Software Requirements for RFI Management
ngVLA Computing Memo #3

R. Amestica, R. Hiriart, P. Brandt

August 4, 2021

Contents

1 Introduction

In the present note we elaborate on Computing and Software (CSW) features that are necessary to implement the RFI mitigation plan as described in the ngVLA General memo #71 [1]. Emphasis is given to the different RFI stages through the signal path, highlighting the necessary software interfaces and communication channels. The purpose is to highlight what communication interfaces have to be defined between sub-systems, and what sort of computing capacity is required in real-time and during post-processing.

In [1] there are three types of RFI mitigation strategies:

- Avoidance,
- Detection and Flagging, and
- Modeling and Subtraction.

We will identify to what extent each strategy is implemented in software, what parameters are to be transmitted between sub-systems and the frequency at which those parameters have to be updated.

Also in [1] (see Figure 5), different types of algorithms are associated to the different strategies listed above. Pointing out where exactly through the signal path each mitigation action takes place; which help clarify how to allocate tasks among different ngVLA sub-systems. The following is our summary of the information in memo #71 [1] most pertinent for the present note:

1. Outlier detection on the time series data: the Digital Back End (DBE) in each antenna should be able to reliably detect outliers and flag the event.
2. Outlier detection on the real-time spectrum: after the spectral channelizer in the CBF (low resolution tied-array or proper pre-correlator channelizer), or some RFI specialized channelizer.
3. Automatic flagging on the visibilities at high time/frequency resolution: where high resolution relates to the data environment within the CBF before results towards the Correlator Backend (CBE) are time averaged or/and spectral averaged. NRC's initial approach is based on a "simple power threshold flagging" (e.g. spectral Kurtosis, sliding window power, look-ahead RFI flagging), with the possibility for more sophisticated approaches in the future (see NRC [2] section 7.11).
4. Interference modeling and subtraction: provided that time and/or spectral resolution are suitable for image domain or subspace projection algorithms, the interferer contribution could be filtered out [1]. The CBE is considered the right place for the implementation of these algorithms (before averaging for archiving purposes).
5. RFI avoidance in the observation scheduler: based on the real-time RFI detection generated by the system (RFI manager) and the historical record of RFI occupancy (sky direction, frequency and time), observations could be scheduled such that known RFI sources are avoided. That is, specific observations could be moved around in time such that they do not collide with known interferers.
6. Post-processing data analysis: current VLA and ALMA practices provide well understood flagging algorithms. Algorithms like rflag and tfcrop [3] already exist in CASA and it is expected that the RFI database will serve as an additional tunable parameter space for that type of algorithms.

In the listing above, different types of RFI mitigation strategies are associated to different ngVLA sub-systems. In the following sections, actions within each of those sub-systems are analyzed and linked to operations of sub-systems upstream and downstream, highlighting the relation between RFI mitigation strategies across sub-systems and addressing interface requirements between those sub-systems.

Given that some acronyms are frequently used for different ngVLA subsystems, the following list provides some functional context for each one of them:

- Computing and Software (CSW): the ngVLA collection of subsystems that provide support from low level monitor and control functions, up to the proposal management of observations.
- Digital Backend (DBE): a component in each antenna that receives RF broadband ADC samples, formats them and stream them out to the Central Signal Processor (CSP),
- Central Signal Processor (CSP): the component that receives signals from all antennas, and process them according to a configurable functional mode (interferometry, VLBI, pulsar timing and search),
- Correlator and Beamformer (CBF): the component in the CSP that implements the cross-correlator sub-element, and a beamformer VLBI and pulsar functional modes,
- Very Coarse Channelizer (VCC): in the NRC's Frequency Slice Architecture design (FSA) [2], the VCC is a CSP sub-element that converts DBE broadband sample streams down to FPGA manageable sub-bands (Frequency Slices). In that design, the implementation of the VCC and CBF are collocated and the LRU technology applied to each one is the same (TALON modules).
- Pulsar Engine (PE): a CSP part that receives beam-channels from the CBF and process them into average pulsar profiles.
- Dynamic Scheduler: a CSW sub-system that decides what to observe next and what to observe in parallel. It reacts to current configuration and conditions (e.g. array configuration, weather, etc.) and previously generated knowledge (e.g. project ranking, periodic RFI interferers.)

Also, note that data flagging and data masking in the present note are given the following meaning:

1. Flagging: the action of qualifying the data by aggregating metadata to it, and
2. Masking: based on RFI algorithm parameters take action on the current data (e.g. drop samples or replace with a synthetic value) and flag the data accordingly, to report the actual masking action that took place.

Flag values generated in one stage are essential for the next one across the data path. At the end of this chain, the CBE qualifies each product item with a flag value within the project standard format sent to the archive.

2 Digital Backend Stage

The DBE is the first place in the system where antenna samples can be inspected and flagged accordingly. Basic algorithms rely on detecting abnormal voltage levels [3].

If the functionality of the ngVLA Very Coarse Channelizer (see next section), is moved to the DBE in each antenna (see ngVLA Electronic memo [4]), then, flagging information will need to be carried over to the CSP from each antenna, including those antennas connected through the public network. Related Frequency Slice Processors (FSP) in the CSP process the oversampled sub-bands as usual, attaching the effective integration time value (needed by the CBE for further processing and metadata aggregation.)

The corresponding interface between DBE and CSP defines how the flagging data is transmitted through the network from DBE to CSP. The CSW would normally not need to know in detail how that interface works. Except that if some of that flagging information must be preserved after applied at run-time by the CSP, then, a software interface will be required to let DBE or/and CSP to publish that flagging information while the observation continues normally.

We understand that any element capable of flagging the data stream before the VCC (which implies the fastest sample rate in the system), does so by clipping or replacing samples with some predefined values (e.g. noise). The corresponding flagging information is then used by the CSP to act accordingly on the saturated/replaced data. The CSP should provide for a configuration parameter to enable or disable such action. This is a generalization to NRC's [2] requirement L3-1988.

Memo #71 [1] also foresees some capability at this level to remove/replace samples that contain RFI (i.e. voltage clipped.) Plus a means to monitor these events at a convenient frequency, and ingest them to the RFI database. From an Monitor and Control (M&C) point of view, we assume a simple interface like that depicted in Fig. 1. M&C function calls provided to configure the mechanism are supported by data structures like those in the following pseudo-code example:

```
enum class DetectionType
{
    DISABLED,
    SPECTRAL_KURTOSIS,
    SPECTRAL_PEAK
};

enum class ActionType
{
    LEAVE_SAMPLE_ALONE,
    REMOVE_SAMPLE, /* if actually possible */
    REPLACE_WITH_SYNTHETIC_NOISE
};

struct DetectionEvent
{
    AntennaId antenna;
    Time timestamp;
    Duration duration;
    float centerFrequency;
    float bandwidth;
};

int setDetection(DetectionType typ, ActionType action);
```

A DetectionEvent structure is expected through the M&C data stream every time that the algorithm has triggered a detection. Temporal and frequency data members in the structure help characterize the temporal and frequency occupancy characteristics of the interferer. Implicit are any parameters needed to tune the detection algorithm and any additional metric that the detection might produce. Their exact definition will happen with the corresponding DBE ICD documentation.

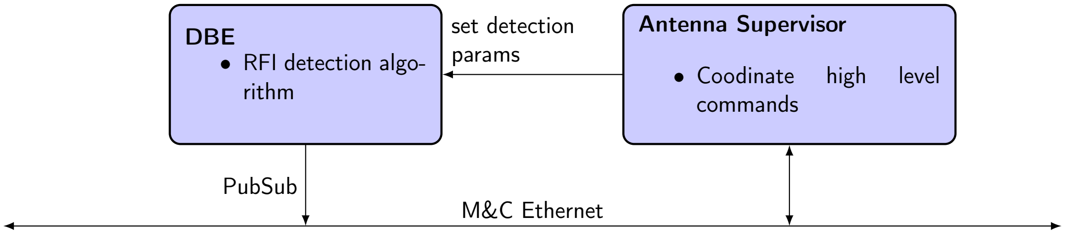

Figure 1: simplified M&C view to parametrize RF broadband RFI detection algorithms that have access to raw ADC samples. A publish/subscribe mechanism transports detection events to the higher level software environment for ingestion into a permanent database and for real-time monitoring feedback.

A priori, a publish/subscribe mechanism seems appropriate to move detection events from each antenna to the supervisory environment in the control building, and from there to the RFI database. Based on the exact M&C technology in place, the DBE will certainly have access to a publish/subscribe mechanism (e.g. OPC UA PubSub) to communicate data to the antenna supervisor computer. Or better, the DBE RFI algorithm might skip the antenna supervisor all together and publish detection events directly to the M&C bus, to be consumed by other subsystems downstream (e.g. RFI Manager.)

Alternative to a publish/subscribe mechanism, there is also the option to attach flagging information to the data itself. In that case, the design must pay attention to how to accommodate the payload without interfering with the science data and the execution of the next component downstream.

To avoid overloading and saturating the M&C bus, reporting of detection events at different levels must be equalized down when the RFI condition is permanent. In that case, the detection algorithms might need to be disabled automatically or by an explicit command, and the data of that antenna flagged or discarded permanently during that observation. Before archiving the science data, the CBE will always flag data that was partially averaged (in time and frequency), but only at the archiving resolution (seconds and megahertz.)

Also related to data-rates streamed through the system, memo #71 [1] suggests a capability to snapshot the RF broadband signal per antenna and transport the result through the M&C bus. If we assume that all IRD modules are sampled at 8-bits (including band #6 [5]), then, each IRD module will sample 7 GHz of RF signal (let's not consider here the reduction due to alias filtering), physically represented as two streams (I-Q) of data sampled at $7\,\mathrm{GS\,s^{-1}}$ each.

A one second snapshot of a single IRD module would then amount to a total of:

$$2 \times 7\,\mathrm{GS\,s^{-1}} \times 1\,\mathrm{s} \times 1\,\mathrm{B} \approx 13.04\,\mathrm{GiB}$$

A corresponding RAM buffer has to be accommodated within the DBE/DTS Module, and FPGA logic to access the buffer and stream out its content. Let's say that a conservative number for the speed at which the snapshot is streamed out is $10\,\mathrm{MiB\,s^{-1}}$. In that case, it will take approximately 22 minutes to transmit the buffer.

That is, to snapshot a single bandwidth slice of 7 GHz (single polarization) requires a RAM module of 13.04 GiB in the DBE and 22 minutes of 'waiting' time to complete its transmission to a receiving end.

To scale up to a larger bandwidth simply multiply by the actual number of IRD modules to be snapshot.

These notional figures are provided just as a draft idea of the required DBE effort to implement a one second RF snapshot capability. From a CSW point of view, the relevant parameter is the maximum data-rate at which data is transferred and, therefore, the maximum expected wait time. A corresponding CSW requirement must be provided to limit the maximum data-rate and, therefore, avoid interfering with other M&C activity through the network.

3 Central Signal Processor Stage

The CSP implements a number of functional modes:

- interferometry (fully connected correlator, including auto-correlations),
- VLBI phased-array,
- pulsar timing, and
- pulsar search.

Interferometry and VLBI modes directly produce data products that are consumed by sub-systems downstream (e.g. CBE and VLBI recorders). On the other hand, beams for pulsar modes are instead consumed by the Pulsar Engine (a CSP sub-element) that in turn produces pulsar profiles as end CSP products.

The correlation functionality needed for interferometry and beamforming are both implemented in the CSP by means of highly reconfigurable hardware modules. The collection of these hardware modules creates what is called the Correlator and Beamforming (CBF) sub-element in the CSP.

In summary, there are four functional parts within the CSP:

- Very Coarse Channelizer (VCC): receives wideband sample streams from each antenna, and produces narrower oversampled sub-band streams,
- Beamformer: a number of complex true-delay or phase delay beams, channelized with a coarse resolution of up to 4k channels,
- Correlator: generates visibilities at a spectral resolution of 16k channels, or only 1k/4k channels as a so called tied-array to support the calibration of beamforming modes, and
- Pulsar Engine: it processes true-delay and phase-delay beams into pulse profiles.

CSP functional parts are concerned about RFI in different ways. But as presented in the CSP Reference Design (Fig. #5) [6], their commonality is to detect and flag RFI artifacts right before the antenna samples are further channelized in frequency. NRC's design encapsulates such functionality as a pre-correlation threshold detector/flagger FPGA IP block [2] (section 7.11); which should not preclude adding post-correlation RFI functionality to the CSP design in the future (for example, see block B-5 description in NRC [2].)

In the following subsections we overview the RFI detection functionality in the CSP and how that capability must be supported by the online software. This overview is not exhaustive but it helps weighting the CSW effort to establish the necessary interfaces with the CSP.

3.1 Pre-correlation RFI detector/flagger

The detector's logic inspects antenna samples (at their wideband and sub-band sampling rate) and flag those that do not match a predefined metric. Flagging information associated to each sub-band is then conveyed to all involved Frequency Slice Processors (FSP) downstream. Involved FSPs detect flagged samples and avoid them from further integration. At the end of an integration, the effective integration time is reported accordingly to the CBE.

During real-time execution, access to a detector's functional statistics (e.g. event's time stamp, duration) are available through the M&C bus. The following NRC design requirement [2] is an example of such functionality:

- L3-1889: communicate applied time-dependent receptor weights When commanded, $\text{CSP}_{\text{Mid.CBF}}$ shall communicate to $\text{CSP}_{\text{Mid.LMC}}$ all time-dependent receptor weights calculated and applied internally to any tied-array beam sum, with at least the temporal resolution of the currently configured subarray visibility integration time (or 1.4 seconds if none are currently being produced), and with the frequency resolution of the beamformed or channelized bandwidth of each beam.

The convenience to regularly monitor RFI detection events and the algorithm's performance, before metadata has been collected downstream in the system, is appreciated as a system feature useful to optimize the telescope operation as soon as a certain condition is detected (e.g. switching to an alternative source in the sky when excessive flagging is occurring). See additional arguments at the end of the next subsection.

We explore now how the NRC design [2] characterizes run-time RFI events by means of specific parameters, and how these parameters are measured (tuned) to the current conditions. This is useful to appreciate how the M&C subsystem must accommodate RFI calibrations and mitigation actions during an observation.

The FSA design concept [2] includes the following related specifications to deal with RFI detection before correlation and beamforming (reproduced here verbatim from [2]):

- L3-1832: RFI detection and flagging When commanded, $\text{CSP}_{\text{Mid.CBF}}$ shall automatically detect RFI in the processed bandwidth for any Band and, when detected, flag data to prevent corruption of any downstream data products. Conformance to this requirement shall be determined by testing of all SKAO-mandated RFI use cases.
- L3-1833: RFI masking When commanded, $\text{CSP}_{\text{Mid.CBF}}$ shall flag real-time (pre-visibility integration and pre-beamforming summing) data streams at all available channel resolutions according to a pre-selected RFI Mask, in accordance with the CSP LMC to CSP Sub-elements ICD.

That is, NRC's approach consists on a calibration procedure to quantitatively characterize the current RFI occupancy, and to translate that information as parameters to the algorithms that detect RFI and act accordingly on the data. The parameters, or RFI mask as defined in NRC [2], is valid only during the interval associated to that calibration. Encapsulating the parameters in a custom data structure and storing them in a permanent database makes possible to analyze them at a later point and discover repetitive patterns among the different calibrations (e.g. time of the day, geographic direction.) Basically, a data-mining operation to extract new information from previously measured RFI occupancy data points.

NRC's design includes different RFI detector/flagger blocks through the signal path. Thus, providing mechanisms to address RFI contamination at different temporal and frequency resolutions in the system

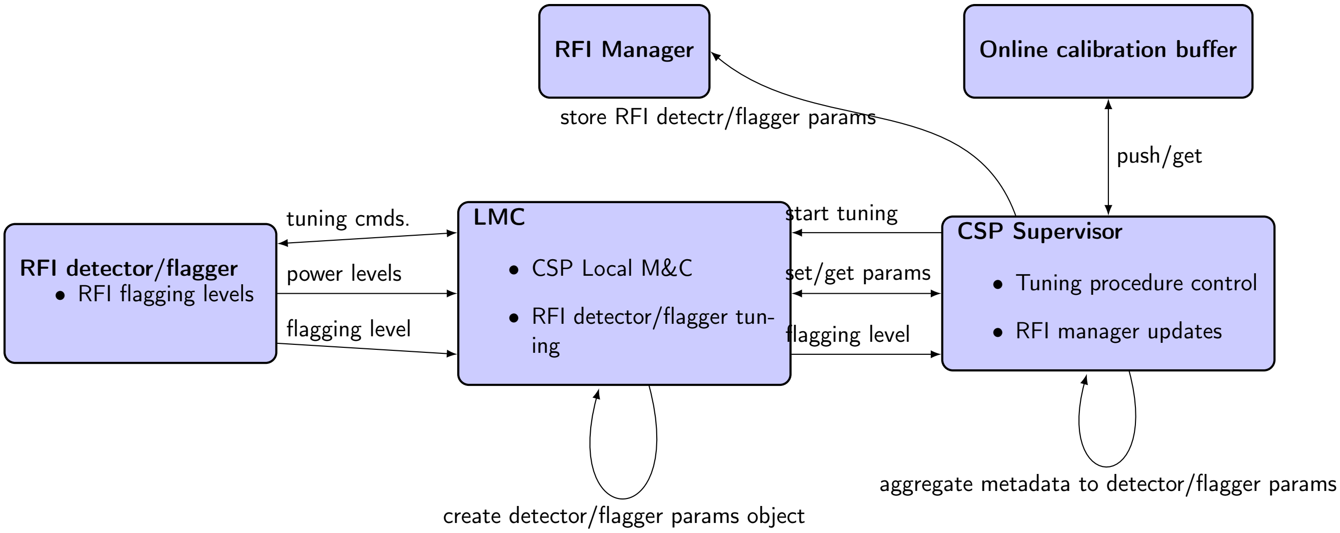
RFI Manager
Online calibration buffer
store RFI detectr/flagger params
push/get
tuning cmds.
LMC
start tuning
CSP Supervisor
RFI detector/flagger
• RFI flagging levels
power levels
• CSP Local M&C
set/get params
• Tuning procedure control
flagging level
• RFI detector/flagger tuning
flagging level
• RFI manager updates
aggregate metadata to detector/flagger params
create detector/flagger params object

Figure 2: a feedback loop to get the RFI flagging level settings for each receptor "about right", for optimal end-user data quality [2].

(raw samples or sub-bands). For a correct operation, some parameters have to be tuned to the current power levels of the RFI interferer and science signals. One specific tuning procedure is described in NRC [2] (section 8.2.5) and a schematic overview is shown here in Fig. #2.

From the description in NRC's [2], it is clear that a proper implementation of the calibration procedure will require some experimentation. Providing best-guess default values for the parameters (e.g. 'dwell' times, power levels, etc.) is critical to integrate the procedure early during system verification and commissioning.

On every successful iteration, the tuning procedure will produce a set of parameters applicable to the current observing conditions. These parameter values are part of the larger instrument calibration as a whole. Individual instances of a calibration object must be accessible from a buffer for online reuse. At the same time, they must be stored in the permanent RFI database together with pertinent metadata for its future offline analysis (e.g. observation time stamp, sky direction, frequency, etc.) The logic is that a history of RFI calibration parameters and metadata should help characterize RFI occupancy as a function of operational conditions.

The following are the suggested RFI detector/flagger calibration parameters in NRC [2]:

- flagging ON/OFF,
- non-RFI level establishment integration time,
- non-RFI level setting (optional, overrides internal level determination),
- RFI trigger algorithm (Power, Kurtosis),
- RFI trigger detect integration time,
- flagging trigger level, and
- flagging dwell time or "Auto".

The information shown above works as input parameters for the proper functioning of the RFI detector/flagger during an ongoing observation. But it will only be after aggregating metadata to those parameters, that their history in the RFI database makes possible to analyze and deduce the interferers characteristics (e.g. power level, position coordinates, frequency of occurrence, etc.), as part of a continuous RFI data mining process.

A priori, the following is the basic information that should be aggregated as metadata to each detector/flagger parameters:

- time interval during which the parameters were applied by each antenna,
- sub-array description (i.e. set of antennas),
- antenna in the subarray to which the parameters correspond to,
- antenna pointing position when parameters were measured,
- frequency selection (implies what sub-bands were in use),

Once RFI flagging is turned on, monitoring the level of RFI detection and flagging will play an important role. It allows operators or some software automatism in the dynamic scheduling logic to take immediate corrective actions to improve how the telescope resources are being used at that moment. For example, if the flagging level is too high across a large fraction of the array, then, it might be better to switch the telescope to an alternative observation project.

As prescribed by L3-1889 in NRC [2] (see above), the CSP will allow for the periodic monitoring of pertinent RFI detection and flagging statistics. As for the basic estimation of the data-rate produced by an RFI detector, let's assume that a detection event must contains the following information:

- timestamp (4 bytes),
- antenna identifier (2 bytes),
- RFI detector stage in the antenna (1 byte),
- calibration unique identifier (8 bytes),
- power level measurement (4 bytes), and
- detection counts during the dwell-time (4 bytes)

that's a total of 23 bytes per event. If an RFI detector block is located before the first channelization (output samples of each IRD module in the band) and one for each subband (~200 MHz) then a notional figure for the total RFI monitoring data-rate per antenna is:

$$23 \times 2 \times \left(8 + \frac{28\,\text{GHz}}{200\,\text{MHz}}\right) \times \frac{1}{1\,\text{s}} \approx 6.7\,\text{KiB}\,\text{s}^{-1}$$

where it is assumed that both polarization streams report independently, the maximum number of IRD modules is 8 in band #6, samples are 8-bit, the maximum bandwidth is processed (28 GHz) and an average time of one second is used to report the current RFI detection level. If we scale up to include

263 antennas in the array, then, the total RFI monitor data rate (pre-correlation) reaches a value of 1.7 $\mathrm{MiB\,s^{-1}}$.

It is probably unlikely that the whole array would detect and report RFI activity at the same time, and simultaneously across the whole 28 GHz processed bandwidth. The above figures are presented just as a notional worst case scenario at a reporting rate of one event per second.

3.2 Post-correlation flagging

Visibilities produced by the CSP must be accompanied with a flag value that informs of any RFI condition that affected each datum. The CBE downstream could then take specific actions based on the flag value, and aggregate that information to the final data product sent to the archive to enable post processing decisions based on the reported data quality.

One aspect to consider is that the extra flagging information must be accommodated seamlessly through the communication channel between CSP and CBE. For that, it is interesting to consider the total payload binary increase. Because the specification for the switched interconnect hardware between CSP and CBE is based on the total rate of bits per unit time transported.

The most apparent approach for flagging is to add information to each visibility datum. That is, each visibility (single precision floating-point complex value, 64 bits) is extended with a 1-bit flag, that qualifies whether RFI is present or not. In this case, the extra binary information to be transported amounts to $\sim$1.6 %. If additional flagging qualification is required, then, extra bits are added to the flag value with the corresponding data rate increase.

On the other hand, a different flagging mechanism can be envisioned in which a range of visibility channels are assigned the same flag value. This optimizes the flagging information transported, and minimizes the fractional data-rate increase. This could prove useful only if a large variety of RFI occupancy conditions fits a range-based flagging scheme.

For example, DME transmissions in [3] are characterized as as 0.1 MHz frequency occupancy slices spaced 1 MHz apart ((\190 MHz(\) total bandwidth). In a 16k channels subband, two 14-bit integer values can represent the start and end channels of each detected RFI bandwidth slice, leaving 4 bits to represent a flag value (8 different values) in a 32-bit flag word. For an anti-aliased subband of 200 MHz and correlated at 16k spectral channels, then, the fractional payload increase (full subband DME occupancy) is:

$$\frac{\left\lceil\frac{200\,\mathrm{MHz}}{1.1\,\mathrm{MHz}}\right\rceil \times 32\,\mathrm{B}}{16 \times 1024 \times 64\,\mathrm{B}} \approx 0.56\,\%$$

Both approaches are to be compared based on the cost associated to a fractional data rate increase, any requirement to qualify the visibilities with more than 1-bit of information, and firmware and software complexity to implement one or the other scheme. A final conclusion should translate into a CBE requirement to specify how the online software has to extract and interpret flag values from visibility data streams.

3.3 Time and spectral resolution

The integration duration and the spectral channel bandwidth in interferometry mode are parameters that make possible to control the total data rate and data volume. As such, they are usually applied to constrain an observation to the minimum system resources that accomplish its science goal. The parameters also have an influence on the RFI detection algorithms [1].

The following two CSP requirements define both parameters [7]:

- CSP0221: The channel bandwidth shall be selectable within the range from 1 kHz to 7.2 MHz in steps less than or equal to one octave.
- CSP0225: The CSP shall support visibility integration times ranging from 2 ms to 10 s, configurable in steps smaller than or equal to one octave.

As a function of the RFI conditions during an observation, it should be possible to tune these two parameters and thus optimize how efficiently the RFI is detected and later excised post-correlation by the CBE. It is then advisable to design in a mechanism to adapt their values dynamically during an observation [1]. That is, adapt the averaging to not dilute the RFI signal below the detector thresholds and into the science data. This capability should be provided through the dynamic scheduling subsystem and it should make use of the CSP parameters listed above, plus their equivalent counterparts used by the CBE to average CSP results before archiving.

The merits of such a scheduling capability depend on the reliability to monitor RFI conditions through the M&C system (e.g. detector/flagger statistics) and the observational margin available for the observation's science goals. A higher resolution from CSP to CBE implies a higher data rate that the system must allocate together with other sub-arrays at that moment. If that's not a concern at a given moment, then, the resolution could be increased giving the CBE a chance to excise data at a finer level that better matches the interferer's characteristics.

Altering the CSP resolution parameters at run time requires a reconfiguration of the specific observing mode, which is a proper CSP feature. A priori, there are two aspects that complicate the implementation of this feature:

- reconfiguration of the CSP hardware takes some time to complete before the observation can continue (typically a few seconds [2]), and
- it is also necessary to define how the actual CSP resolution is reported together with other metadata in the final data products.

Note that in [1] only the temporal resolution is listed as a desirable feature. But provided that controlling both CSP resolutions (temporal and spectral) follow similar implementation requirements, then, we suggest designing in both options. However, it must be noted that adapting the spectral resolution to the current RFI condition, necessarily requires increasing the resolution of the final product sent to the archive; which in most cases will not be actually possible due to CBE to archive data-rates constraints.

The above description informs about the software effort necessary to implement a capability mentioned in [1]. Real-time RFI monitoring and a dynamic scheduler are features that make this implementation possible, as a very organic feature of the system (the second bullet above requires some additional discussion, though.) If further analysis validates the value of a variable CSP integration time, then a formal CSW requirement should be derived from that.

3.4 Pulsar Engine

As per the current CSP reference design [6], the area of interest are the pulsar timing modes (sparse and dense.) That is, those pulsar modes that require the Pulsar Engine to process and produce a pulsar profile for each beam. Pulsar Search mode products are instead, the subject of the offline processing stage.

In Pulsar Timing mode, the pulsar time period is already known and the processing expedient in the PE consists of the following actions per sub-band and beam:

- remove de-dispersion delay by means of coherent de-dispersion,
- stitch sub-bands results, and
- add many pulses together (time folding)

It is during the coherent de-dispersion step that RFI mitigation actions take place (see Lorimer & Kramer [8]). The actual example in section 7.5 [8] points to the following parameters for an excision algorithm:

- level of individual samples above median,
- in a sample, level that a number of bins exceed over the median,
- level above running median exceeded by a block of bins.

The actual levels are parameterized as a factor of the input signal level. The description in Lorimer [8] is emphatic on pointing out that tuning the parameters will require experimentation. From a CSW point of view, the critical aspect is to make sure that the parameters are part of the instrument calibration process and, therefore, that their values could be tuned and changed as often as a calibration observation takes place.

If data samples are excised due to the signal level constrains listed above, then, the effective folding time duration should be attached to the averaged output profile. The CBE downstream takes this information and retrofits such information to the Metadata Capture component. Thus, completely qualifying the data product for later offline processing.

4 Correlator Backend Stage

The experimental status and the complexity of RFI modeling and subtraction algorithms makes of the CBE a convenient place to deploy them in the system [1]. This is because visibilities are handled here at a few tens of milliseconds and kilohertz (before averaging down to archive rates), and also because the CBE computing capability can be expanded in the future by means of readily available COTS compute nodes, as needed to adapt to evolving algorithms.

The basic functions associated to the CBE are the following (see CSW reference design [9]):

- collect array visibilities, average them in time and frequency as configured, pack them in an observatory standard format and stream them out to the Science Data Archive,

- collect pulsar timing profiles (Pulsar Timing mode) in the PSRFITS format and stream them to the Science Data Archive,
- collect beam-channels (Pulsar Search mode) in the PSRFITS format and stream them to the Science Data Archive, and
- collect VLBI beam-channels in the standard VDIF data format and record them on a collocated specialized hardware (a.k.a. VLBI recorders).

Beside its basic functionality, the CBE must also be aware of RFI flagging and weights from the CSP upstream, and it should also accommodate algorithms to detect and remove RFI artifacts. That is:

- take action on already flagged data products, and decide to extract or retain the datum for further integration, maintaining the associated flag,
- adjust the weight or integration time based on the action taken in the previous point, and
- detect the presence of RFI artifacts and mitigate their effect by means of real-time imaging and/or subspace projection methods to detect and characterize the interferer's signal.

RealFast [10] and VLITE-Fast [11] are science experiments capable of real-time imaging (real-time visibilities) to study astronomical transient signals [3]. Given the transient nature of some RFI interferers, it results interesting to consider similar approaches for RFI detection and mitigation efforts at CBE level.

However, hardware and software platforms required for real-time imaging and subspace projection excision, like those cited above, are currently an active research area. Therefore, as a best design effort the CSW should consider enough CBE flexibility to execute RFI algorithms on the data stream, and software interfaces to facilitate the commissioning and actual development of those algorithms.

A priori, an additional set of compute nodes to execute the algorithms, or designing the CBE cluster itself with hardware plugins (e.g. PCI boards) on which those algorithms can be accelerated. Let's call this conceptual CBE element the CBE co-processor.

Mapping a subarray to CBE resources involves several compute nodes in the CBE cluster. Channeling data from all those CBE nodes to the co-processor is a design challenge for such a system implementation. However, one should also take into account parameters that reduce the stress on that system. Like for example, those associated to the subspace projection algorithms investigated in memo #38 [12]:

- Integration time: over which the interferer signature is assumed constant, and
- Sub-array partitioning: based on baseline lengths, partition the sub-array in a few subsets.

That makes possible to limit how often visibility data has to be sent and results retrieve from the co-processor, and how much data is there to transport; which helps limiting the complexity and performance at which memory is shared between the CBE cluster and its co-processor. For example, the subspace projection method in [12] requires looking at all the visibilities but grouped in smaller baseline subsets. These smaller subsets could then be processed in parallel in the co-processor for eigenvector decomposition (TBD: simulate how big a subset of visibilities could actually be).

As a quantitative metric, consider that a sub-array containing all the 263 antennas, at an arbitrary integration time of 50 milliseconds (STI in [12]) and 16×1024 spectral channels per sub-band, then, transporting single precision floating-point visibilities implies a data rate of $\sim$85 GB s^{-1} per subband and polarization product. If compared with the current PCI Express bandwidths then it becomes clear that this data rate is too fast. It is only with specialized interconnect solutions (e.g. PCI-Ex 6.0, Infiniband) that transporting data at 85 GB/sec is theoretically possible.

Another CBE functionality to consider while commissioning RFI algorithms, is the ability to compare the modified and unmodified results during offline analysis. Accordingly, [1] takes note of this aspect and suggests to output two streams of data: uncorrected (raw) and RFI corrected data products. Now, to stay within limits of what the CBE can actually deliver to the archive, this feature is in general constrained to only half the size of the otherwise normal data products. A similar functionality was used in ALMA to validate WVR phase correction algorithms.

In summary, the notional interpretation given to a CBE co-processor and RFI corrected/uncorrected data streams are depicted in Fig. 3. The exact technology for the co-processor hardware is to be discussed beyond the scope of the present note. To summarize these features:

- CBE co-processor: a set of hardware nodes where the CBE cluster can offload compute execution to accelerate RFI related algorithms, before or during further data integration and spectral averaging, and

- parallel UNCORRECTED and CORRECTED output streams, where a flag in the scheduling block is used by the CBE to enable a given RFI mitigation algorithm, and stream out its output in parallel with the raw data. Possible values for the flag are: UNCORRECTED (algorithm not executed), CORRECTED, and UNCORRECTED+CORRECTED.

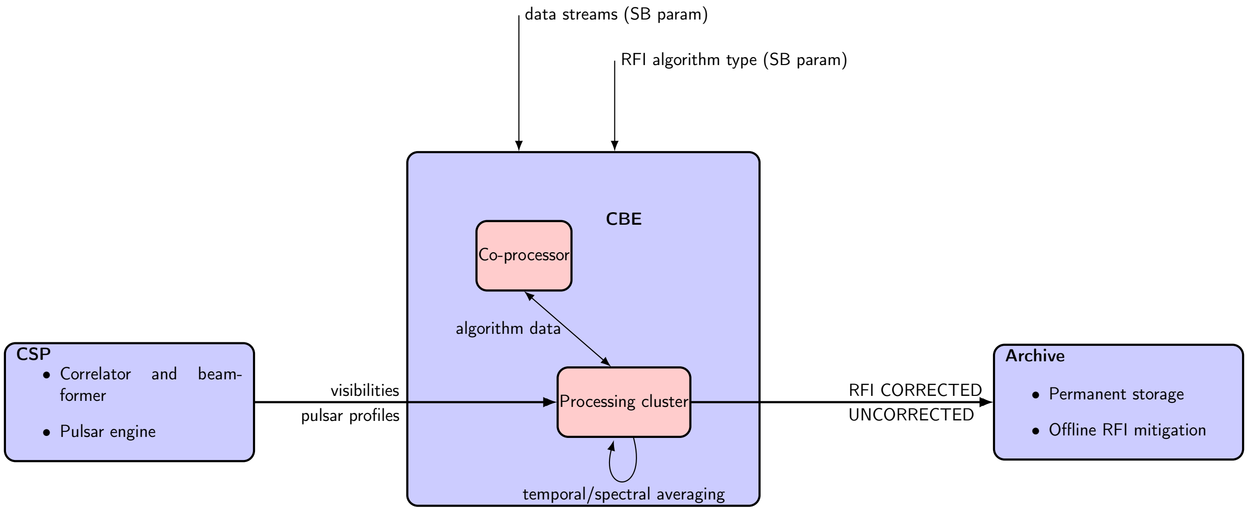

Figure 3: the notional depiction of a CBE that incorporates a co-processor hardware to accelerate the execution of heavily parallelizable RFI mitigation algorithms (e.g. matrix operations for subspace projection) and a software feature to simultaneoulsy output corrected and uncorrected data, to help develop and commission the related RFI algorithms.

5 Post-processing Stage

As described in [1], auto-flagging algorithms like 'rflag' and 'tfcrop' in CASA, are examples of how current pipelines automate detection of RFI in already archived science data (typically with a ~1 second and ~1 MHz time and spectral resolutions, respectively.) In the same memo, Fig. #2 shows different stages in the ALMA data analysis pipeline. Emphasis is made about the need for some expert supervision of the pipeline results.

For ngVLA the aim is to enable the aggregation of the online flagging information produced by each online component (DBE, CSP and CBE); plus the information in the RFI database. All this together informs the post-processing auto-flagging algorithms, such that the pipeline reliability is improved. The CSW sub-system just needs to create the interface necessary for the post-processing pipeline environment to access flags created online and to query the RFI database.

Online flags are expected to be kept together with the science data products themselves, as entries in the corresponding data container (e.g. CASA measurement set); which is the standard assumption when thinking about the products created in the CBE and sent to the archive subsystem. However, note that these products are generated at a temporal and spectral resolution compatible with the archiving metrics.

On the other hand, DBE and CSP are also expected to produce flagging and RFI occupancy information, and at a higher resolution. This aspect and provisions for accessing the RFI database are discussed in the next section of the present memo.

6 RFI Manager and Database

The concept of an RFI manager and its associated database is a key aspect to how ngVLA plans to address its RFI challenges. The unavoidable evolution of RFI sources (e.g. Low Earth Orbit satellites) and the remote location of a number of antennas in the array, will certainly challenge every RFI mitigation strategy put in place.

The RFI Manager component [1] makes possible to conceptualize the hardware and software interfaces that are necessary to aggregate and shared RFI related information between sub-systems. That is, between sub-systems that generate RFI related information (e.g. flags) and others that require tuned parameter values to flag and remove RFI artifacts. All this during the real-time execution of individual observations and during data post-processing stages.

An RFI Manager is the entry point for every involved ngVLA sub-system to either inform the detection of RFI events, to input events that are of a predictable nature (e.g. satellite ephemeris) and to access that information whenever required to tune algorithm parameters (e.g. real-time flagging, clever scheduling, etc.) This makes evident two distinct aspects for an effective RFI manager:

- data ingestion and query: as part of the M&C implementation plan, proper mechanisms to transport data to the RFI manager and to retrieve information back, must be identified as a standard mechanisms within that particular M&C solution (i.e. remote method invocations and data packets across the network.) At the same time, specific data volumes and associated data rates would impose requirements to the M&C solution at play, or/and restrict how much RFI information could actually travel at any moment across the ngVLA system.
- data mining: the main idea behind the RFI database is to let the system learn characteristics of

the RFI environment. This suggests the need for algorithms (e.g. AI algorithms to type and auto-classify RFI events) to match the information collected in real-time together with data already in the longer term storage, and to tune parameters used elsewhere in the system (real-time or post-processing) to flag and/or excise RFI corrupted data. Mining for information in the database must be an automatic algorithm and also a process supervised by an expert user.

As shown in Fig. 4, collected and aggregated RFI information (online and offline) is continuously processed for classification purposes and pattern discovery. This process is meant to happen all the time as an automatic data-mining operation; which depends on custom algorithms developed to explore the database. The RFI Manager sub-system must grant a seamlessly access to the database for both, automatic online algorithms and human experts whose work create new algorithms or improve already existing ones.

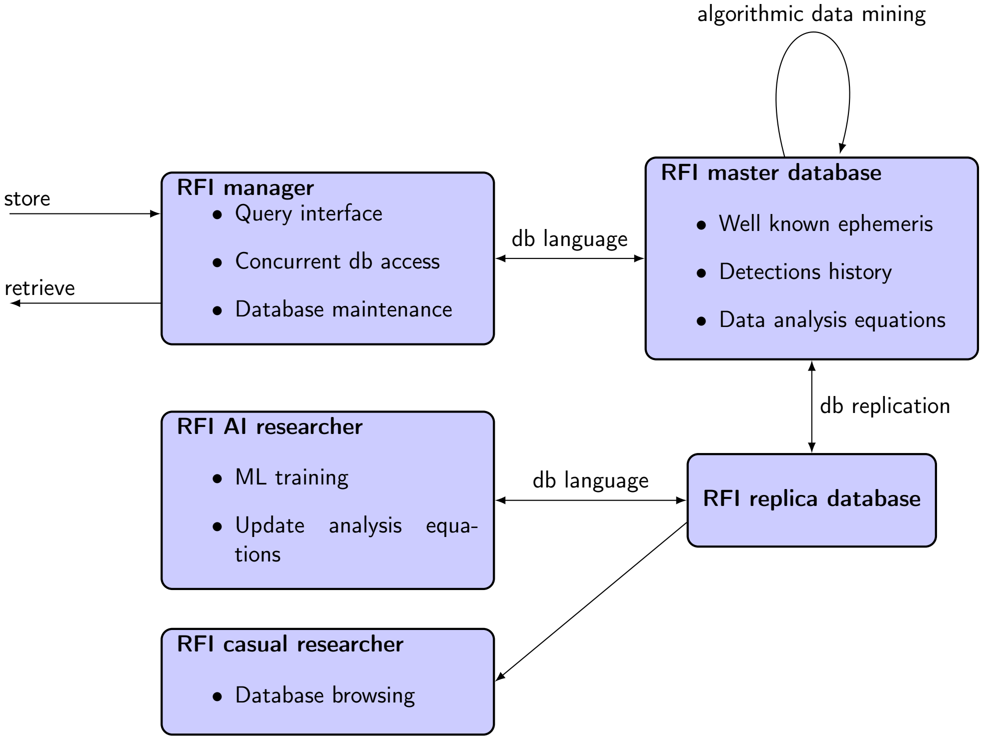

Figure 4: the RFI manager provides an online interface to ingest and retrieve information in/from the permanent RFI database. Data mining algorithms will routinely process new fields in the database to improve RFI occupancy models.

Now, from a database implementation point of view, the RFI detector/flagger parameters described earlier and satellite ephemeris are the only information to be stored in the database. But in order to grow from already available experience at NRAO, it is interesting to note the type of information and actual tables described in A. Erickson [13] about the database implemented for the JVLA instrument. That experience should be retrofitted to the implementation of an ngVLA specific RFI Management component.

7 RFI Occupancy Data Stream

The idea of a data stream for flags is briefly mentioned in [1] (section 5.2.1). The concept is of relevance as a transport mechanism for input parameters to RFI algorithms, and as a mechanism to enable the periodic monitoring of the current RFI environment. However, we make here a difference between both. That is, between flags and the information used to describe the RFI occupancy.

On one hand, flags flow together with the data to let the receiving component take specific action based on flags values. The actual action that a given flag value informs, is specific to the interface between the producer and the receiving parts in that segment of the signal path. The CBE is at the end of this chain and it populates the final data product with flag values as seen at that stage (e.g. integration contains clipped values or reduced integration time due to dropped samples.) Therefore, their time and spectral resolution is not finer than that used for archiving purposes.

On the other hand, every time an RFI algorithm detects a condition, then, the online monitoring of those events is necessary for the system to automatically or manually adapt to the current RFI occupancy conditions. For example, under a high occupancy condition the dynamic scheduler might decide to switch to a different observation all together.

From a software point of view, a publish/subscribe mechanism through the M&C bus seems like a promising alternative to transmit RFI information encapsulated as RFI occupancy data structures. In the sense that establishing and handling connections between senders and receivers is left to a middle-ware entity; which helps reduce software implementation complexities in senders and receivers across different hardware and software sub-systems.

On the other hand, at a hardware centric level, like the interconnect between DBE and CSP, transporting flags metadata from each antenna to the RFI detector/flagger in the CSP, can be chosen to fit hardware specific requirements eventually bypassing the M&C bus. But still, we realize that in order to grow the 'intelligence' accumulated in the RFI database, the detected events should always be summarized and transmitted to the RFI Manager from the respective hardware and software part.

In Figure 5 the idea of a flags stream is shown relaying on the M&C bus as a backbone to interconnect senders and receivers. Detection algorithm generate flags and occupancy statistics associated to the respective data products, and receivers consume them to inform other algorithms or to aggregate the observation's metadata as a whole.

CSP specific figures shown earlier in this note, give a draft idea of the involved volume of data to transport per time unit.

8 Algorithms Version Management

System requirement SYS0606 establishes a functionality to upload different versions of the online software, to control different sub-arrays at the same time. In a similar way, there should also be a requirement to upload different versions of individual RFI algorithms [1] to the operational system.

Note that when referring to algorithms implemented in hardware components (e.g DBE and CSP), SYS0606 is necessarily addressing FPGA personalities and embedded software at the same time. In both cases, it should be possible to upload the new bitstream through the M&C network interface.

We now extend that idea to the CSW components within the CBE that implement RFI related algorithms. That is, a CSW specific requirement that departs from SYS0606 in that only CBE modules

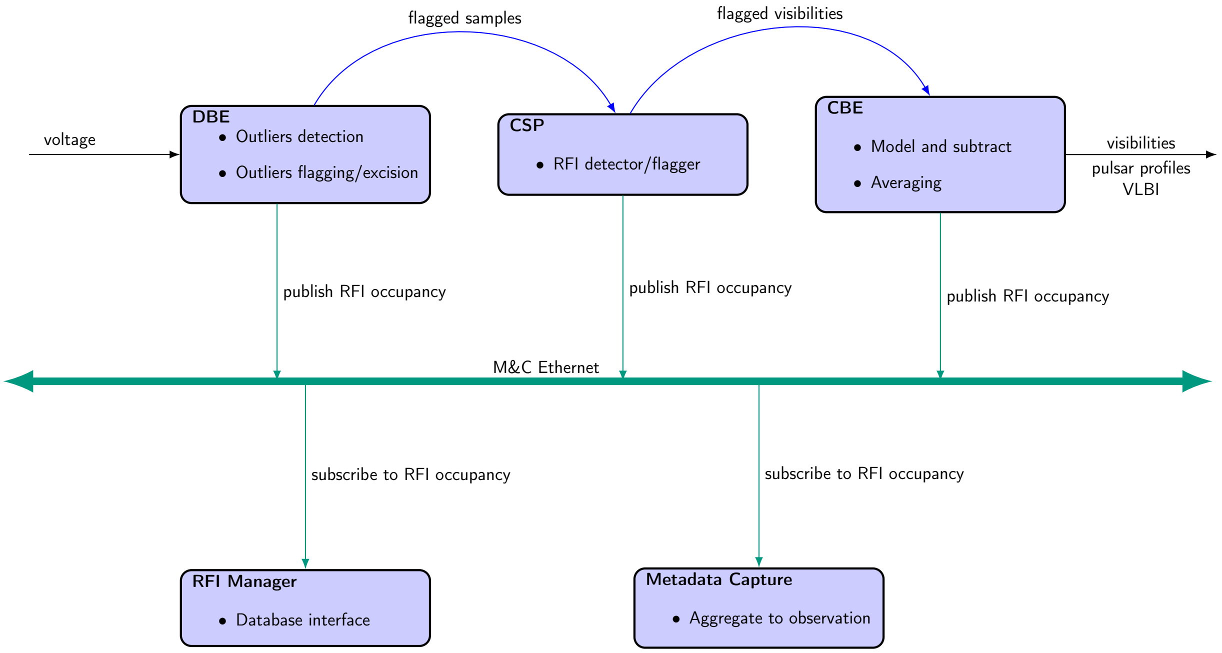

Figure 5: in this representation of a flags stream, the interconnect between the hardware modules DBE and CSP, and between CSP and CBE (blue), transports not only the data products (voltage samples, visibilities) but flagging information as well; which contains the information needed by the pertinent RFI algorithms in the receiving part. However, for online monitoring and RFI database ingestion, the current RFI occupancy information must also be communicated to other software sub-systems through an ad-hoc mechanism over the M&C bus (green). For that purpose, an optional direct tapping onto the M&C bus by the DBE and CSP is also depicted on this diagram.

are updated with a different version, and not the CSW software version at large.

An important parameter of both these requirements (firmware and software module versions) is how often an upload is expected to occur. The DBE, CSP and CBE subsystems will most certainly present different capabilities to upload within a given lapse of time.

During an observation, the obvious moment at which an upload could happen is in between scheduling blocks and in between scans within a scheduling block. The effort involved to implement one or the other upload frequency is different, and RFI stakeholders would have to refine these requirements to finally assess their implementation details and limitations.

9 Conclusions

Based on the project's available information, we have elaborated on mechanisms to transport RFI related information and infrastructure to aggregate and exploit that information.

The most salient aspects are the following:

- RFI detector/flagger logic in DBE and CSP have to be periodically tuned to the changing RFI conditions,

- RFI detector/flagger parameters represent the current RFI occupancy and their values must be stored in the RFI database,
- software and hardware infrastructure (co-processor) within the CBE for accelerating RFI excision algorithms,
- DBE and M&C infrastructure for a short (1 second) RF snapshot,
- RFI database interface for data ingestion and retrieval,
- RFI database replication for maintenance and algorithms research activities decoupled from the online system,
- in-database algorithms for RFI data-mining.

The CBE and post-processing pipeline are the CSW components in which RFI algorithms take place. In both cases, there is already existing experience around those algorithms, coming from other projects and research (elsewhere and NRAO itself.) Nevertheless, new developments are expected in this area and the CSW interfaces and infrastructure (e.g. database access) are to be streamlined to facilitate access to RFI experts.

The information in this memo should help itemize components and requirements to be included in the CSW conceptual design. Each item will certainly require additional analysis before arriving to an exact requirement item in the SysML model and its implementation in hardware and software. For example, prescriptions from RFI algorithm experts to tune specific parameters; which we should finally capture in the respective ICD, as well.

References

[1] Rob Selina et al. *RFI Mitigation in the ngVLA System Architecture*. Tech. rep. ngVLA Memo #71. NRAO, Feb. 2020.

[2] Mike Pleasance. *SKA1 CSP Mid Correlator and Beamformer Sub- element Detailed Design Document*. Tech. rep. SKA-TEL-CSP-00000066. NRC, Dec. 2017.

[3] Urvashi Rau, Rob Selina, and Alan Erickson. *RFI Mitigation for the ngVLA : A Cost-Benefit Analysis*. Tech. rep. ngVLA Memo #70. NRAO, Dec. 2019.

[4] Omar Yeste Ojeda. *Trident 2.1 Concept: Updates to the CSP Reference*. Tech. rep. ngVLA Memo (Electronic) #5. NRAO, Sept. 2020.

[5] R. Selina and O. Yeste Ojeda. *Headroom, Dynamic Range, and Quantization Considerations*. Tech. rep. ngVLA Electronics Memo #8. NRAO, Jan. 2021.

[6] Ojeda. *Central Signal Processor: Preliminary Reference Design*. Tech. rep. 020.40.00.00.00-0002-DSN-A-CSP_PRELIM_REF_DESIGN. NRAO, July 2019.

[7] Ojeda. *Central Signal Processor: Preliminary Technical Requirements*. Tech. rep. 020.40.00.00.00-0001-REQ-A-CSP_PRELIM_TECH_REQS. NRAO, July 2019.

[8] Duncan Lorimer and Michael Kramer. *Handbook of Pulsar Astronomy*. Cambridge, UK: Cambridge University Press, 2005.

[9] R. Hiriart, J. Robnett, and M. Pokorny. *Computing and Software Architecture: Reference Design*. Tech. rep. 020.50.00.00.01-0002-REP-A-COMPUTING_SOFTWARE_ARCHITECTURE_REF_DSN. NRAO, July 2019.

[10] C. J. Law et al. "realfast: Real-time, Commensal Fast Transient Surveys with the Very Large Array". In: *The Astrophysical Journal Supplement Series* 236.1 (May 2018), p. 8. DOI: 10.3847/1538-4365/aab77b. URL: https://doi.org/10.3847/1538-4365/aab77b.

[11] Suryarao Bethapudi et al. "The First Fast Radio Burst Detected with VLITE-Fast". In: *Research Notes of the AAS* 5.3 (Mar. 2021), p. 46. DOI: 10.3847/2515-5172/abea22. URL: https://doi.org/10.3847/2515-5172/abea22.

[12] Mitchell C. Burnett et al. *Subarray Processing for Projection-Based RFI Mitigation in Radio Astronomical Interferometers*. Tech. rep. ngVLA Memo #38. NRAO, Mar. 2018.

[13] Alan Erickson. *RFI Database*. Tech. rep. NRAO, Mar. 2017. URL: https://gitlab.nrao.edu/rfi/rfidb/-/tree/master/documents.