Measurement Set Version 4 Review Report

PREPARED BY	ORGANIZATION	DATE
Severin Gaudet (Chair)	Canadian Astrophysics Data Center	20 December 2024
Mark Kettenis	Joint Institute for VLBI in Europe	20 December 2024
Ludwig Schwardt	South African Radio Astronomy Observatory	20 December 2024
Mark Wieringa	Commonwealth Scientific and Industrial Research Organisation	20 December 2024
Paul Marganian	National Radio Astronomy Observatory	20 December 2024

Change Record

VER.	AUTHOR	DATE	AFFECTED SECTIONS	REASON
1.0	Review Panel	20 Dec. 2024	all	Initial Version
1.1	Jeff Kern	6 Jan. 2025	all	Formatting and Appendix added

Table of Contents

Glossary

Acronym/ Initialism	Definition
MSv2	Measurement Set version 2
MSv4	Measurement Set version 4

1. Purpose

This document reports the outcome of the design review of the Measurement Set Version 4 Review held on December 12 and 13, 2024. As defined in the Measurement Set version 4 Review Plan, the goal of the review is to evaluate the MSv4 design accomplished by the NRAO and SKA-led collaboration against the following criteria:

1. **Assess Suitability for Radio Astronomy:** Evaluate whether the MSv4 format meets the specific needs and requirements of radio astronomy, ensuring it supports current and future observational techniques.
2. **Evaluate Metadata Completeness:** Determine if MSv4 adequately captures all metadata necessary for processing and confirm that it addresses all critical use cases within the radio astronomy domain.
3. **Examine Design Efficiency and Scalability:** Analyze whether the design of MSv4 supports efficient data operations and scaling, particularly in the context of handling large data volumes and meeting high throughput and low latency requirements.
4. **Review Reference Design Adoption:** Assess the suitability of the MSv4 reference design for adoption by various projects, ensuring it provides a robust framework for implementing and utilizing the MSv4 format effectively.

2. Review Committee

The Review Committee for the MSv4 review is comprised of the members listed below.

Reviewer	Organization
Severin Gaudet (Chair)	Canadian Astrophysics Data Center
Mark Kettenis	Joint Institute for VLBI in Europe
Ludwig Schwardt	South African Radio Astronomy Observatory
Mark Wieringa	Commonwealth Scientific and Industrial Research Organisation
Paul Marganian	National Radio Astronomy Observatory

3. Executive Summary

The review committee is pleased to report that Measurement Set version 4 review outcome is **Satisfactory**. Gaps have been identified, especially for single-dish use, and the team should address those before the model is widely used. The committee makes six key recommendations along with the many findings for each of the charge questions.

3.1 Summary of Findings

The specific questions in the charge to the committee are:

1. ***Assess Suitability for Radio Astronomy:*** *Evaluate whether the MSv4 format meets the specific needs and requirements of radio astronomy, ensuring it supports current and future observational techniques.*
2. ***Evaluate Metadata Completeness:*** *Determine if MSv4 adequately captures all metadata necessary for processing and confirm that it addresses all critical use cases within the radio astronomy domain.*
3. ***Examine Design Efficiency and Scalability:*** *Analyze whether the design of MSv4 supports efficient data operations and scaling, particularly in the context of handling large data volumes and meeting high throughput and low latency requirements.*
4. ***Review Reference Design Adoption:*** *Assess the suitability of the MSv4 reference design for adoption by various projects, ensuring it provides a robust framework for implementing and utilizing the MSv4 format effectively.*

Below the Review Committee responds to each question separately:

1. **Assess Suitability for Radio Astronomy**

   The MSv4 format is suitable for Radio Astronomy. Minor issues have been identified and should be addressed before wider use.

2. **Evaluate Metadata Completeness**
   The metadata included in MSv4 captures nearly all metadata for the interferometry use cases and most of the metadata for the single-dish use case.

3. **Examine Design Efficiency and Scalability**
   The efficiency and scalability has not been demonstrated however it can be inferred from other adopters of underlying technologies that the approach taken in MSv4 could meet the throughput requirement and latency requirement (if not too low).

4. **Review Reference Design Adoption**

   The reference design is suitable for adoption by the community after some minor issues are addressed. If adoption by existing packages is desired, a C++ module to facilitate this would be helpful.

3.2 Review Outcome

The possible review outcomes and the criteria for each are defined below:

- **Satisfactory:** The design is found to be fit for purpose, projects should proceed to utilize the data model. Minor modifications and clarifications as identified and agreed by the review must be completed.
- **Conditional:** Significant issues were found in the design. Data model should only be utilized by projects after the identified issues have been addressed.
- **Unsatisfactory:** The data model as presented is unsuitable for use, a significant redesign is required or an alternative approach should be pursued.

The Review Committee has determined the outcome of the MSv4 review to be: **Satisfactory**. However, gaps have been identified, especially for single-dish use, and the team should address those before the model is widely used.

4. Justification

The committee presents our findings and recommendations according to the charge questions. Note the committee found a significant overlap between charge questions 1 and 2 and thus merged the findings and recommendations into one subsection.

4.1 Suitability for Radio Astronomy and Metadata Completeness

The metadata for the interferometry use cases seems to be mostly complete. The metadata available in MSv2 that is in active use has been carried over into MSv4 and metadata extensions that have been added since the definition of MSv2 (such as various VLBI-related tables) have been included in MSv4.

As identified in RID 295.0, the metadata for the single-dish use case still has some gaps. The review committee recommends finishing the analysis on SDFITS keywords that provide important metadata that should be included in the MSv4 schema and extend this analysis to the new SDHDF data format that has been developed for the single-dish facilities at CSIRO. Also, an improved guide to MSv4 with GBT data should be included with the documentation.

During the review an additional use-case in the form of space-based (VLBI) antennas was brought forward (RID 290.0). While there have been a few space-based antennas in the past and there are several proposals for new missions involving such antennas, there currently isn't an active space-based antenna. We think it makes sense to consider this use-case out of scope until such a mission is at least in the design phase.

The review comments related to scans (RIDs 316.0 and 333.0) suggest that moving from scan **numbers** to scan **names** would improve the usability of the MSv4 schema and is probably desirable.

The MSv4 schema includes some controlled vocabularies, for example for specifying intents and antenna mount types. The review committee recommends creating a lightweight process for extending these vocabularies such that the needs of observatories can be met without creating divergence.

A significant number of review comments point out issues with the documentation of the MSv4 schema (RIDs 297.1, 312.0, 313.0, 322.0, 330.0, 335.0, 337.0, 340.0, 340.3). In their responses the team has agreed to improve the documentation and the review committee encourages making those improvements to the documentation as it will improve the usability of the schema and reduce divergence between data sets from different observatories as a result of different interpretation of the metadata.

The review committee recognizes that the nomenclature used in the MSv4 schema is biased towards the use of parabolic antennas and isn't always appropriate for instruments that are based on aperture array technology. However changing the term "antenna" into "station" (as suggested by 297.2) would have cascading effects throughout the entire schema and would break continuity between MSv2 and MSv4. Therefore the review committee recommends including an explanation how the concepts of "antenna", "feed" and "receptor" used in the MSv4 schema apply to aperture arrays.

It is recommended to apply and document an ordering to the polarizations.

Discussion of RID 336.0 resulted in the the realization that the relationship between antennas and weather stations may be more complicated that what can be expressed in the schema; information that is relevant to a particular antenna may come from multiple weather stations and at the same time a weather station may provide information for multiple antennas (e.g. antennas may each have their own wind speed

measurement, but pressure may be measured by a central weather station).

The suggestion to add a QUALITY data variable to track data quality issues (RID 322.2) resulted in an extended discussion within the review committee. While some of the requested functionality could be handled by adding data groups with suitable FLAG and WEIGHT data variables, the review committee encourages the team to consult the wider community about this suggested feature. The review committee recognizes that it may be difficult to provide a definition of the meaning of quality across observatories.

It is recommended that the performance of flag cubes compared to MSv2 row flagging be evaluated.

4.2 Design Efficiency and Scalability

The review committee pointed out that the current xradio implementation of MSv4 has not yet demonstrated that it is efficient and scalable enough for the end goals of the NRAO and SKAO. However, Zarr, the default underlying storage format of MSv4, has been adopted by other communities with similar data constraints, such as geoscience (the Pangeo project) and brain research (scalable minds), which strongly suggests that it will be sufficient. In addition, the MeerKAT telescope has been successfully operating with a storage solution very similar to Zarr on a medium scale.

The committee agrees that it is good to have a storage layer that is independent of the Xarray API, which would allow Zarr to be swapped out for another option if the need arises, although Zarr support will most likely have to be maintained once bits really start to hit disks.

The Zarr backend also supports both traditional filesystems and object stores, which allows the end users of MSv4 to tailor their pipelines to either option. The Zarr format can be accessed using all the major programming languages found in high-performance pipelines in radio astronomy (at least Python, C++ and Julia), which is another plus.

The review committee recommends more benchmarking of xradio for various access patterns of the data.

There were questions about the chunk size in the storage layer and how to determine it. It would be advantageous to have a single optimal chunk size in bytes for visibilities, weights and flags, which would require a different number of elements per chunk for the three data variables. This could also be affected by compression and the need for rechunking.

It is recommended that the chunking scheme be investigated in order to improve our understanding of efficiency and scalability.

The committee notes that some latency is unavoidable with storage solutions such as object stores, but doesn't consider it a blocker for adoption as latency is typically not as relevant as throughput in the use cases for of MSv4.

4.3 Overview of Reference Design Adoption

The community review identified 20 RIDs related to design adoption. No blockers to adoption remained after review and resolution.

The potentially most serious issue was probably the charge (340.8) that the xradio package had too many dependencies to be used as a stable platform into the future. The resolution stated that while it is true there are many dependencies, most related to plotting features and other non-essential parts and the

reading of an MSv4 only required 5 basic dependencies. The committee was satisfied with this response, but noted that the dependence on code outside our control is indeed a departure from past formats. The proponents of xradio would of course argue that that is exactly the point of the change, we get access to advanced storage and processing solutions developed by the wider community. Another warning (340.13) noted and agreed to was to ensure we avoid the possibility of vendor lock-in of tools and software required to install and run xradio.

There were questions about handling sparse data (322.1) and the possibility of 32-bit complex numbers (304). The response here suggested this will be dealt with using compression during storage, both using built-in Zarr NaN-filling and data compression and in future with a Dysco compression option. There should not be any performance or storage space impacts, but this may need to be demonstrated.

There is a current interoperability issue with C++ Zarr versions because of the use of unicode strings in python (294). The agreed resolution here will be a change to ASCII or UTF-8 strings.

ALMA would like to see direct ASDM read support (340.2). In the response it was noted that while the pyASDM package (currently under development) will allow this, its efficiency cannot be guaranteed.

There were some concerns (315,318,319,320) about the limited feature set of xradio's processing_set, but this will grow with time, especially in the plotting area and the committee has no further concern.

Some telescopes were left out of the initial set of testing data for MSv2 conversion (303), but the team assured us this will be rectified for both interferometer and single dish data and conversion guides will be created. The community is encouraged to submit small test datasets.

In RID 340.14 the point was made that the MSv4 schema and its checker really need to be language independent. It was agreed the schema will be exported to an alternative format (JSON and/or XML) automatically for use outside of python.

The committee found the prototype of WSClean with C++/Zarr reading of an MSv4 written from xradio a nice test of interoperability and this led to the suggestion that a C++ library or casacore module that encapsulated this access to MSv4 would encourage community adoption of the new standard in existing packages. It would also satisfy RID 340.12 which suggests: there should be at least two independent reading methods, they should be simple to use, stable and long lived.

5. Recommendations

In addition to completing the agreed post-review actions from the RID process (see appendix), the Measurement Set Version 4 Review Committee recommends the following actions:

R1. The review committee recommends finishing the analysis on SDFITS keywords that provide important metadata that should be included in the MSv4 schema and extend this analysis to the new SDHDF data format that has been developed for the single-dish facilities at CSIRO.

R2. The review committee recommends creating a process for extending the MSv4 controlled vocabularies such that the needs of observatories can be met without creating divergence.

R3. The review committee recommends adding an explanation how the concepts of "antenna", "feed" and "receptor" used in the MSv4 schema apply to aperture arrays.

R4. The review committee recommends that the performance of flag cubes compared to MSv2 row flagging be evaluated.

R5. The review committee recommends more benchmarking to further evaluate the efficiency and scalability under different access patterns and to understand the trade-offs with chunking.

R6. The review committee recommends that a C++ library or casacore module that encapsulates the access to MSv4 be developed as that would encourage community adoption of the new standard in existing packages.

Appendix A Review Item Summary

RID ID	Status	Name	Summary	Issue Link
339.0	Closed w/ Action	Reference frames	The list of possible reference frames includes the capitalized casacore reference frames, but I think some of those map directly onto the astropy ones (e.g. 'GEO' is the same as 'gcrs')? Is 'icrs' indeed the most sensible choice for the default velocity reference frame? The linked astropy documentation still has a warning that SpectralCoord is experimental (stating that it is new in Astropy v4.1). It is probably not wise to to base MSv4 on something that is expected to change at some point. Maybe clarification should be sought from the astropy developers about this? Also ITRF and ICRS have multiple realizations. The difference might be important for (geodetic) VLBI, although for current astrometric VLBI I don't think there is an expectation amongst users that things are labelled according to the actual realization of the reference frame that was used.	https://github.com/casangi/xradio/issues/339
338.0	Closed w/ Action	SystemCalibrationXds: receptor_label	Specified as int64 in the schema, where I think this should be <U0 like in the GainCurveXds with polarization_type added.	https://github.com/casangi/xradio/issues/338
337.0	Closed w/ Action	WeatherXds: IONOS_ELECTRON	Is this the column density in zenith direction? Or line of sight (i.e. where the antenna is pointing at the time of observation)? I think the former would make this more useful.	https://github.com/casangi/xradio/issues/337
336.0	Closed w/ Action	WeatherXds: STATION_POSITION	If this data variable isn't present, is the position expected to match the associated antenna specified by the antenna_name coordinate? And how does association between station_name and antenna_name work? My guess would be that the association would be done via the position in the last and that this allows for stations not associated with a particular antenna by specifying a shorter list of antennas? Would be good to make this a bit more specific in the documentation.	https://github.com/casangi/xradio/issues/336
335.0	Closed w/ Action	WeatherXds: time_weather	Description talks about time_cal instead of time_weather. Also in PhaseCalibrationXds we have time_phase_cal. Maybe it would make sense to call all of these time_cal?	https://github.com/casangi/xradio/issues/335
333.0	Closed w/ Action	VisibilityXds: polarization_mixed	Using numbers to identify scans is probably workable; after all MSv2 uses numbers as well. However, VEX schedules used for VLBI use a string. Most astronomy observations are scheduled with scan labels that contain a number (i.e. "No0035") but geodetic VLBI uses strings that are basically - (i.e "134-1944"). So if there is an efficient way to assign scan labels instead of just numbers that would probably make it easier to trace back scans to the original schedule.	https://github.com/casangi/xradio/issues/333

332.0	Closed w/ Action	VisibilityXds: xradio_version	Is this really specific for MSv4's created from an MSv2? Probably makes sense for xradio to always create this attribute when it creates an MSv4. Also what should other implementations of the schema do? Add their own attribute? Or would it make sense to have a generic "creator" attribute?	https://github.com/casangi/xradio/issues/332
331.0	Closed	VisibilityXds: scan intents	Intents are likely to be telescope/array/observatory specific. Would it make sense to make the intent vocabulary scoped such that each telescope/array/observatory can defined their own vocabulary? Also the way intents are specified as an attribute seems to imply that an MSv4 can only have a single intent, which implies that a processing set always has to be partitioned by intent? That makes it hard to assign intents at a later stage. The defined vocabulary talks about a .sub_intent attribute, but such an attribute isn't defined in the schema.	https://github.com/casangi/xradio/issues/331
330.0	Closed w/ Action	VisibilityXds: polarization_mixed	I think the schema needs to document how polarization_mixed is supposed to be used. My guess at how this would work is that for a mixed polarization MSv4 one would use ['PP', 'PQ', 'QP', 'QQ'] as the polarization labels and then specify the actual polarization basis for each baseline using labels from the set of all combinations of 'X', 'Y', 'R' and 'L'. Is that correct?	https://github.com/casangi/xradio/issues/330
327.0	Closed w/ Action	Partition info: optional field antenna_name?	Now xradio supports partitioning by antenna for single-dish data. Should we add (optional) antenna_name field to partition info?	https://github.com/casangi/xradio/issues/327
324.0	Closed w/ Action	data_groups management	I really like the concept of data_groups. I have a few considerations on it. I would appreciate it if you could consider them in the future development. Maybe it's better to define reserved group name for common usecases such as "after applycal" or "ready for imaging" Any mechanism to manage chronological evolution of data_groups might be necessary There will be many versions of data_groups especially for data flagging. Now you have only group name to describe the group. But, as number of groups grows, users might need more detailed description on the data group. For example, they might need a field to store short description of the data group, just like the comment field in the flagmanager task. Another option would be to make group name self-descriptive, but it would make the name too long.	https://github.com/casangi/xradio/issues/324
341.0	Closed w/ Action	partition_info attribute doesn't contain information on antenna	For single-dish data, xradio allows to separate MSv4 by antenna. Add antenna information when MSv4 is split by antenna for SD.	https://github.com/casangi/xradio/issues/341
325.0	Closed	selecting visibility within a range of uv-distance?	It is a common exercise to select visibilities within a range of uv-distance (in meter or (k)lambda) for imaging experiments. With the current schema, is it straightforward to do so, for example, main_xds.sel(query='uv_distance > 20 AND uv_distance < 50')? Or it is up to the users/API developers to implement it?	https://github.com/casangi/xradio/issues/325

321.0	Closed	convert_msv2_to_processing_set performance	I did a test with two measurement sets (MSv2) with different file sizes and found that it seems the performance of MSv2 to MSv4 does not scale linearly.	https://github.com/casangi/xradio/issues/321
320.0	Closed	prosessingset plot functions	currently there are only two plot functions (plot_phase_centers(), plot_antenna_positions()) associated with a processing set. Is there a plan to add more plot functions (for example plot uv coverage, plot passband) in the future? Or these are just a demonstration what could be plotted?	https://github.com/casangi/xradio/issues/320
319.0	Closed w/ Action	processingset plot_antenna_positions()	Currently there will be three plots (pairs of xyz) showing up one by one. It might be good to show them using subplot (2x2 grid) for easy data exploration and visualization.	https://github.com/casangi/xradio/issues/319
318.0	Closed	prosessingset summary text output?	It might be good to support an option for export the processing set summary as a text file, like the option in casa-listobs.	https://github.com/casangi/xradio/issues/318
316.0	Closed	SCAN subtable	I think a scan "subtable" (i.e scan_xds) is needed where an association of SCAN_NUMBER to more info that includes the intent, the time interval that the scan was observed and/or any other hardware settings that are specific to this scan (may be combined with intent). Otherwise we already discussed antenna, pointing and field xds.	https://github.com/casangi/xradio/issues/316
315.0	Closed w/ Action	ProcessingSet plot_phase_centers() field labels ignored	plot_phase_centers(): label_all_fields argument is ignored. Side note: would it be useful to users to label antenna names in plot_antenna_positions() (or show in hover)?	https://github.com/casangi/xradio/issues/315
314.0	Closed	Handling of Mosaics in MSv4	As discussed with Jan-Willem, the MSv4 originally was meant to contain data for only one Field such that multi-field MSs would need to be stored in several MSv4 inside one Processing Set. Later the MSv4 was modified to permit more than one Field. So, for mosaics, i.e. observations of the same target with the same array in one execution block sequentially moving over a set of Fields (in the ASDM and MSv2, they all have the same Field Name but not the same Field ID), the questions are: Do the data belonging to the many fields (potentially hundreds) of the same mosaic all get stored in one MSv4 or do they get spread over many MSv4s? What should be the default behaviour? How can a piece of software reading the data recognise that it is dealing with a Mosaic? How can the reading software determine the extend of the Mosaic (total number of fields, spatial extent)? In other words, how is the metadata of the mosaic set in the first place and then made available?	https://github.com/casangi/xradio/issues/314

313.0	Closed w/ Action	Clarifications in the MSv4 description	As discussed also in person with Jan-Willem Steeb, I think the MSv4 is a good step forward, but to make the the transition to it easier for long-term users of the MSv2, it would be good to extend the MSv4 documentation/description further by (a) clarifying the high-level structure of the MSv4 as a database. and (b) providing a "cheat sheet" of corresponding concepts between MSv2 and 4.	https://github.com/casangi/xradio/issues/313
312.0	Closed	Transition from MSv2 to MSv4	I find the Overview documentation of the MSv4 very useful, especially the tables summarising the translation from MSv2 to MSv4 and the key changes, which help me understand the MSv4. As a developer coming from the MSv2 to use the MSv4, I think it would be useful to have some documentation showing main workflows for development using the new MSv4. This could be in Jupyter tutorials or simply a section in the Development section of the documentation. For example, as a CASA developer if I want to use an MSv2 in a test and I apply flags to the MS, for the subsequent test I will unflag the MS to apply different flags later. I believe the grouping will allow me to have several versions of flags, although I didn't quite understand how.	https://github.com/casangi/xradio/issues/312
304.0	Closed	Add a Complex32 datatype	Many radio telescope A2D samplers are 8 bit, e.g., those used by the VLA. This means that the visibility data can be stored in half-float (16-bit) format without loss of precision. Use of half-floats will reduce the storage of such data by about half. It is recommended that half-floats be included as an xarray datatype.	https://github.com/casangi/xradio/issues/304
303.0	Closed w/ Action	GMRT conversion guide	We would like to have a GMRT conversion guide in the "Guides" section. I am attaching a small dataset created by my student Arpan Pal for the same. We would also like to contribute the conversion guide page. Thank you. gmrt.ms.tar.gz	https://github.com/casangi/xradio/issues/303
295.0	Closed w/ Action	GBT data in MSv4		https://github.com/casangi/xradio/issues/295
294.0	Closed w/ Action	Use of string arrays in the schema (especially unicode)	The problem is that arrays with unicode datatypes aren't supported by any of the C/C++ zarr implementations listed here https://zarr.dev/implementations/. So, I propose that null-terminated byte sequences "<S*" should be used in place of unicode "<U*" data types for arrays (there are 59 instances of unicode dtypes in v4.0.0 of the schema).	https://github.com/casangi/xradio/issues/294
290.0	Closed	Space-based antennas.	Extend schema to support space-based antennas.	https://github.com/casangi/xradio/issues/290
322.0	Closed w/ Action	VLBI Support: Mixed Polarization	We need full support for mixed polarization data. I saw that there is a polarization_mixed coordinate, but it is not clear to me if XR, XL, YR etc. are explicitly supported as polarization labels.	https://github.com/casangi/xradio/issues/322
322.1	Closed	VLBI Support: Sparse Matrices	It is not clear to me how sparse data will be supported and if it would be computationally efficient. Example: We have an array with varying number of baselines from scan to scan + different antennas have recorded different bandwidths, so the number of spectral windows also change for different baselines. Will MSv4 allow us to easily combine the data from multiple scans to solve for telescope bandpasses?	https://github.com/casangi/xradio/issues/322

322.2	Closed	Source Intents	It would be nice to have source intents (science target, phase ref, bandpass calibrator etc.) self contained in the MS. These could be set during correlation and it should be possible to later adjust the intents in the MS.	https://github.com/casangi/xradio/issues/322
322.3	Closed	QUALITY Data Variable	A QUALITY (or similar) array that can be used to describe and track data quality issues would be useful to have. It could be similar in structure to the FLAG array, with either integer code for data issues or preferably a string associated with each visibility data point to have more flexibility in tracking data issues and easily describe multiple issues that affect a single data point. The information should propagate through when averaging the data (any issues affecting any data point in the averaging interval also affects the averaged data). Examples of what I have in mind: VLBI scans with linear feed antennas where PolConvert solutions were not good, part of data in a scan that has poor phase coherence, data with low amplitudes that you can likely correct later via self-calibration when imaging the source, edge-channels that you may want to keep when deriving phase calibration solutions but exclude for steps like ACCOR. Generally, this information could allow you to skip certain problematic parts of the data when deriving calibration solutions only for specific calibration steps. During imaging, you could first exclude or down-weight the problematic data -> self-calibrate -> see if the data is now usable or needs to be flagged.	https://github.com/casangi/xradio/issues/322
298.0	Closed	Primary Beam Dataset	Can the MS v4 also include primary beam tables which will be particularly useful when processing cross baselines with two different size antennas?	https://github.com/casangi/xradio/issues/298
298.1	Closed	Storage of Binary Data	Without storing it as an n-dimensional array, is it feasible to store it as a raw binary and then provide a header for the interferometric software to interpret it?	https://github.com/casangi/xradio/issues/298
298.2	Closed	Store Visibilities as Grid	Can we only store the gridded data instead of the whole picture and then run calibration, flagging everything in the uvw plane?	https://github.com/casangi/xradio/issues/298
298.3	Closed	UVW Optional	I feel the uvw keyword is unnecessary as it can be calculated immediately. But I do not have any idea if it makes more sense to calculate it once, store it, and then read it again and again or calculate it every time when needed. I think the same goes for the weights, which can be calculated while gridding.	https://github.com/casangi/xradio/issues/298
298.4	Closed	Observatory Flags	I think the MS should include a section specifically for observatories to populate with timestamps, antennas, or stations that are unavailable or have known corruption issues throughout the observation. Based on this information, we could automatically delete portions of data known to be corrupted, saving significant storage space. This would also improve detection of faint and moderate RFI outliers, allowing faster and more efficient processing.	https://github.com/casangi/xradio/issues/298
297.0	Closed	On disk format	Will there be a preferred file format associated with MSv4? While xarray supports many formats (zarr, netcdf, CSV), but having an intended format may help with support, optimizing performance, and managing user expectations. That said, it is nice that the storage format is not dictating the data model.	https://github.com/casangi/xradio/issues/297

297.1	Closed	baseline_id	baseline_id - Why assign a number, not a pair of numbers (i.e. tuple/MultiIndex)? - If IDs have to be greater than zero, should use uint instead of int - Relatedly, make sure the baseline convention is clear: i.e. if the baseline is ANT1 - ANT2 or ANT2 - ANT1. - If storing as an array of two 32-bit numbers, it can be converted into a single 64-bit number using a view e.g.	https://github.com/casangi/xradio/issues/297
297.2	Closed	antenna_xds naming conventions	Re ANTENNA_DISH_DIAMETER, ANTENNA_POSITION and ANTENNA_RECEPTOR_ANGLE: from a semantic perspective, these names aren't a good fit for aperture arrays (like SKA-Low). Would STATION_DIAMETER etc be reasonable (and not too abstract/esoteric)?	https://github.com/casangi/xradio/issues/297
297.3	Closed	UVW Optional	UVW - For storage, could these be optional? These are computed from antenna positions + phase center, so I've never been convinced these should be taking up precious disk space. It may even be quicker to recompute from antenna positions than to load from disk (particularly for cold storage).	https://github.com/casangi/xradio/issues/297
297.4	Closed	WEIGHT Optional	WEIGHT - Can these be optional, presumed to be zero if missing? (Again for storage space). Alternatively, consider compression algorithms for storage that will squish this down to nothing if they are all zeros.	https://github.com/casangi/xradio/issues/297
297.5	Closed w/ Action	Polarization Coordinate ordering	Default order of polarization products: I recall that UVFITS expects data to be in XX, YY, XY, YX (for CRPIX -5), whereas a flattened Jones matrix would be XX XY YX YY. I prefer the latter!	https://github.com/casangi/xradio/issues/297
340.0	Closed w/ Action	MSv4 Content as subset of ASDM	The documentation says: "The current MS v4 schema focuses on offline processing capabilities and does not encompass all information present in the ASDM." - It would be useful to list all the aspects of the ASDM that are not (yet?) covered by the MSv4. - If there is work on "future expansion to incorporate additional data", would that be mere additions or is there a risk that more fundamental changes to the MSv4 would be needed?	https://github.com/casangi/xradio/issues/340

340.1	Closed	MSv4 Backwards Compatibility	It is clear that the MSv4 is not backwards compatible with MSv2/3. What about the backwards compatibility of MSv4 itself? - Will all MSv4 be backwards compatible with all other future MSv4? - I.e. does a paradigm like 'Once FITS always FITS' also apply to MSv4? - I.e. MSv4.X.X would only contain additions, no changes of existing concepts, nor stucture - If not: this will reduce the longetivity of the data stored in MSv4 massively. (See also below) if yes: this guarantee should be prominetly spelled out and it should be made clear which organizational structure will guarantee the back/forwards compatibility of MSv4 itself.	https://github.com/casangi/xradio/issues/340
340.2	Closed	Support for current ASDM	"The sub-package currently allows direct opening of data from zarr and will support WSU ASDM (Wide Band Sensitivity Upgrade) and NetCDF in the future." It would be very beneficial if the XRADIO package would support reading of current ALMA ASDMs for bulk reprocessing. Comparing the read times with the processing times seems to indicate that for all practical purposes, the reading time from current ASDMs would be entirely negligible compared to the processing time.	https://github.com/casangi/xradio/issues/340
340.3	Closed w/ Action	Clarify Documentation Data Model and Serialization	In general, the documentation seems to be mixing the data-model, the serialization (zarr) of the data as well as the implementation of the data-access quite a bit. The data model can and should exist independently of which serialization (zarr) is used. The serialization can and should exist independently of which data-access mechanism/language/tool is used. I would suggest to make extremely clear that - data model with semantics - serialization - software to access are entirely separate throughout the documentation.	https://github.com/casangi/xradio/issues/340
340.4	Closed	Data model and the serialization language independence	In particular, following 4), the data model and the serialization must be entirely programming language independent! It seems to be the case that by using zarr the serialization is programming language independent. And it looks really good and future-proof, too.	https://github.com/casangi/xradio/issues/340

340.5	Closed	Tracking Transformations to Data	One issue that was problematic for ALMA was that if manipulations were done to a MS, the information of which original row was which was lost due to the constant renumbering. In other words, the provenance information was missing. I suggest that the MSv4 is augmented so that the data-model itself allows tracking of what happened to each spectral window in a machine-readable way. This might be alleviated in MSv4 compared to MSv2 as the SPWs seem to be identified by a name rather than by a row number.	https://github.com/casangi/xradio/issues/340
340.6	Closed	Naming of Processing Set	The current MSv4 is centered around processing. That's also why several SPWs are called a processing set. Is that good enough also for the future? Would a different name and concept be more useful than 'processing set'? Measurement sets typically would exist independently of the action of processing. Maybe 'observation set' instead of 'processing set'?	https://github.com/casangi/xradio/issues/340
340.7	Closed w/ Action	ALMA specific data	Depending on how self-contained the MSv4 should be, there would be other metadata required, like Project code, title of observation, PI name, ... For the ObservationInfoDict there are also items that are missing for ALMA: Member_OUS_ID, Group_OUS_ID, and certainly many more.	https://github.com/casangi/xradio/issues/340
340.8	Closed	XRADIO Dependencies and Code Longevity	It is clear that a new implementation of data-access software should make use of existing tools rather than writing everything by itself. However, the danger is that there are many dependencies which render the product unstable. The list of dependencies of the XRADIO installation is 155 packages long! Each of these packages has a version requirement. Compare that to CFITSIO who's only dependency is that a working c-compiler exists. It seems to be that this is an enormous risk. XRADIO data-access will need to exist for decades. While the existence of a c compiler can be guaranteed for 40 years, none of these packages can, probably not even python itself can. We have seen that it is very hard to run old versions of CASA on modern OS even though the entire python and all packages were shipped together with the tar-ball. We also know from machine learning applications that typically use a similar amount of packages, that is essentially impossible to install software that was put onto github even just a couple of years earlier because packages have changed, versions have changed, functions are deprecated, functionality has moved. To me this is a BLOCKER item: The access function of a serialization of a next generation data-model must be relying on a very small number of libraries, which are all	https://github.com/casangi/xradio/issues/340

			under the control of the entity providing the serialization and which can be reasonably well guaranteed to be existing in e.g. 40 years from now. (Not that relevant for ALMA itself as ALMA will most likely continue to store in ASDM and FITS, but highly relevant for all other observatories who might store data in MSv4). The conversion from MSV2 to MSv4 uses casatasks. Are they guaranteed to be available for 40 years? Maybe several packages are needed. One that is only using numpy as dependency and allows to read the MSv4. That could also move to astropy natively, like fitsio. And then other packages with more and more remove functionality that users can but do not have to install. Overall, the number of packages that XRADIO uses must be reduced to the absolutely bare minimum.	
340.9	Closed	Virtual Environment	Related the BLOCKER in 340.8) is that I got the following error message when trying install the XRADIO package ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts. tensorflow 2.13.0 requires numpy<=1.24.3,>=1.22, but you have numpy 2.0.2 which is incompatible. tensorflow 2.13.0 requires typing-extensions<4.6.0,>=3.6.6, but you have typing-extensions 4.12.2 which is incompatible. boto3 1.24.85 requires botocore<1.28.0,>=1.27.85, but you have botocore 1.35.36 which is incompatible. Sure, people can use venv, but that already proves the point: The software to just *read* a serialization of a MSv4 *can not already be so complicated that it requires to use an entire virtual environment*!	https://github.com/casangi/xradio/issues/340
340.10	Closed	Archival Format	Related to the BLOCKER in 340.8), is that with these dependencies for the data-access the data-format can *not* be used for long-term storage of data, i.e. archiving. Here is the definition of sustainability that the US Library of the Congress uses: https://www.loc.gov/preservation/digital/formats/sustain/sustain.shtml Here is the entry for FITS just as an example: https://www.loc.gov/preservation/digital/formats/fdd/fdd000317.shtml zarr is suitable for long-term storage. It defines the byte-layout on the disk. The main	https://github.com/casangi/xradio/issues/340

			reading routines (C, C++, Java) should be so low-level that also in 40 years they would be useable. Whatever language/AI will be fancy in 40 years from now, can then still at least wrap that reading routine. And XRADIO should make use of that minimal function itself, so that it is guaranteed that this core reading function of MSv4 is always up to date and maintained and bug-free.	
340.11	Closed	Archival Format	Related to the BLOCKER in 340.8), is a non-exhaustive list of properties I think are needed to call a data-format a format suitable for archiving. - Format needs to be open, widely adopted, transparent and documented, patent-free, (self) documented, standardized, without external dependencies - Needs to be readable with as many tools that astronomers use as possible - Needs low-entry barrier for usage (simple is better than complex) - Needs to be managed by deliberate process under configuration control (e.g. a standardization body/group) - Must be efficient to download, read and manipulate - Has to remain stable (forwards compatible) in time (once FITS always FITS) - Must be described by the byte-layout on the disk, not by an API (Must be possible to write a reading routine in a few decades from now in a then modern programming language) - Must be programming-language agnostic and have readers in many programming languages available ALMA is currently planning to store the final products in FITS also for the WSU era. Therefore the XRADIO dependencies and risks are not an issue. For processing it is good enough if the data-format is stably accessible for the time of processing a single dataset. But for other facilities, in case they would plan to store MSv4 for longer than a month or two, this is a serious issue in my opinion that needs sufficient attention.	https://github.com/casangi/xradio/issues/340
340.12	Closed w/ Action	Specify non-functional requirements	Related to the BLOCKER in 340.8) it would be probably good to explicitly specify non-functional requirements for the MSv4 as well as for XRADIO somewhere. I.e. the vision and mission goal. Even if only a single version of data-access is provided for the MSv4 with the zarr serialization, i.e. XRADIO using python as a progrmming language, the requirements would need to be that * the reading software can be installed on operating systems at least going 10 years	https://github.com/casangi/xradio/issues/340

			back in time * the reading software must work without containers and be simple to use, not more complicated than cfitsio (or e.g. pyfits). * there must be at least two reference implementations for the reading software (that's for example a requirement that the IVOA gave themselves for each new standard they approve) * ...	
340.13	Closed	Do not use Conda	Do not use Conda This page https://github.com/casangi/xradio starts with using conda. Do not use conda! Anaconda is a commercial entity. Organizations with more than 200 employees need to get a license: "Use of Anaconda's Offerings at an organization of more than 200 employees requires a Business or Enterprise license. For more information, see our full Terms of Service, or read Frequently Asked Questions about our Terms of Service." The entire workflow and usage *must* only rely entirely open-source and free software (e.g. GPL, LGPL, ...)! Just like conda was free previously, all software used must be carefully checked so that even in 10 or 20 years from now, there is no dependence that could turn into a vendor lock in.	https://github.com/casangi/xradio/issues/340

340.14	Closed w/ Action	Schema Checker	15) While the serialization (zarr) is really programming language independent, I had overlooked that the definition of the data model and thus the origin from which code-generation can happen is python https://xradio.readthedocs.io/en/latest/_modules/xradio/measurement_set/schema.html Python is in general not really suited for longevity. There is no standard as in 'ANSI C' for example. The schema definition imports 'annotations' from '__future__', it uses 'typing', it uses 'xarray_dataset_schema', definitions in 'nympy' etc. All those are subject to substantially change over time. This seems to be a risk. The main source of the schema should in my opinion be written in a standardized language (UML?, XML?) that is not subject to change for decades and that can be used for code-generation in any other language where needed. In the same way zarr is language agnostic, the schema needs to be, too. In other words: if there is a change to pyhon6 and a change to numpy, xarray, ... none of those should require changing the schema. And as well-supported xarray is, it is not guaranteed to live for e.g. 40 years either. A suggestion would be to make such a programming language independent schema. And then to prove that the schema and serialization are programming language dependent, one way could be to implement code-generation for reading and writing directly from the schema (as it does exist for ASDM) in several programming languages e.g. python, C and java. And to make that test a standard regression test included into the xradio test suite.	https://github.com/casangi/xradio/issues/340