.

It is a matter of opinion as to whether more/richer detailed data models in the format standard are better or not. The NDF core data model metadata are certainly more detailed than the metadata in the FITS standard. On the other hand FITS is certainly more widely adopted than NDF. Nevertheless, we believe FITS would benefit from an expansion in its standard data model as there are certainly common semantics which may be found in other data formats (e.g. NDF, XDF, etc) and FITS-based model extensions (e.g. such as MBFITS or local data dictionaries) which the community can benefit from.

In this section we detail some important missing (component) data models.

3.1. Scientific Errors

The measurement of physical properties with their associated uncertainties is fundamental to astronomical research. It is thus ironic that FITS, which is purposely designed for supporting astronomical research, has no standard data model for capturing information about scientific errors.

We could easily list a great number of possible error types which might be useful but trying to encompass all of the needs of the community at once is likely to create an unwieldy data model. We suggest that the community needs to provide for the most common needs, and target that subset as a first, shared model. Earlier efforts which might inform and help this work include local data models at sites such as CADC (Dowler, 2012) and the error models implemented in other data formats like NDF (although see for example Meyerdierks, 1991), and software efforts underway in scientific programming communities

such as Astropy (Astropy Collaboration, 2013). Each of these has valuable insight into the requirements.

Nevertheless, we can anticipate that the following general characteristics might be part of the model:

- Allow for both metadata and data to have errors.
- Allow for extensible classification of the error type. For example, "Gaussian" errors are also a subclass of "statistical" errors.
- Allow association of more than one error class/type per measurement. For example, allow for both systematic and statistical errors to be associated with each measurement.
- Allow for additional properties to be associated with each error class. For example, "statistical" errors may have an assigned "sigma" value.

3.2. Extended Coordinate Support

The existing FITS WCS data models illustrate some of the limitations associated with FITS. The "once FITS, always FITS" idea required that the current WCS standards were developed as an extension of the older AIPS standard, and so inherited many of the inherent limitations of that system. Even so they took a long time to be agreed. They are complex yet incomplete and inflexible. They are inadequate for many modern telescopes, and restrict creative use of novel coordinate transformations in subsequent data analysis. For instance, raw data must handle more distortion issues than the FITS WCS standard projections can handle. There are some provisions for handling more arbitrary distortions, but they are either cumbersome or too simple. Perhaps the biggest limitation is that different transformations of coordinates cannot be combined in flexible ways. The user is effectively limited to choosing only one of the solutions available.

This is unfortunate. Not only does it reduce the range of transformations that can be described, but it also makes it harder to decompose the total transformation into its component parts thus making understanding and manipulation of the total transformation harder. The alternative approach – a "toolkit"-style system that creates complex transformations by stacking simpler atomic mappings – is usually the most efficient representation as far as data storage is concerned (for example AST, see below).

To illustrate the problem consider the imaging data taken by the Hubble Space Telescope which require multiple distortion components (see e.g., Hack et al., 2013). Some are small but discontinuous. Others are linear but time varying. There is no FITS WCS compatible solution that handles these needs well. As another example, SCUBA-2 raw data (see e.g., Holland et al., 2013) include focal plane distortions which are combined with other transformations but must also support the dynamic insertion of other distortion models when a Fourier transform spectrometer (Gom and Naylor, 2010) is placed in the beam.

Another case with poor support is Integral Field Unit (IFU) data. Many of these datasets have discontinuous WCS models. The only way to support these in FITS now is to explicitly map each pixel to the world coordinates. Besides being space inefficient, it is difficult to manipulate in any simple way.

In addition to limiting the description of raw telescope data, FITS WCS also restricts what can be done with such data during subsequent analysis. There are many potentially interesting transformations that would result in the final WCS being inexpressible using the restrictive FITS model. For instance, transforming an image of an elliptical galaxy into polar or elliptical coordinates is currently not possible. Another case which is unworkable is an alternate coordinate system to an image to represent the pixel coordinates of a second image covering the same part of the sky. These may not be common requirements, but they illustrate the wide range of transformation that should be possible with a flexible WCS system.

The inflexibility in the FITS solution arises from multiple issues, but lack of namespaces is a serious barrier to providing a more flexible solution. If one has multiple model components each with similar parameters, how does one distinguish between them? One may use the letter suffix, but that is also used to distinguish between alternate WCS models. The limitation on keyword sizes presents limitations on how many coefficients can be supported. The lack of any explicit grouping mechanism requires complex conventions on how to relate whole sets of keywords. With more modern structures, such contortions and limitations are not necessary.

The reality is that to solve these problems, many software systems have chosen alternate solutions and save their WCS information in FITS files in other ways (or in separate files). For example, the AST library (Warren-Smith and Berry, 1998; Berry and Jenness, 2012) is not subject to these limitations, but is forced to use non-standard FITS keywords when serializing mappings to FITS files (see Fig. 2).

3.3. History and Provenance

The FITS standard encourages people to store processing history information in the header using a pseudo-comment field named `HISTORY`. This works from the perspective of making the information available to a sufficiently interested human (assuming that each step in the data processing adds information to the end of the history section of the header) but the free-form nature of the entries makes it essentially impossible for a software system to understand what was done to the data. This may be possible within the constraints of a single data reduction environment but it is highly unlikely that the content of the `HISTORY` block can be understood by any other software packages. History needs to be treated as a first-class citizen with a standardized way of registering important information such as the date, the software tool and any relevant arguments or switches.

A related issue is data provenance; that is, sufficient records of how files were created to permit their reproduction. For a given processed data product it is, for example, impossible to determine which data files contributed to the creation of that product. While there is no metadata standard for specifying this information in output files, experimental systems have been developed which, when fully developed, aim to offer programmatic interfaces that will simplify recording prove-

```
PLRLG_A =     5.50788096462284 / Polar longitude (rad.s)
ENDAST_K= 'SphMap  '           / End of object definition
MAPB_A  = '        '           / Second component Mapping
BEGAST_O= 'CmpMap  '           / Compound Mapping
NIN_D   =                    3 / Number of input coordinates
NOUT_B  =                    2 / Number of output coordinates
INVERT_C=                    0 / Mapping not inverted
ISA_I   = 'Mapping '           / Mapping between coordinate systems
INVA_B  =                    1 / First Mapping used in inverse direction
MAPA_C  = '        '           / First component Mapping
BEGAST_P= 'MatrixMap'          / Matrix transformation
NIN_E   =                    3 / Number of input coordinates
INVERT_D=                    1 / Mapping inverted
ISA_J   = 'Mapping '           / Mapping between coordinate systems
M0_A    =    0.426766777415161 / Forward matrix value
M1_A    =    0.699933471661958 / Forward matrix value
M2_A    =    0.572680760059142 / Forward matrix value
M3_A    =   -0.418237169285184 / Forward matrix value
```

Figure 2: Example header of a representation of an AST WCS object in a FITS header when the mapping is too complex to be represented using the FITS-WCS standard.

nance information. One such example is Provenance Aware Service Oriented Architecture (PASOA; Moreau et al., 2008, 2011), an open source architecture already used in fields such as aerospace engineering. In brief, when applications are executed they produce documentation of the process recorded in a repository of provenance records that are persisted in a database. In astronomy, PASOA was successfully demonstrated by integrating it into the Pegasus workflow management system for running the Montage mosaic engine (Groth et al., 2009).

At the JCMT Science Archive (JSA; Gaudet et al., 2008; Economou et al., 2015) data are created with full provenance information using the native provenance tracking that is part of NDF (Jenness et al., 2009). This provenance includes every ancestor along with history information that contributed to each ancestor. When these files are converted to FITS for ingestion into the JSA using the CAOM-2 data model (Redman and Dowler, 2013) the provenance is trimmed to include just the immediate parent files (using `PRVnnnnn` headers) and the observation identifiers of the root ancestor observations (using `OBSnnnnn` headers). The full richness of the provenance information is available in FITS binary tables but the lack of a standard leaves this information hidden from applications other than the ones that created it originally.

Finally, astronomy may benefit from methodologies used to develop provenance systems custom to Earth Science and remote sensing (Tilmes and Fleig, 2008; McCann and Gomes, 2008).

3.4. Data Quality

One of the more pressing needs in our era of shared and distributed data is the need to know which data are "good" or, to put it another way, of sufficient quality. We are long past the era when the data volume was so small that it is practical to download all of the possible data of interest and examine it locally.

Some might insist that this is an easily solved problem. Simply declare a keyword, like `DQUALITY`, and allow it to take a boolean value. To be sure, that example is an exaggeration, but it helps to illustrate that there is no single optimum between the virtue of simplicity and the vice of being simplistic. Data quality cannot be judged on a single, or even a small set, of parameters. The data which are adequate for one type of use, may be wholly inadequate in another usage context. Consider that engineering data generally are unsuitable for science and vice versa. Science data may be unsuitable for other types of science (for example, studies of sky background vs. pointed source science).

A data quality model then, should be an ensemble of common statistical measures of the type of dataset which may be used to derive higher-level judgments of the quality/suitability of the data for some other declared purpose. There are many higher types of data quality models which will need be created from the lower-level measures (image data quality, pointed catalog data quality, etc) and from these particular, targeted, statistical measures data quality may be judged by the dataset consumer without directly examining the data themselves.

3.5. Units

A strength of FITS is that it includes support for units within its core standard. There are, however, limitations in the utility of the provided specification.

First, while it syntactically flexible, there are a few specification ambiguities which could be resolved by an explicit grammar. This limitation has perhaps been one reason that others have felt the need to publish more explicit prescriptions for units (George and Angelini, 1995). Another limitation is that the model does not accommodate the full range of contemporary astronomical data. This is evident from the adoption of other units systems by some major archives such as the CDS

(Centre de Données astronomiques de Strasbourg[5]) which contains a large number of published astronomical tables (see the units section within Ochsenbein 2000). Finally, there is also a provenance issue to defining new units, since – to pick the example of the unit of 'Jupiter radius' – two different groups may prefer the mean or equatorial values for the radius, or in contrast may regard it, not as simply an abbreviation for a certain number of kilometers, but instead as a distance whose value is determined at a certain atmospheric pressure level.

The solution to these limitations is not simply to expand the list of recommended units since, as well as being slow, this fails to distinguish between, for example, different definitions of the second, or to communicate places where the distinction does or does not matter.

The purely syntactical issues surrounding unit strings are being addressed by the IVOA's 'VOUnits' work (Demleitner et al., 2014), but the higher level questions – of communicating and defining new units, of indicating documentation, and of converting between them in a scientifically meaningful manner – are out of scope for that work by design, since experience has shown them to be more contentious than one might expect. These questions should be taken up by the FITS community.

Solutions then may involve cherry-picking VOUnits syntactical fixes and alteration of the units model to use some namespacing mechanism which would help to disambiguate sources of extended units models found within a file. Finally, we believe that an analysis of other more recent work, such as that found in Demleitner et al. and Ochsenbein can help to quickly round out the roster of standard units.

4. Metadata and data representation

What may be construed to be "FITS data" has changed significantly since the founding of FITS. The original FITS specification mandated only the capture of astronomical images. Almost a decade later, FITS extensions (Grosbol et al., 1988), allowing for gathering multiple related data structures in one FITS file, and ASCII tables (Harten et al., 1988) were introduced. Binary tables followed in the next decade (Cotton et al., 1995). Changes have also occurred in metadata capture. Over the intervening years the FITS community has added new metadata conventions such as `HIERARCH` (Wicenec et al., 2009) and `GROUPING` (Jennings et al., 2007; Jennings et al., 1995), which have allowed for greater flexibility in capturing metadata. This expansion in capability to serialize information is to be expected and is all to the good. Nevertheless, as we shall illustrate below, the expansion is still insufficient relative to actual need.

4.1. Rich metadata representation

The basic element of metadata capture in a FITS file is the FITS "card" which comprises a keyword name, a keyword value and a comment. All FITS cards must contain the keyword name and keyword value pair while comments in cards are optional. Comments may sometimes contain information about units of the metadata value.

Metadata may comprise a rich assortment of data structures and single-valued metadata are only the beginning. There is a need to capture sets, lists, vectors and objects within metadata (to name only the most basic of structures). Yet FITS cards, without additional conventions, are only capable of capturing single, scalar keyword-value pairings.

The expression of both objects and non-scalar, multi-valued keywords is difficult in FITS and data model designers have to resort to conventions to achieve this. Object storage is enabled in part by utilizing a hierarchical convention such as `HIERARCH` or the *record-valued* system proposed in the FITS distortion paper (Calabretta et al., in preparation). In order to hold a keyword with either a 'set' or 'list' value, a common local convention adopted is to create a set of keywords sharing the same base name followed by a integer value which may (or may not) indicate order of the values (such as `ICMB001`, `ICMB002`, and so on). Another example is the IRAF *multispec* format (see Valdes, 1993, and references therein) which uses this scheme to specify related world coordinate information (see Fig. 3 for an example). The AST library (Warren-Smith and Berry, 1998, and see also §3.2) takes a similar approach in converting the WCS objects into FITS headers when the transformations are too complex to be represented by standard WCS headers (see Fig. 2).

There are also restrictions on the expression of scalar values in headers. Consider that FITS cards are limited to 80 characters and FITS keyword names may be no longer than 8 characters. The result of these constraints is that keyword values may be no longer than 68 characters. Of course, if you use all of the space for keyword values, then the comment, or keyword values longer than 68 characters will need another convention in order to capture it (such as creating a continuation line in the header using the `CONTINUE` convention (HEASARC FITS Working Group, 2007)).

Let us now consider the impact of keyword name constraints. Not only are keyword names limited to a small set of characters but keyword names are restricted to no more than 8 characters. Often these restrictions prevent clear labelling of the metadata element because authors are forced to map longer, more descriptive, names into the truncated size. Non-English speaking authors are additionally forced to map into the limited character set. If you doubt this leads to problems, try the following experiment: open any non-trivial FITS file and scan the header. Unless you are an expert in the data models present in the file (and sometimes even if you are) it is easy to find that the cramped names of the keywords often leads to arcane and confusing metadata.

These restrictions on the FITS card have impact on conventions with resulting limits on the utility of any implementation. Due to the limited namespace and size of the keywords, different conventions often reuse the same keywords for different purposes. For example, compare the use of `PV` keywords in the products of the SCAMP tools (Bertin, 2006), used for polynomial distortion coefficients, to the more common `PV` keywords used in the WCS convention for generic parameter val-

[5] `http://cds.u-strasbg.fr/`

```
WAT0_001= 'system=multispec'
WAT1_001= 'wtype=multispec label=Wavelength units=Angstroms'
WAT2_001= 'wtype=multispec spec1 = "1 113 2 4955.4428886353510.055675655'
WAT2_002= '83 256 0. 23.22 31.27 1. 0. 2 4 1. 256. 4963.0163112090 5.676'
WAT2_003= '976664 -0.3191636898579552 -0.8169352858733255" spec2 ="2 112'
WAT2_004= '9.09" spec 3 = "3 111 2 5043.5"                              '
```

Figure 3: Example header from an IRAF *multispec* dataset indicating the use of multi-line headers that differs from the `CONTINUE` convention.

ues. These two conventions, when used in the same file, cause ambiguity and incorrect representation of the data. It is true that FITS libraries will often provide a means to choose between one duplicate keyword or another so that in the strictest terms the issue may be resolved. Nevertheless, there is no guarantee that the author of the files intended resolution per the manner of library in use, nor is there any guarantee that different libraries will resolve the matter in the same way so the same file may wind up with different meaning if read by different parsers. Finally, even if libraries were consistent in behavior, the original intention of the author is still ambiguous (for example, they may not want to resolve in favor of one model or the other and both models are important to keep).

4.2. *Expanded missing values support*

Missing values are a common feature of most datasets, and are distinct from invalid values (such as `NaN` or *Not a Number*) that may occur for example in floating point calculations. For images with integer data types, one can make use of the `BLANK` keyword to represent missing values, and for tables with integer and string columns, one can make use of the `TNULL` header keyword. However, for floating point images or table columns, there is no mechanism for specifying missing values. This has led to the common use of `NaN` to represent missing floating point values. However, one should carefully distinguish between true missing values (which in an image could indicate for example an area of sky that was not observed), versus an invalid value (represented by `NaN`) which may represent for example a saturated pixel; such a distinction is not currently possible in FITS.

4.3. *Data associations*

As data acquisition and data reduction systems have become more complex there has been a move to storing multiple image data components in extensions within a single FITS file. The FITS extension mechanism provides a scheme for having multiple images but, as noted in Greisen (2003), in essentially a flat structure without hierarchy or inheritance. If you have nine images in the file there is no way of indicating that three of them are data, three are an error and three are a quality mask. Indeed, there is no way of specifying which triplets are related. You can use the `EXTNAME` header to indicate relationships but this relies on convention and string parsing rather than being a standard part of the format.

As a real world example of this problem, consider the data processing system for the Herschel Space Observatory which includes context products that serve as containers for groups of data products with each product capable of being mapped to a FITS file stored on disk (see the Herschel architecture and design document; Herschel Team, 2008). In particular, Herschel's observational data hierarchy allows all products associated with an observation (telemetry, calibration, raw and processed data) to be linked with the capability of lazy loading of products from the archive "cloud". Satisfying the requirement that all products are storable as FITS files has forced the links in these hierarchies to be specified in a very convoluted form, understandable only within the Herschel interactive processing environment (HIPE; Ott, 2010) and not by other FITS readers.

Another approach to this situation is the conversion of NDF format files to FITS and back to NDF (Currie et al., 2012; Currie, 1997). They demonstrated that you can represent a hierarchical data grouping in the FITS multi-extension format, but this is done using `EXTNAME` conventions combined with headers representing the extension level in the hierarchy and the type of component and so is not understood by other FITS tools.

In both cases above, a standardized way of specifying relationships between extensions would be extremely valuable to data and application interoperability.

4.4. *Declaring byte order*

The original FITS standard specification (Wells et al., 1981) requires that a series of consecutive bytes in multi-byte data items is stored in order of decreasing significance (known as big endian format). Sometimes the byte order needs to be checked and swapped to the opposite byte ordering (little endian format) in systems that do not support non-native data formats. This is the case in some implementations of FITS readers that do not use the CFITSIO library (ascl:1010.001) and which use C routines to implement other scientific capabilities. Programmers on little endian platforms who work with large data volumes may find that this limitation results in a performance penalty as marshaling data to and from the FITS big endian ordering will be required. This is a frequent problem for astronomical programs. Little endianness is found on x86 and x86-64 processors that are commonly used in universities and research laboratories.

The inability to specify the byte order will obviously result in a need to byte swap data. In most cases, this is not a significant problem or impact on performance for modern software systems and can be discounted. There is however, another, more significant issue tied to this limitation. The ability to wrap/-translate existing data products into FITS files, without reprocessing them to the specified byte-order in the FITS standard, is

important. From the perspective of an archivist with the responsibility of preserving the records of astronomical observations, the less the data are altered, the more efficient and reliable the archival data management will be.

4.5. Alternative encodings

The allowed character set for metadata and data in FITS is overly restrictive and is limiting its application. The restrictions between metadata and data do not differ significantly. For metadata, FITS only supports the 7-bit ASCII encoding for keyword values and comments. For data encoding, authors may again only use 7-bit ASCII for text (string) capture in either ASCII or binary FITS tables although the NULL character is allowed in certain cases in binary tables.

The world of astronomy has evolved beyond capturing of scientific information in 7-bit ASCII encoding. The FITS community has grown. Many data are captured by instruments designed, built and run by investigators based in non-English speaking countries and astronomical research has grown significantly elsewhere in the world. Whether as original observational data products, reduced data, information from new services or in capturing theoretical data, FITS is now required to hold data which are not exclusively originating in English-speaking countries.

The current restrictive character set is an anachronism, particularly considering that the most common language on the planet, Mandarin, cannot be used easily in a FITS file. Forcing information into English can easily result in loss of valuable meaning, unnecessarily limit the audience who may use the file, or force the author to use some other format to store their data.

Support for alternative encodings is needed. Simple issues which revolve around the value of keywords like the expression of a person's name (with accents, for example), or the ability to use special scientific and mathematical symbols (like the ångström symbol Å or the degree symbol °) should be handled. Tabular text values should similarly be allowed alternative encodings for the same reasons. Furthermore, while not as critical, the format should also allow keywords themselves to be expressed in a broader range of characters.

5. Large or distributed datasets

At the time when FITS was developed, the primary media used for archiving and transporting the data were tapes. A magnetic tape is unlike a hard-drive in that it is a serial access device. The concept of sequentially accessing the data was naturally adopted for the FITS data model. Although tapes are still widely used for archiving data, such an access mode is no longer commonly available as the files are usually transferred to a hard-drive before being accessed. What is more, the serial nature of FITS has become a significant bottleneck when it comes to working with large datasets.

Consider that many new instruments, especially in radio astronomy (ASKAP; DeBoer et al., 2009, MWA; Tingay et al., 2013, LOFAR; van Haarlem et al., 2013, and SKA; Cornwell and Humphreys, 2010) have been producing, or are planning to produce in the near future, spectral-imaging data-cubes of unprecedented volumes in the order of tens and hundreds of petabytes per year. Due to the increased spatial and frequency resolutions there are individual datasets which can now be expected to be as large as tens of terabytes.

For many reasons which we will detail below, FITS does not provide sufficient support for these types of large data.

5.1. Parallel write/read operations

Large datasets require parallel read/write operations to be processed on parallel computers. FITS, however, cannot support optimization for parallel read/write operations.

This has been the driving factor for LOFAR to invest a significant effort into development of a new format using HDF5 (Alexov et al., 2012). Most of LOFAR's standard data products are now stored using the HDF5 format, as well as HDF5 analogs for traditional radio data structures such as visibility data and spectral image cubes. The HDF5 libraries allow for the construction of distributed files and impose no limits on their sizes. The nature of the HDF5 format further provides the ability to custom design a data encapsulation format, specifying hierarchies, content and attributes.

5.2. Streaming imaging data

While FITS supports cutout capabilities, serving large datasets to an end user requires support for multiple data representations of the same data (e.g. multiple resolutions or fidelities) that may aid in visual exploration of multi-petabyte imaging data. It should also be possible to stream the data progressively to the end-user, displaying an image as soon as the first data become available. Kitaeff et al. (2015) and Peters and Kitaeff (2014) demonstrate the applicability and effectiveness of such approach on radio astronomy imagery.

5.3. Capturing indeterminately sized datasets via streaming

Frequently there is a need to store data from an instrument or remote site that is being transmitted over a network. It is common that when the transfer begins the final size of the dataset is not known. Those using FITS have handled this by writing such data to a file without specifying the size of the last dimension in an image or table, and when the stream is completed, the header is appropriately updated.

Nevertheless, there are applications for which one would like to access all the other information before the file is complete. This may be to integrate the data that are being read out, or to monitor metadata. A library supporting the data format should support such usage.

5.4. Virtual and distributed datasets

When FITS was created, the 'file' (bytes stored on durable physical medium such as spinning disk or magnetic tape) was more or less the only way to store and transfer data. The networked solutions which we enjoy today were absent from the world of astronomy and storage of astronomical data in databases was unusual. Code was run locally by experts and

the results, if shared at all, were usually only reported in published papers. In the intervening years, computer and information technologies have evolved and broadened; we now enjoy many new means of accessing, providing and storing data. FITS should join this revolution.

We should start to consider thinking of FITS as a 'container' of astronomical information which is not necessarily a file. Is there any reason to prevent our FITS 'file' from overlaying a portion of a database? Why not allow FITS to be a wrapper about bytes held within a distributed mass store such as iRODS[6] (see e.g., Rajasekar et al., 2007) or a cloud? Similarly, we would want FITS to contain, and adequately access, data generated by a service (simulation data, for example). More speculatively, FITS could itself execute simple stored algorithms to generate a portion of its data.[7]

These use cases are only examples of where we might go and some may be arguably of limited value. Nevertheless, the generalized use case that may be derived from all of these is certainly of importance: a science data storage format should be able to support both local and remote data access, providing immediate and secure access to data contained within itself, and providing transparent access to data held in non-local entities such as cloud storage, databases, services, and other files.

6. Discussion – Lessons Learned

What then are the lessons we might draw from the above analysis of FITS? Are there any deeper issues and commonalities which thread throughout these issues? In fact, there are several.

6.1. Lesson 1. The format should be versioned

Contrary to the conclusion drawn in Wells (1997) the first lesson that we may draw is that the format needs to be versioned. As we have argued in this paper, we disagree with the premise that FITS has never undergone significant change and hence, there is only one version. Without versioning, it becomes a significantly harder task to write parsers for the format, requiring the software developer to encompass as many design rules as possible in order to robustly handle format instances. As the format evolves and adds new design rules, it only becomes more difficult to write the next parser and, just as bad, older parsers of the format may fail quietly. Ultimately, this experience is contrary to the espoused goal of archivability as one can ultimately never know for certain which permutation of the format the FITS file being read conforms to.

In contrast, when versioning is present, implementers of parsers are able to target a subset of the format design, and declare that within the software so that, should it inadvertently be used on a version it does not understand, it may fail gracefully and in a planned manner.

Because it is so important to understanding the design of the format, versioning metadata should be part of the standard. The choice to implement this as an optional add-on data model (such as a FITS convention) is to be avoided. This is because, without the enforcement of being part of the standard, versioning is unlikely to be implemented where it is needed most, in the generation of new instances.

6.2. Lesson 2. The format should be self-describing

The next lesson that we may draw is that the format needs to be "self-describing" in a machine-readable manner. We consider a self-describing format to be one where the formatted instance is capable of conveying and validating the semantic information it holds where the formatted "instance" may be a file, or a collection of related files or perhaps something more exotic (see above section 5.4).

As we have already seen, FITS lacks semantic validation and its syntactic validation is very limited, achieved only by the creation of hard-coded rules in software utilities such as FITSVERIFY (part of the FTOOLS package; ascl:9912.002) Furthermore, the limitation on keyword length to 8 characters all but guarantees that semantic information within the header is obfuscated. As we have shown, this in part contributes to the problem of being able to detect, and implement, multiple data models within a single FITS file and can lead to the inadvertent creation of informal (and undetectable) variants. Furthermore, without this validation, archiving and interchange of information in the format suffers. It is harder to build robust software systems as any components involved in the interchange of FITS are unable to adequately detect, and handle, invalid files fed to it.

The declaration of validation rules should be flexible. In FITS, where syntactic rules are hard-coded, it is not possible to declare syntactic rules which check the range or data type of metadata fields[8] without re-coding the utility. Ideally, the data format should not rely on hard-coding these rules in software. Rather, a means to capture and associate the data model/namespace information with the contents of the formatted instance, in a machine-readable manner, should be found so that validation can be possible without human inspection or specifically written software programs (similar, perhaps, in the way that JSON or XML formats have schemata).

This approach has additional benefits downstream. First, it will help to avoid misinterpretation because it is better for the creator of the data and/or data model to provide the machine-readable information rather than a downstream programmer. Second, there is a saving in effort in that the model is done once and need not be repeated by numerous downstream programmers. Finally, good validation tools will allow the community to better detect informal variant models and reject them, promoting good practice.

[6] http://irods.org

[7] This probably implies the need for a "FITS language" to generate these data.

[8] Beyond the few canonical keywords which are part of the FITS standard such as GCOUNT or PCOUNT

6.3. Lesson 3. The format should not limit expression of desired data models

FITS was originally designed around a data model which contained a single, generic, two dimensional image and an associated header for metadata. This basic data model has been expanded to allow more instances (extensions), as well as types of data (data cubes and tables). As we discussed in prior sections, working with the basic data model of FITS, authors have implemented their own data models with ever greater demand being placed on the type of data (models) FITS may hold. Holding the line on format changes via the "once FITS, forever FITS" doctrine has been harmful. Original format design decisions have largely been held onto, and have limited the expression of new user data models.

There are two general classes of problem which have held back realizing many needed models. One class concerns the limits created by the format of the serialization itself. Specifically, we mean the limits on metadata representation enumerated in section 4.1 and character encoding in section 4.5. This class causes difficulties in realizing the WCS model, for example.

The other class of problem is that some needed machinery for data modeling within the FITS standard itself is missing. Beyond the aforementioned need to declare models in a machine-readable manner (detailed in Lesson 2), this class encompasses a broader range of issues which include the missing ability to declare byte order, no standard means to make associations between data and metadata, and an inability to create data models which extend both *metadata and the data*. We have already discussed the former two issues in sections 4.4 and 4.3 respectively. The last issue is related to the fact that there is no provision in the standard for extending the existing capture of data itself. For example, if one creates a convention for a new image type which supports multiple representations of the data (section 5.2) it is no longer readable by any FITS parser. Other limitations which arise and/or are unsolvable as a result of this class of problem include no support for optimization of parallel IO (section 5.1), streaming issues (sections 5.2 and 5.3), lack of distributed/virtual data representation and representation of information held in "non-file" instances such as in a database (section 5.4).

It is critical that this data modeling machinery be integral to the format. As we have already noted, in many of the above cases it is possible to create a solution using one or more conventions, but doing so will result in files which are unparsable by downstream readers which do not implement these conventions. In addition, there is no apparent solution, using a FITS convention, than can solve the limitation on the restricted expression of keyword names, nor can one utilize conventions to describe how to serialize *data*.

The data format should do as little to impede the expression of data models. In practice this means having very few hard-coded rules within the format itself such as the 2880 byte record or 80 byte card. Schemata should be capable of describing the layout of *data and metadata* and the format serialization should be flexible enough to handle desired data models and associations between them.

6.4. Lesson 4. Conventions are not standards

As currently envisaged, *conventions* have no path of migration to become part of the FITS standard. There are many useful conventions that provide features that many people use but no-one can guarantee that a particular convention will be supported by a FITS reader. A FITS library cannot simply state that a particular version of the standard is supported but must also state all the conventions that are supported. Multiple conventions exist for continuing long header lines (*multi-spec* and `CONTINUE`) and for supporting hierarchical headers (`HIERARCH` and *record-valued*) but the standard does not have anything to say as to which convention is preferred. Tile compression is rightfully thought of as a success for FITS but again tile compression (e.g., Seaman et al., 2007; Pence et al., 2009) is a convention and not a standard with no guarantee that a particular reader will be able to understand the compression scheme.

CFITSIO, because it is so heavily relied upon by the community to implement FITS reader software, may be considered to be the *de facto* implementation of FITS. This fact, along with the lack of a migration path for conventions, effectively means it is also acting as a *de facto* standards body. CFITSIO supports some conventions but not others. Effectively, the conventions it supports have become mainstream while others have not. This is the wrong process for making worthy conventions widespread. We feel that it is important that the lessons learned from implementing these conventions provide feedback to the standards process to allow the standard to continue to grow and evolve over time.

7. Summary

The limitations which we have described in this paper are significant and we have tried to provide an analysis of their deeper origin. From our investigation, it is clear that FITS suffers from a lack of sufficient evolution. Original design decisions, such as the header byte layout and fixed character encoding made a certain sense at the time FITS was founded. The later enshrinement of the FITS "Once FITS, always FITS" doctrine, which has been utilized to effectively freeze the format, was a mistake in our opinion. Adherence to the doctrine, and lack of any means to version the format in a machine-readable manner, has stifled necessary change of FITS.

More positively, the limitations identified in FITS provide an opportunity to draw a number of important lessons to be learned from the FITS experience. Furthermore, we can use our analysis to identify root causes and turn these into requirements which might be used as goals for future work. For example, some possible requirements might include:

- The data format shall be versioned.
- The data format shall allow for syntactic and semantic validation.
- The data format standard contain provision for declaration of common advanced data structures. These structures include non-scalar values, sets, objects and associations between other metadata and data.

- The data format shall allow declaration of the character set (in both metadata and data).
- The data format shall allow user models to declare new data structures in data[9].

We should not neglect FITS strengths either. In particular, FITS inclusion of semantic content (data models) as part of the standard should continue to be pushed. There are many critical missing data models (section 3) that, if included in the standard, would provide compelling reasons to use a data format.

To be clear, we do not wish to recommend specific corrective action to any particular problem or derived requirement. Instead we hope that action will flow from constructive community discussion including the analysis of other astronomical data formats and any lessons learned from their use and construction. We may anticipate that the form of the possible resolutions to problems in FITS may involve moving existing FITS conventions into the core standard, modification of the FITS standard to remove limitations, or even transferring the FITS data model over into a new serialization.

Our effort will also continue. We plan to extend the work started here in a future paper in which we will gather use cases or "lessons learned" which also show FITS strengths, and gleaning the same from other data formats. From these we plan to extract and publish an overview of the requirements for a modern astronomical data format.

8. Acknowledgments

The authors wish to thank Bill Pence and Rob Seaman for many insightful discussions. We thank Bob Hanisch and Michael Wise for providing helpful comments that improved the contents of this paper.

This research has made use of NASA's Astrophysics Data System.

References

Alexov, A., et al., 2012. Status of LOFAR Data in HDF5 Format, in: Ballester, P., Egret, D., Lorente, N.P.F. (Eds.), Astronomical Data Analysis Software and Systems XXI, volume 461 of *ASP Conf. Ser*. p. 283.

Allegrezza, S., 2012. Flexible Image Transport System: a new standard file format for long-term preservation projects?, in: European Week of Astronomy and Space Science, 1-6 July 2012, Rome.

Anderson, K., Alexov, A., Bähren, L., Grießmeier, J.M., Wise, M., Renting, G.A., 2011. LOFAR and HDF5: Toward a New Radio Data Standard, in: Evans, I.N., Accomazzi, A., Mink, D.J., Rots, A.H. (Eds.), Astronomical Data Analysis Software and Systems XX, volume 442 of *ASP Conf. Ser*. p. 53.

Astropy Collaboration, 2013. Astropy: A community Python package for astronomy. A&A 558, A33. doi:10.1051/0004-6361/201322068, arXiv:1307.6212.

Barnes, D.G., 1998. Realtime, Object-oriented Reduction of Parkes Multibeam Data using AIPS++, in: Albrecht, R., Hook, R.N., Bushouse, H.A. (Eds.), Astronomical Data Analysis Software and Systems VII, volume 145 of *ASP Conf. Ser*. p. 32.

Berry, D.S., Jenness, T., 2012. New Features in AST: A WCS Management and Manipulation Library, in: Ballester, P., Egret, D., Lorente, N.P.F. (Eds.), Astronomical Data Analysis Software and Systems XXI, volume 461 of *ASP Conf. Ser*. p. 825.

Bertin, E., 2006. Automatic Astrometric and Photometric Calibration with SCAMP, in: Gabriel, C., Arviset, C., Ponz, D., Enrique, S. (Eds.), Astronomical Data Analysis Software and Systems XV, volume 351 of *ASP Conf. Ser*. p. 112.

Calabretta, M.R., Greisen, E.W., 2002. Representations of celestial coordinates in FITS. A&A 395, 1077–1122. doi:10.1051/0004-6361:20021327.

Calabretta, M.R., Valdes, F.G., Greisen, E.W., Allen, S.L., in preparation. Representations of distortions in FITS world coordinate systems. A&A http://fits.gsfc.nasa.gov/wcs/dcs_20040422.pdf.

Cornwell, T., Humphreys, B., 2010. SKA exascale software challenges. SKA Memo Series 128.

Cotton, W.D., Tody, D., Pence, W.D., 1995. Binary table extension to FITS. A&AS 113, 159.

Currie, M.J., 1988. Standard data formats. Starlink Bulletin 2, 11.

Currie, M.J., 1997. Data-format conversion. Starlink Bulletin 19, 14.

Currie, M.J., Privett, G.J., Chipperfield, A.J., Berry, D.S., Davenhall, A.C., 2012. CONVERT – A Format-conversion Package. Starlink User Note 55.

DeBoer D. R. et al., 2009. Australian SKA Pathfinder: A High-Dynamic Range Wide-Field of View Survey Telescope. IEEE Proceedings 97, 1507–1521. doi:10.1109/JPROC.2009.2016516.

Demleitner, M., Derriere, S., Gray, N., Louys, M., Ochsenbein, F., 2014. Units in the VO. http://www.ivoa.net/documents/VOUnits/.

Disney, M.J., Wallace, P.T., 1982. STARLINK. QJRAS 23, 485.

Dowler, P., 2012. CAOM-2.0: The Inevitable Evolution of a Data Model, in: Ballester, P., Egret, D., Lorente, N.P.F. (Eds.), Astronomical Data Analysis Software and Systems XXI, volume 461 of *ASP Conf. Ser*. p. 339.

Economou F. et al., 2015. Observatory/data centre partnerships and the VO-centric archive: The JCMT Science Archive experience. Astron. Comp. in press. doi:10.1016/j.ascom.2014.12.005, arXiv:1412.4399.

Economou, F., Jenness, T., Currie, M.J., Berry, D.S., 2014. Advantages of extensible self-described data formats: Lessons learned from NDF, in: Manset, N., Forshay, P. (Eds.), Astronomical Data Analysis Software and Systems XXIII, volume 485 of *ASP Conf. Ser*. p. 355.

Garwood, R.W., 2000. SDFITS: A Standard for Storage and Interchange of Single Dish Data, in: Manset, N., Veillet, C., Crabtree, D. (Eds.), Astronomical Data Analysis Software and Systems IX, volume 216 of *ASP Conf. Ser*. p. 243.

Gaudet, S., Dowler, P., Goliath, S., Redman, R., 2008. The JCMT Science Archive and the Virtual Observatory, in: Argyle, R.W., Bunclark, P.S., Lewis, J.R. (Eds.), Astronomical Data Analysis Software and Systems XVII, volume 394 of *ASP Conf. Ser*. p. 135.

George, I.M., Angelini, L., 1995. Specification of Physical Units within OGIP FITS files. http://heasarc.gsfc.nasa.gov/docs/heasarc/ofwg/docs/general/ogip_93_001/ogip_93_001.html.

Giaretta, D., Taylor, M., Draper, P., Gray, N., McIlwrath, B., 2003. HDX Data Model: FITS, NDF and XML Implementation, in: Payne, H.E., Jedrzejewski, R.I., Hook, R.N. (Eds.), Astronomical Data Analysis Software and Systems XII, volume 295 of *ASP Conf. Ser*. p. 221.

Gom, B., Naylor, D., 2010. Testing results and current status of FTS-2, an imaging Fourier transform spectrometer for SCUBA-2, in: Millimeter, Submillimeter, and Far-Infrared Detectors and Instrumentation for Astronomy V, volume 7741 of *Proc. SPIE*. p. 77412E. doi:10.1117/12.857667.

Greisen, E.W., 2003. FITS: A Remarkable Achievement in Information Exchange. Information Handling in Astronomy - Historical Vistas 285, 71. doi:10.1007/0-306-48080-8_5.

Greisen, E.W., 2011. AIPS Memo 114r: The FITS Interferometry Data Interchange Convention.

Greisen, E.W., Calabretta, M.R., 2002. Representations of world coordinates in FITS. A&A 395, 1061–1075. doi:10.1051/0004-6361:20021326.

Greisen, E.W., Calabretta, M.R., Valdes, F.G., Allen, S.L., 2006. Representations of spectral coordinates in FITS. A&A 446, 747–771. doi:10.1051/0004-6361:20053818.

Greisen, E.W., Harten, R.H., 1981. An Extension of FITS for Groups of Small Arrays of Data. A&AS 44, 371.

Greisen, E.W., Wells, D.C., Harten, R.H., 1980. The FITS Tape Formats: Flexible Image Transport Systems, in: Elliott, D.A. (Ed.), Applications of Digital Image Processing to Astronomy, volume 264 of *Proc. SPIE*. p. 298.

[9] FITS conventions are unable to specify new data structures to capture "data". All data models must use existing n-dimensional array and (binary or ASCII) table data structures

doi:10.1117/12.959819.
Grosbol, P., Harten, R.H., Greisen, E.W., Wells, D.C., 1988. Generalized extensions and blocking factors for FITS. A&AS 73, 359–364.
Groth, P., Deelman, E., Juve, G., Mehta, G., Berriman, B., 2009. Pipeline-Centric Provenance Model, in: Deelman, E., Taylor, I. (Eds.), Proceedings of the 4th Workshop on Workflows in Support of Large-Scale Science, ACM. WORKS '09. p. 4. doi:10.1145/1645164.1645168, arXiv:1005.4457.
Hack, W.J., Dencheva, N., Fruchter, A.S., 2013. DrizzlePac: Managing Multi-component WCS solutions for HST data, in: Friedel, D.N. (Ed.), Astronomical Data Analysis Software and Systems XXII, volume 475 of *ASP Conf. Ser.* p. 49.
Hanisch, R.J., Farris, A., Greisen, E.W., Pence, W.D., Schlesinger, B.M., Teuben, P.J., Thompson, R.W., Warnock, III, A., 2001. Definition of the Flexible Image Transport System (FITS). A&A 376, 359–380. doi:10.1051/0004-6361:20010923.
Hanisch, R.J., Schlesinger, B.M., Brotzman, L.E., Kemper, E., Teuben, P.J., Steenbert, M.E.V., Warren Jr., W.H., White, R.A., 1993. Definition of the Flexible Image Transport System (FITS). NOST 100-1.0, NASA/-Science Office of Standards and Technology, http://fits.gsfc.nasa.gov/standard10/fits_standard10.pdf.
Harten, R.H., Grosbol, P., Greisen, E.W., Wells, D.C., 1988. The FITS tables extension. A&AS 73, 365–372.
HEASARC FITS Working Group, 2007. The CONTINUE Long String Keyword Convention. http://fits.gsfc.nasa.gov/registry/continue/continue.pdf.
Herschel Team, 2008. Herschel Data Processing: Architecture and Design. http://herschel.esac.esa.int/hcss-doc-11.0/load/hcss_drm/ia/doc/add/index.html.
Holland, W.S., et al., 2013. SCUBA-2: the 10,000 pixel bolometer camera on the James Clerk Maxwell Telescope. MNRAS 430, 2513–2533. doi:10.1093/mnras/sts612.
Hotan, A.W., van Straten, W., Manchester, R.N., 2004. PSRCHIVE and PSRFITS: An Open Approach to Radio Pulsar Data Storage and Analysis. PASA 21, 302–309. doi:10.1071/AS04022, arXiv:astro-ph/0404549.
Jenness, T., 2015. Reimplementing the Hierarchical Data System using HDF5. Astron. Comp. in press.
Jenness, T., et al., 2015. Learning from 25 years of the extensible *N*-Dimensional Data Format. Astron. Comp. in press. doi:10.1016/j.ascom.2014.11.001, arXiv:1410.7513.
Jenness, T., Berry, D.S., Cavanagh, B., Currie, M.J., Draper, P.W., Economou, F., 2009. Developments in the Starlink Software Collection, in: Bohlender, D.A., Durand, D., Dowler, P. (Eds.), Astronomical Data Analysis Software and Systems XVIII, volume 411 of *ASP Conf. Ser.* p. 418.
Jennings, D.G., Pence, W.D., Folk, M., 1995. Convert: Bridging the Scientific Data Format Chasm, in: Shaw, R.A., Payne, H.E., Hayes, J.J.E. (Eds.), Astronomical Data Analysis Software and Systems IV, volume 77 of *ASP Conf. Ser.* p. 229.
Jennings, D.G., Pence, W.D., Folk, M., Schlesinger, B., 2007. A Hierarchical Grouping Convention for FITS. http://fits.gsfc.nasa.gov/registry/grouping/grouping.pdf.
Kitaeff, V.V., Cannon, A., Wicenec, A., Taubman, D., 2015. Astronomical imagery: Considerations for a contemporary approach with JPEG2000. Astron. Comp. in press. doi:10.1016/j.ascom.2014.06.002, arXiv:1403.2801.
Library of Congress, 2012. Flexible Image Transport System (FITS), Version 3.0. http://www.digitalpreservation.gov/formats/fdd/fdd000317.shtml.
McCann, M., Gomes, K., 2008. Oceanographic Data Provenance Tracking with the Shore Side Data System, in: Freire, J., Koop, D., Moreau, L. (Eds.), Provenance and Annotation of Data and Processes. Springer-Verlag, pp. 309–322. doi:10.1007/978-3-540-89965-5_30.
Meyerdierks, H., 1991. Error handling applications. Starlink Bulletin 8, 19.
Moreau L. et al., 2011. The Open Provenance Model core specification (v1.1). Future Generation Computer Systems 27, 743–756. doi:10.1016/j.future.2010.07.005.
Moreau, L., Freire, J., Futrelle, J., Mcgrath, R.E., Myers, J., Paulson, P., 2008. The Open Provenance Model: An Overview, in: Freire, J., Koop, D., Moreau, L. (Eds.), Provenance and Annotation of Data and Processes. Springer-Verlag, pp. 323–326. doi:10.1007/978-3-540-89965-5_31.
Muders D. et al., 2006. APECS - the Atacama pathfinder experiment control system. A&A 454, L25–L28. doi:10.1051/0004-6361:20065359.
Ochsenbein, F., 2000. Standard for Documentation of Astronomical Catalogues. http://vizier.u-strasbg.fr/doc/catstd.htx.
Ott, S., 2010. The Herschel Data Processing System – HIPE and Pipelines – Up and Running Since the Start of the Mission, in: Mizumoto, Y., Morita, K.I., Ohishi, M. (Eds.), Astronomical Data Analysis Software and Systems XIX, volume 434 of *ASP Conf. Ser.* p. 139. arXiv:1011.1209.
Pence, W.D., Chiappetti, L., Page, C.G., Shaw, R.A., Stobie, E., 2010. Definition of the Flexible Image Transport System (FITS), version 3.0. A&A 524, A42. doi:10.1051/0004-6361/201015362.
Pence, W.D., Seaman, R., White, R.L., 2009. Lossless Astronomical Image Compression and the Effects of Noise. PASP 121, 414–427. doi:10.1086/599023, arXiv:0903.2140.
Peters, S.M., Kitaeff, V.V., 2014. The impact of JPEG2000 lossy compression on the scientific quality of radio astronomy imagery. Astron. Comp. 6, 41. doi:10.1016/j.ascom.2014.06.003, arXiv:1401.7433.
Petry, D., CASA Development Team, 2012. Analysing ALMA Data with CASA, in: Ballester, P., Egret, D., Lorente, N.P.F. (Eds.), Astronomical Data Analysis Software and Systems XXI, volume 461 of *ASP Conf. Ser.* p. 849. arXiv:1201.3454.
Rajasekar, A., Moore, R., Vernon, F., 2007. iRODS: A Distributed Data Management Cyberinfrastructure for Observatories. AGU Fall Meeting Abstracts , B1214.
Redman, R.O., Dowler, P., 2013. Implementing a Common Database Architecture at the CADC using CAOM-2, in: Friedel, D.N. (Ed.), Astronomical Data Analysis Software and Systems XXII, volume 475 of *ASP Conf. Ser.* p. 159.
Seaman, R., Pence, W., White, R., Dickinson, M., Valdes, F., Zárate, N., 2007. Astronomical Tiled Image Compression: How and Why, in: Shaw, R.A., Hill, F., Bell, D.J. (Eds.), Astronomical Data Analysis Software and Systems XVI, volume 376 of *ASP Conf. Ser.* p. 483.
Shaya, E., Thomas, B., Cheung, C., 2001. Specifics on a XML Data Format for Scientific Data, in: Harnden, Jr., F.R., Primini, F.A., Payne, H.E. (Eds.), Astronomical Data Analysis Software and Systems X, volume 238 of *ASP Conf. Ser.* p. 217.
Shupe, D.L., Laher, R.R., Storrie-Lombardi, L., Surace, J., Grillmair, C., Levitan, D., Sesar, B., 2012. More flexibility in representing geometric distortion in astronomical images, in: Software and Cyberinfrastructure for Astronomy II, volume 8451 of *Proc. SPIE*. p. 84511M. doi:10.1117/12.925460.
Thomas B. et al., 2014. Significant problems in FITS limit its use in modern astronomical research, in: Manset, N., Forshay, P. (Eds.), Astronomical Data Analysis Software and Systems XXIII, volume 485 of *ASP Conf. Ser.* p. 351.
Thomas, B., Shaya, E., Cheung, C., 2001. Converting FITS into XML: Methods and Advantages, in: Harnden, Jr., F.R., Primini, F.A., Payne, H.E. (Eds.), Astronomical Data Analysis Software and Systems X, volume 238 of *ASP Conf. Ser.* p. 487.
Thureau, N.D., Ireland, M., Monnier, J.D., Pedretti, E., 2006. Software tools for optical interferometry, in: Advances in Stellar Interferometry, volume 6268 of *Proc. SPIE*. p. 62683C. doi:10.1117/12.670560.
Tilmes, C., Fleig, A.J., 2008. Provenance Tracking in an Earth Science Data Processing System, in: Freire, J., Koop, D., Moreau, L. (Eds.), Provenance and Annotation of Data and Processes. Springer-Verlag, pp. 221–228. doi:10.1007/978-3-540-89965-5_23.
Tingay S. J. et al., 2013. The Murchison Widefield Array: The Square Kilometre Array Precursor at Low Radio Frequencies. PASA 30, 7. doi:10.1017/pasa.2012.007, arXiv:1206.6945.
Valdes, F., 1993. The IRAF/NOAO Spectral World Coordinate Systems, in: Hanisch, R.J., Brissenden, R.J.V., Barnes, J. (Eds.), Astronomical Data Analysis Software and Systems II, volume 52 of *ASP Conf. Ser.* p. 467.
van Haarlem M. P. et al., 2013. LOFAR: The LOw-Frequency ARray. A&A 556, A2. doi:10.1051/0004-6361/201220873, arXiv:1305.3550.
Warren-Smith, R.F., Berry, D.S., 1998. World Coordinate Systems as Objects, in: Albrecht, R., Hook, R.N., Bushouse, H.A. (Eds.), Astronomical Data Analysis Software and Systems VII, volume 145 of *ASP Conf. Ser.* p. 41.
Warren-Smith, R.F., Wallace, P.T., 1993. The STARLINK Software Collection, in: Hanisch, R.J., Brissenden, R.J.V., Barnes, J. (Eds.), Astronomical Data Analysis Software and Systems II, volume 52 of *ASP Conf. Ser.* p. 229.
Wells, D.C., 1997. Speculations on the Future of FITS, in: Hunt, G., Payne, H. (Eds.), Astronomical Data Analysis Software and Systems VI, volume 125 of *ASP Conf. Ser.* p. 257.
Wells, D.C., Greisen, E.W., 1979. FITS: a flexible image transport system,

in: Sedmak, G., Capaccioli, M., Allen, R.J. (Eds.), Image Processing in Astronomy, p. 445.
Wells, D.C., Greisen, E.W., Harten, R.H., 1981. FITS: a flexible image transport system. A&AS 44, 363–370.
West, J.L., Cameron, I.D., 2006. Using the Medical Image-Processing Package, ImageJ, for Astronomy. JRASC 100, 242.
Wicenec, A., Grosbol, P., Pence, W.D., 2009. The ESO HIERARCH Keyword Convention. http://fits.gsfc.nasa.gov/registry/hierarch_keyword.html.