Computing and Informatics, Vol. 39, 2020, 838–859, doi: 10.31577/cai_2020_4_838

# PROCESSING RADIO ASTRONOMICAL DATA USING THE PROCESS SOFTWARE ECOSYSTEM

Souley Madougou, Hanno Spreeuw, Jason Maassen

*Netherlands eScience Center*
*Science Park 140 (Matrix I)*
*1098 XG Amsterdam, The Netherlands*
*e-mail:* {s.madougou, h.spreeuw, j.maassen}@esciencecenter.nl

**Abstract.** In this paper we discuss our efforts in "unlocking" the Long Term Archive (LTA) of the LOFAR radio telescope using the software ecosystem developed in the PROCESS project. The LTA is a large (> 50 PB) archive that expands with about 7 PB per year by the ingestion of new observations. It consists of coarsely calibrated "visibilities", i.e. correlations between signals from LOFAR stations. Converting these observations into sky maps (images), which are needed for astronomy research, can be challenging due to the data sizes of the observations and the complexity and compute requirements of the software involved. Using the PROCESS software environment and testbed, we enable a simple point-and-click-reduction of LOFAR observations into sky maps for users of this archive. This work was performed as part of the PROCESS project which aims to provide generalizable open source solutions for user friendly exascale data processing.

**Keywords:** Radio astronomy, imaging, extreme large scale data processing, PC clusters, distributed computing, grid, cloud computing

**Mathematics Subject Classification 2010:** 68-04

## 1 INTRODUCTION

The LOw Frequency ARray (LOFAR) [40] is a European radio telescope which covers frequencies between 10 and 250 MHz. Designed by ASTRON [18], it became operational in 2010, and its design differs from classical radio telescopes that usually consist of arrays of dishes. Instead, LOFAR combines the signals from a large

number of relatively simple omnidirectional antennas, shown in Figure 1. These antennas are grouped into stations, each typically consisting of 96 Low Band Antennas (10 MHz–90 MHz) and 48 High Band Antennas (110 MHz–250 MHz).

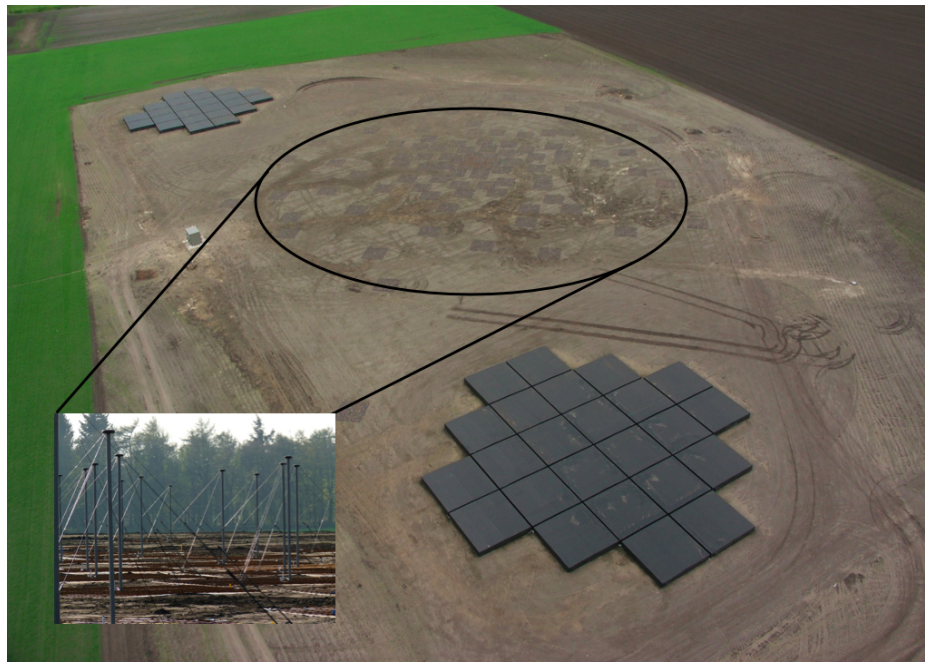


Figure 1. LOFAR LBA (poles) and HBA (boxes) antenna types forming a single station (image courtesy of ASTRON)

The signals received by these antennas are combined into a single station signal, similar to the signal of a single dish from a classical radio telescope. LOFAR currently consists of 52 stations in total (as shown in Figure 2): 24 core stations located within a 2 km radius near the village of Exloo in the east of the Netherlands, 14 additional stations in the Netherlands arranged in an (approximated) logarithmic spiral distribution, and 14 international stations located in Germany, France, Sweden, UK, Poland, Ireland and Latvia and (in future) Italy.

For observations the station signals are correlated per pair of two stations, called a *baseline*, following the principles of aperture synthesis. Every pair of signals, each consisting of a sequence of complex numbers, is multiplied with each other and a complex phase (which determines the direction of the observation), and integrated over the time sampling interval. Depending on the combination of stations used, LOFAR supports baselines from a few hundred meters to several thousand kilometers. While longer baselines provide a higher angular resolution, they complicate signal calibration as ionospheric disturbances vary more over longer distances. Every LOFAR imaging observation results in a large set of such correlations, called


Figure 2. LOFAR station distribution over Europe, including planned LOFAR stations (image courtesy of ASTRON)

visibilities. A visibility is recorded as a complex number for each baseline, frequency and time sampling interval, and polarization product.

The LOFAR Long Term Archive [36] (LTA) was set up to store all LOFAR observations. A typical LOFAR observation takes 8–12 hours and has a size of about 100 TB. Frequency averaging of every eight channels reduces this size to about 16 TB. Initial (coarse) calibration is also applied to improve the signal quality. The last few years the LTA has been expanding by about 7 PB per year and is currently (2020) exceeding 50 PB. The LOFAR LTA is stored on tapes at locations in Amsterdam (The Netherlands), Jülich (Germany) and Poznan (Poland).

LOFAR science drivers are condensed in six key science projects [13] (KSPs): the epoch of reionisation, deep extragalactic surveys, transient sources, ultra-high energy cosmic rays, solar science and space weather, and cosmic magnetism. In addition, the LOFAR data is publicly available for other uses.

The coarsely calibrated observations stored in the LTA are not directly suitable as a starting point for scientific research. KSP-specific processing pipelines are needed to further refine the observations into science products, using different combinations of processing tools and parameters. Each processing step is complex and usually requires both domain and software knowledge to generate useful output. Combined with the massive volumes of the data, further processing of the data

within the LTA is hard for non-experts. These challenges are exacerbated further when one needs to process not just one, but many observations.

In this paper, we describe our efforts to build a user-friendly "point-and-click-processing" system for the users of LTA data: after selecting an observation, an appropriate pipeline and (optionally) a set of parameters, acquiring a well-calibrated sky map just requires waiting for the processing to complete. Staging the data (copying from magnetic tape to disk), transferring this data to a suitable compute infrastructure, launching the processing pipeline, and retrieving the results, is all handled automatically by the platform. We focus on the pipeline producing sky maps, as these typically serve as a starting point for astronomy research. However, we believe the approach is generalizable to other pipelines used for processing LTA data.

This research was conducted as one the five use cases of the EU H2020 PROCESS project [21]. In Europe, we currently do not have any exascale supercomputers. Therefore, any form of processing requiring exascale data processing will have to be distributed over a number of clusters in Europe. Such distributions over many clusters will have to be performed seamlessly and all software packages that run the computations will have to be containerized to guarantee portability. The goal of PROCESS is to offer exascale computing service prototypes to a range of scientists that require big storage and big compute facilities, in such a way that these users can remain mostly agnostic of the location and specifications of the compute clusters where their data will be processed. Portability and scalability are the main requirements for the PROCESS software infrastructure, not only with respect to compute, but also with respect to data access.

## 2 RELATED WORK

One of the KSPs of the LOFAR telescope is to conduct deep wide-field surveys. For instance, the LOFAR Two Meter Sky Survey (LoTSS) [38] is observing 3 000 different fields that will collectively map the entire northern radio sky and create a total of 48 Petabytes of raw observation data that will be stored in LTA. Typically each dataset is 16 TB and is split into 244 files of 65 GB. This data is further processed through the LOFAR imaging pipeline. To complete the LoTSS survey in the project's target five year duration, multiple datasets need to be processed on a daily basis. To cope with the challenge, the LoTSS community has built the LRT (LOFAR Reduction Tools) framework [33] that provides automation, portability, scalability and generalisation. LRT is built on top of a work distribution environment which dispatches LOFAR pre-processing on a computing cluster [35] using a PiCaS server [19] to track progress. Both because of this legacy and the required high transfer rates, the platform has to run on a large computing infrastructure connected to the LTA with high-speed network, which limits its use to that infrastructure. Furthermore, the parallelisation only concerns a single step in each of the calibrator and the target pipelines.

In an extension of the work described above, the same authors present AGLOW or Automated Grid-enabled LOFAR Workflows [34]. AGLOW is a workflow orchestration system that integrates LOFAR processing with a distributed computing platform. It uses Dutch Grid infrastructure and is based on Apache Airflow [2]. According to the authors, AGLOW allows to reduce the setup of complex workflows from months to days. Both contributions are focused on lowering the data reduction time by means of distributed computing. While this was a desirable feature for our use case, our focus is more concerned with ease of use and portability of existing pipelines.

Other authors [37] have investigated the viability of the cloud as infrastructure for processing LOFAR calibration pipeline as opposed to the more traditional dedicated clusters and the grid. They found that while the cloud presents some advantages such as the ease of software installation and maintenance and the automatic scale-out, the most interesting ones, the commercial platforms, are also more expensive that the use of a dedicated cluster for large datasets. The cloud solution is only competitive if the number of datasets to be analysed is not high, which disqualifies it, for instance, for surveys KSP. Furthermore, the pipeline tools used for the tests do not include the most recent developments such as *FACTOR* or DDF.

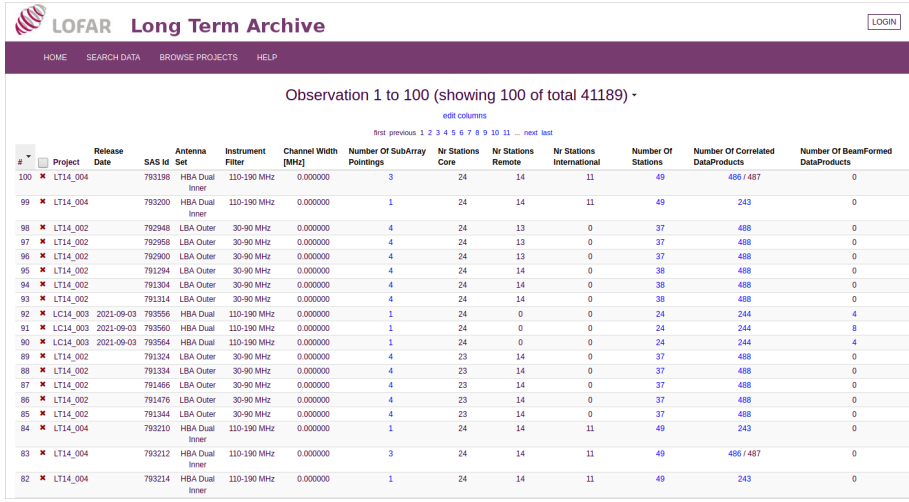
## 3 BACKGROUND

This section consists of two parts where we lay down the foundations for understanding the science case behind our use case and the environment in which it is implemented. First, we describe the LOFAR imaging pipeline with enough details to understand the choices made for our use case implementation in PROCESS. Next, we briefly summarise the PROCESS project and describe its ecosystem components and architecture.

### 3.1 The LOFAR Imaging Pipeline

Astronomers often embark on a scientific investigation by inspecting sky maps. For example, after the detection of a cosmic explosion by a gamma-ray satellite at a particular position on the sky, an astronomer will want to find out which celestial objects are visible near this position on the sky at other wavelengths, like radio. This is where a repository with low frequency radio maps covering a large part of the sky can provide a useful resource. Unfortunately, such sky maps are currently only available for a fraction of the LOFAR observations stored in the LTA (as produced by the LOFAR Two Meter Sky Survey (LoTSS) [38] for example).

When an astronomer wants to produce new sky maps, data first needs to be downloaded from the LTA. ASTRON provides a convenient web portal, shown in Figure 3, which allows users to search through the available data and select the observations which need to be retrieved from tape. Once the data is retrieved, it must be downloaded from temporary disk storage at the LTA to the users own infrastructure. The size of these datasets can be significant, up to 16 TB per observation.




| # | Project | Release Date | SAS Id | Antenna Set | Instrument Filter | Channel Width [MHz] | Number Of SubArray Pointings | Nr Stations Core | Nr Stations Remote | Nr Stations International | Number Of Stations | Number Of Correlated DataProducts | Number Of BeamFormed DataProducts |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 100 | LT14_004 | | 793198 | HBA Dual Inner | 110-190 MHz | 0.000000 | 3 | 24 | 14 | 11 | 49 | 486 / 487 | 0 |
| 99 | LT14_004 | | 793200 | HBA Dual Inner | 110-190 MHz | 0.000000 | 1 | 24 | 14 | 11 | 49 | 243 | 0 |
| 98 | LT14_002 | | 792948 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 24 | 13 | 0 | 37 | 488 | 0 |
| 97 | LT14_002 | | 792958 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 24 | 13 | 0 | 37 | 488 | 0 |
| 96 | LT14_002 | | 792900 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 24 | 13 | 0 | 37 | 488 | 0 |
| 95 | LT14_002 | | 791294 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 24 | 14 | 0 | 38 | 488 | 0 |
| 94 | LT14_002 | | 791304 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 24 | 14 | 0 | 38 | 488 | 0 |
| 93 | LT14_002 | | 791314 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 24 | 14 | 0 | 38 | 488 | 0 |
| 92 | LC14_003 | 2021-09-03 | 793556 | HBA Dual | 110-190 MHz | 0.000000 | 1 | 24 | 0 | 0 | 24 | 244 | 4 |
| 91 | LC14_003 | 2021-09-03 | 793560 | HBA Dual | 110-190 MHz | 0.000000 | 1 | 24 | 0 | 0 | 24 | 244 | 8 |
| 90 | LC14_003 | 2021-09-03 | 793564 | HBA Dual | 110-190 MHz | 0.000000 | 1 | 24 | 0 | 0 | 24 | 244 | 4 |
| 89 | LT14_002 | | 791324 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 23 | 14 | 0 | 37 | 488 | 0 |
| 88 | LT14_002 | | 791334 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 23 | 14 | 0 | 37 | 488 | 0 |
| 87 | LT14_002 | | 791466 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 23 | 14 | 0 | 37 | 488 | 0 |
| 86 | LT14_002 | | 791476 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 23 | 14 | 0 | 37 | 488 | 0 |
| 85 | LT14_002 | | 791344 | LBA Outer | 30-90 MHz | 0.000000 | 4 | 23 | 14 | 0 | 37 | 488 | 0 |
| 84 | LT14_004 | | 793210 | HBA Dual Inner | 110-190 MHz | 0.000000 | 1 | 24 | 14 | 11 | 49 | 243 | 0 |
| 83 | LT14_004 | | 793212 | HBA Dual Inner | 110-190 MHz | 0.000000 | 3 | 24 | 14 | 11 | 49 | 486 / 487 | 0 |
| 82 | LT14_004 | | 793214 | HBA Dual Inner | 110-190 MHz | 0.000000 | 1 | 24 | 14 | 11 | 49 | 243 | 0 |

Figure 3. The LTA web interface for searching and downloading observation data, accessible through `https://lta.lofar.eu/Lofar` (image courtesy of ASTRON)

To produce images from this observation data, a large number of processing steps need to be performed, as shown in Figure 4 (in a simplified form). This pipeline is generally referred to as the Standard Imaging Pipeline (SIP) [16].

The initial steps, known as the Default Pre-Processing Pipeline (DPPP) [5], constitute (among other things) of *flagging* the data to remove radio frequency interference (RFI), optionally *averaging* to reduce the data volume, and *demixing* to subtract the contributions of the brightest sources in the sky to increase the sensitivity.

Next, an initial set of calibration parameters is applied. To do so, a short observation of a reference source (the *calibrator*) is performed immediately preceding or succeeding the main observation of the *target* (the astronomical source of interest). A Local Sky Model (LSM) which matches the area of interest is then extracted from the LOFAR Global Sky Model (GSM). The GSM is a "ground truth" database containing all known sources from various sky survey catalogs, including VLSS [28], WENSS [30], TGSS [31]. Using the LSM and calibrator observation in an iterative process, an estimate can be obtained for instrumental and environmental effects such as electronic station gains and ionospheric delays. The target observation can then be corrected for these effects, a step generally referred to a Direction Independent calibration (DI).

The calibrated data are then converted into an image using an imager that applies the w-projection algorithm [29] to remove the effects of noncoplanar baselines when imaging large fields and the A-projection algorithm [39] to take into account the varying primary beam during synthesis observations. The LSM is expanded and updated in the process by extracting sources from the images. One or more loops

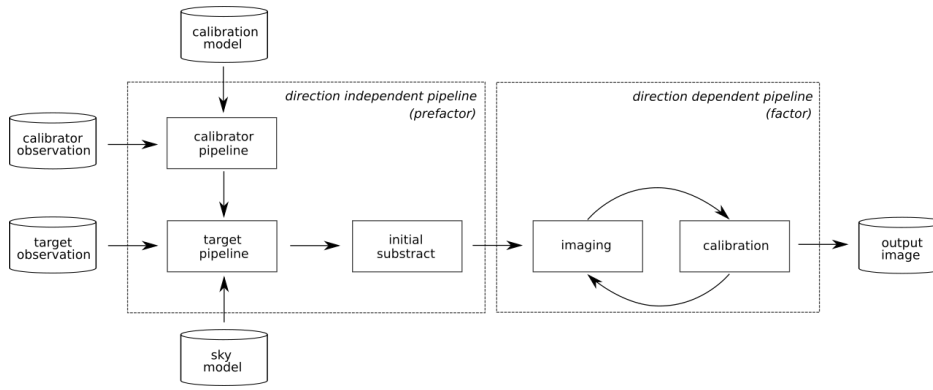



Figure 4. Imaging pipeline main steps

of calibration, imaging and LSM updates are performed. At the end of the process, the final LSM will be used to update the GSM, and the final images are generated. Various types of calibration algorithms can be used in this process, depending on the requirements and user preferences.

Creating such an imaging pipeline is complex and requires detailed knowledge of the domain, tools and pipeline framework (not to mention significant programming skills). Therefore, a *genericpipeline* [9] is offered by ASTRON, which helps astronomer design their processing pipeline without requiring too much technical knowledge. This *genericpipeline* contains predefined pipeline steps for the user to choose from. Creating a new pipeline then boils down to defining a so-called *parset*; a parameter set (or pipeline definition) which selects and configures the relevant steps in the *genericpipeline*.

A tool commonly used in this process is *prefactor* [20] which consists of various *parsets* for the *genericpipeline* to steer the processing of LOFAR data. Originally intended to prepare the data for input to the direction-dependent (DD) calibration software *FACTOR* [7] (hence, its name), *prefactor* performs the steps described above to correct for various instrumental and ionospheric effects on observations, and makes the observation ready for more advanced DD calibration pipelines, such as FACTOR or *killMS*.

In this paper, we decided to use *FACTOR*, as we found it to be one of the most stable tools available for DD calibration. It produces low-noise, high-resolution images from HBA LOFAR data using the facet calibration scheme [41]. *FACTOR* corrects for direction-dependent effects, including ionospheric effects and beam-model errors. *FACTOR* works by dividing up the target observation field into many facets and separately solving for the direction-dependent corrections in each facet. It is designed to minimize the number of free parameters needed to parameterize these corrections to avoid overfitting. This minimization is critical in producing high-fidelity images.

While the available tools are generally well documented and several LOFAR imaging tutorials can be found online, the overall process is quite cumbersome for non-expert users. A large number of tools must be installed and configured, sometimes resulting in complex technical or software dependency problems. Once the tools are installed successfully and configured correctly, the data volumes that need to be transferred and processed are significant and often exceed the capabilities of the infrastructure available to the users. Therefore, a user-friendly "point-and-click-processing" system for the users of LTA data could significantly lower the threshold for using LOFAR LTA data for the non-expert users. In Section 4, we describe how we have implemented such a system as one the five use cases using the software environment developed in the PROCESS project, which is briefly described below.

### 3.2 PROCESS

The aim of the PROCESS project [21] is to provide an open-source, multi-purpose and scalable software environment specially developed for exascale data processing. This goal was achieved by creating various tools and services that support set of heterogenous extreme scale data processing use-cases driven by both the scientific research community and industry [22].

Although these use cases come from very different communities (medical imaging, radio astronomy, airline ancillary pricing, disaster risk management and earth observation), they share the same problems:

1. they need to process very large volumes of data using a diverse collection of tools,
2. these tools are difficult to install and configure by the users who often lack the necessary technical knowledge, and
3. the storage and/or compute requirements exceed the capabilities of the infrastructure that is available to the users.

During the course of the PROCESS project, a modular service architecture was designed and implemented [26], a simplified schematic of which is shown in Figure 5. It can be divided into three main modules: the Interactive Execution Environment [10] (IEE), which provides a web-based user API (as well as a REST API) for submitting pipelines, the LOBCDER data module [12], which offers distributed storage and data transfer services, and the compute module which provides access to both HPC and Cloud compute infrastructure (via RimRock [23] and Cloudify [3], respectively). For testing purposes, these services are deployed on compute and storage infrastructure at Cyfronet[1] and LRZ[2].

[1] `http://www.cyfronet.krakow.pl/en`

[2] `https://www.lrz.de/english`

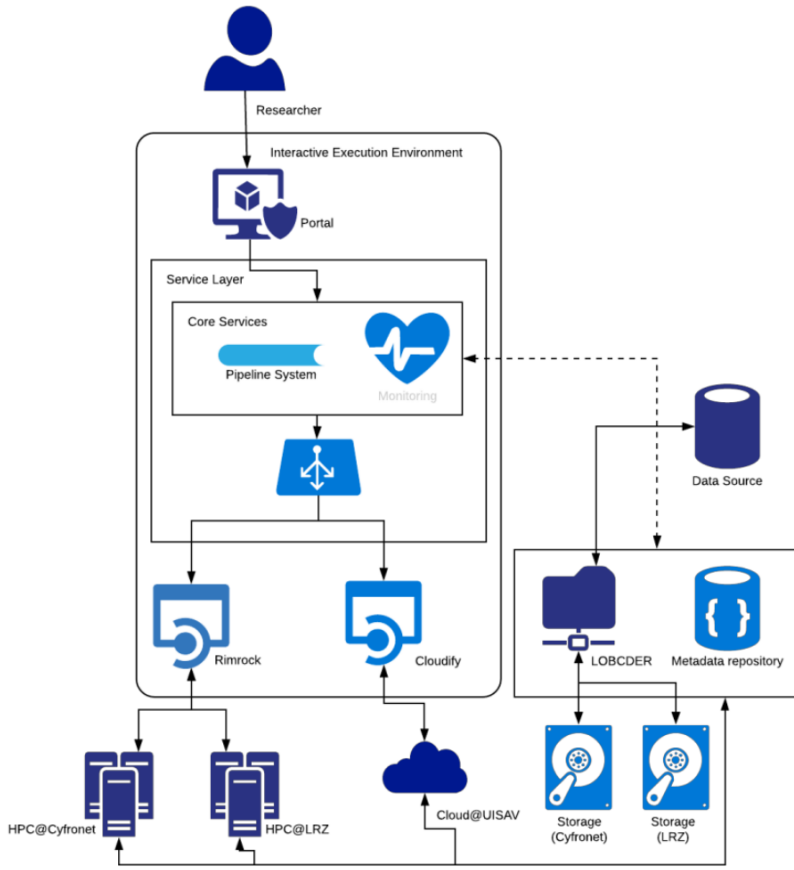



Figure 5. PROCESS platform architecture (image courtesy of [27])

To run a processing pipeline, the user will need a containerized version of the necessary tools. Currently, only Singularity [32] containers are supported. Creating such a containerized version of the tools may be too complex for the average user. However, there is a current trend among tool developers to provide such containers themselves, as this is seen as an easy solution to the dependency problems often encountered by users when installing software.

Using the IEE, a processing pipeline can be defined based on the containerized tools and then submitted to the compute infrastructure (either based on HPC or Cloud technology). Before the processing starts, the IEE will use the LOBCDER data module to transfer the necessary input data from the source location to the selected compute infrastructure.

The architecture designed by PROCESS provided us with the necessary services and infrastructure to create a "point-and-click" solution for the LTA imaging pipeline, which will be described in the next section.

## 4 POINT-AND-CLICK PROCESSING IMPLEMENTATION

To create an easy to use solution for the LTA imaging pipeline, the requirement analysis showed that three components need to be in place:

- A use-case specific web user interface, that enables users to select the desired datasets and processing pipelines.
- Containerized versions of the LOFAR imaging tools.
- Data services for retrieving the observation data from tapes at the LTA (staging), transferring this data to compute infrastructure, and the extraction of resulting images.
- Compute services to run the containerized LOFAR imaging tools on the compute infrastructure.

The solution we have implemented using the PROCESS services is shown in Figure 6, and fulfills all of these requirements. We will first provide an overview of how these components interact and then describe each of the components in more detail below.

At the startup of the web application, the backend connects to the database at the LTA archive and extracts a list of all accessible observations (*step 1*). Next, it retrieves the list of pre-configured pipelines available to the user from local storage (*step 2*). Both are then presented to the user in the web frontend, which is described in more detail in Section 4.1.

Once the user has selected a suitable target and calibrator observation (*step 3*), all necessary information on the selected observations, the pipeline, and various parameters are submitted to the IEE (*step 4*, described in Section 4.2). Before the IEE can execute the pipeline on the compute infrastructure, however, the observation data must be retrieved from the LTA archive. To do so, the IEE requests that the LOBCDER data service to retrieve the necessary data from the LTA (*steps 5-8*, explained in Section 4.3).

Once the data is available, the IEE will execute the pipeline on the compute infrastucture (*steps 9 and 10*) using a containerised version of the pipeline (Section 4.4). Finally, the result is returned to the the user via the web frontend (*steps 11 and 12*, Section 5).

### 4.1 Web Frontend

To enable easy access to the data processing infrastructure, we decided to create a use-case specific web portal, based on LTACAT [14], which was developed earlier

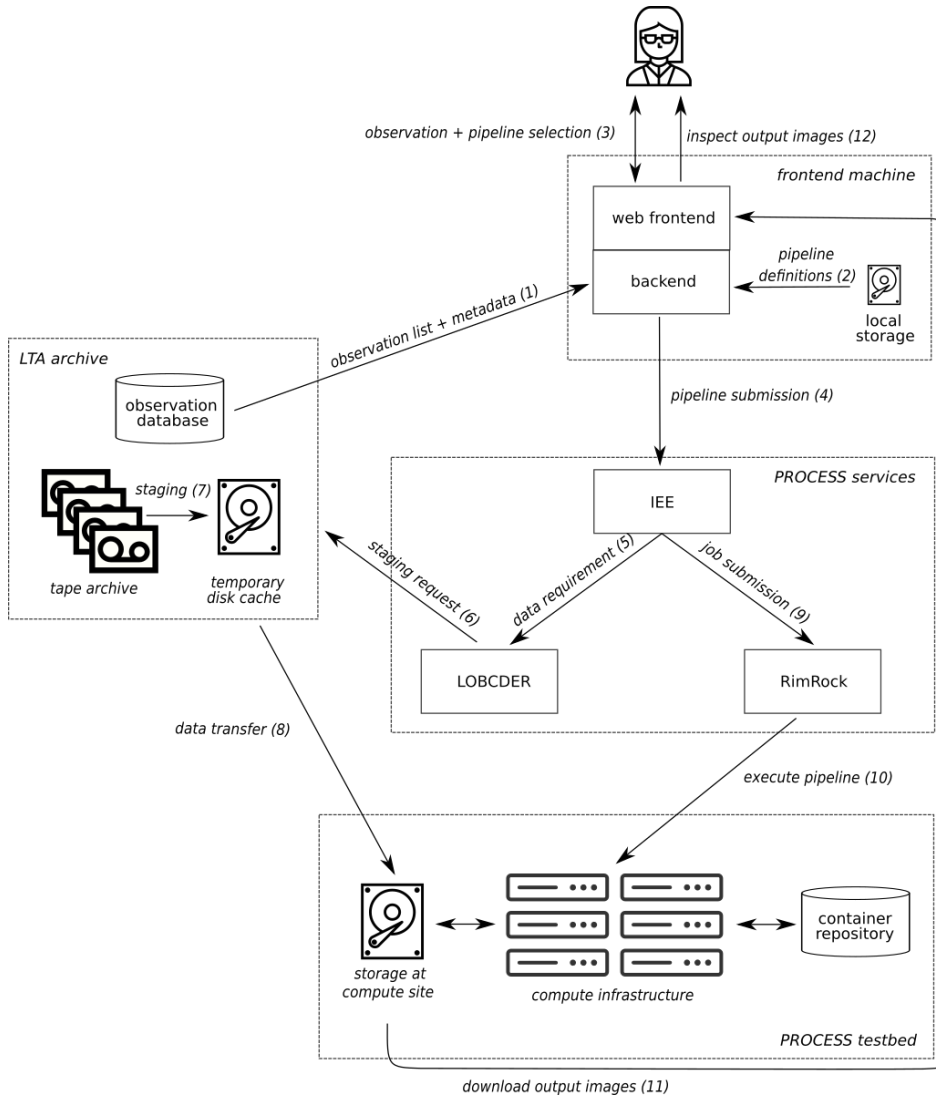



Figure 6. Schematic workflow of interaction between the images processing web application, PROCESS services, and the LTA archive

in the EOSC pilot for LOFAR project [6]. The latter is a React[3] application based on FRBCAT [8] (originally developed as a catalogue for fast radio bursts (FRBs) in the AA-ALERT [1] project). This web application was customised and extended to fit LOFAR LTA imaging pipeline needs, and can be found at [15].

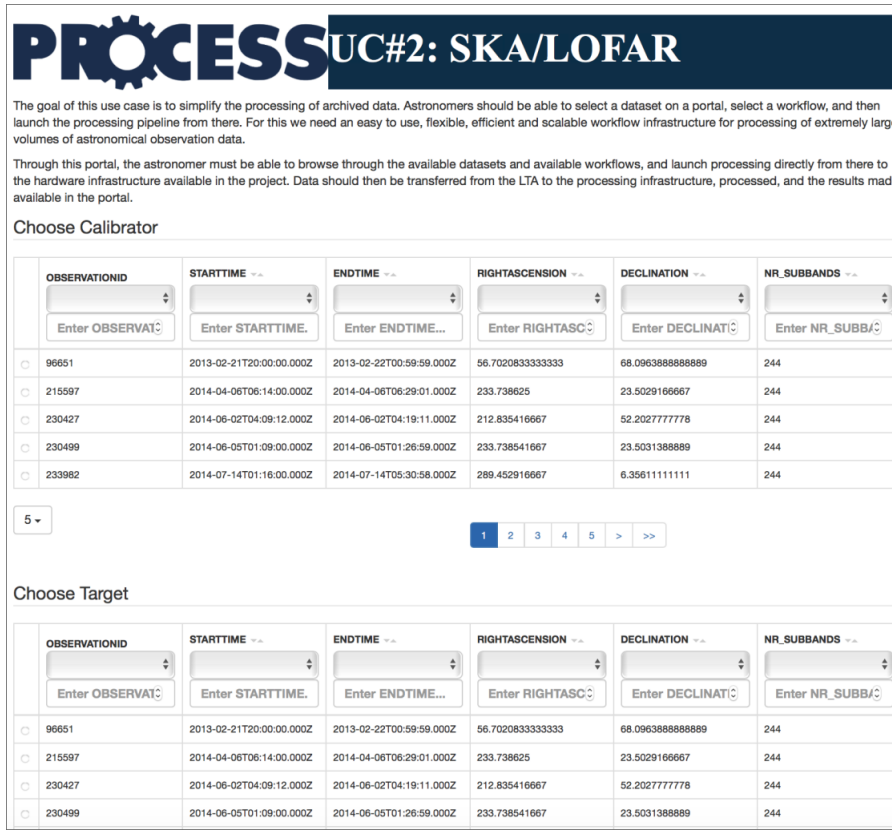



Figure 7. Main LTA database view

The LTA database main view, shown in Figure 7, shows information and metadata about LOFAR observations, similar to the original LTA web portal. As shown in Figure 6, the backend directly connects to the LTA database allowing users to seamlessly access the observation data archived in the LTA. Access to some observations may be restricted, however, due to various policies. Instead of providing a unified view of all data (as the original LTA web portal offers), the user is presented with two lists in order to select the calibrator and the target observations separately. The metadata can be used to filter the selection.

[3] `https://reactjs.org`

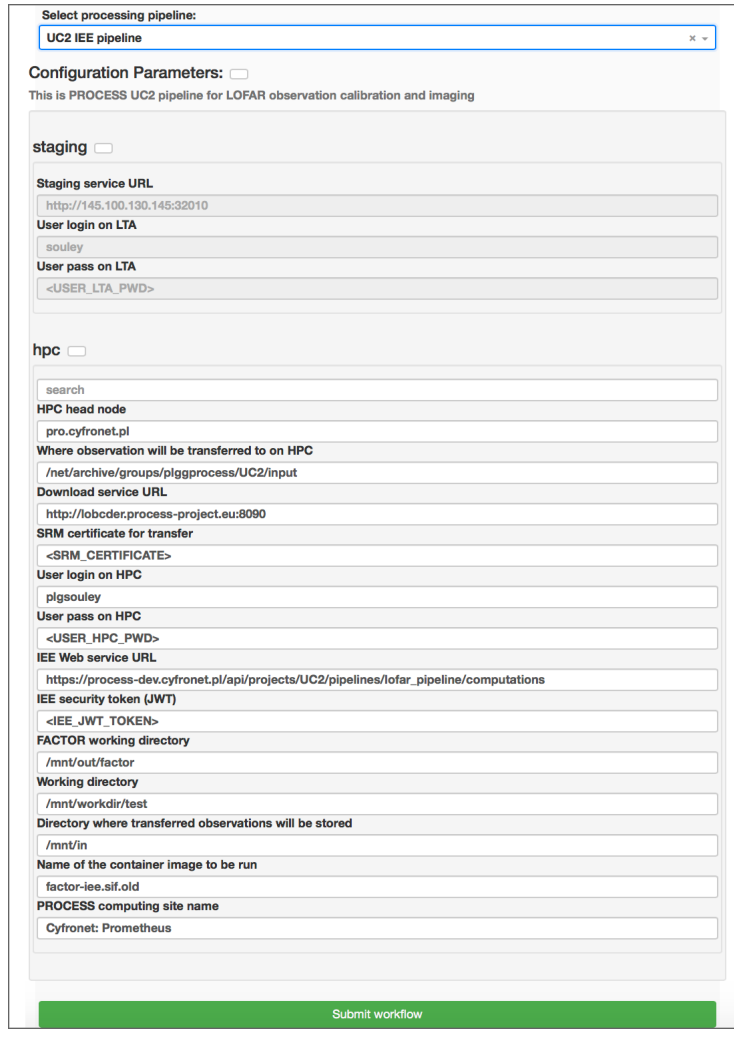



Figure 8. Pipeline configuration and submission

Once the user selects a calibrator and target observations, the web application provides a separate window to select (and optionally configure) a processing pipeline, as shown in Figure 8. This window provides a list of available pipelines, the configuration parameters for the selected pipeline, and a submit button which will submit the pipeline execution to the IEE.

Currently, the definitions of these pipelines are stored locally on the machine running the Web application. In the backend, the pipeline configurator is automatically generated from a *JSON schema* [11] describing each pipeline. This allows for

predefined default values, constraints on inputs, mandatory required properties, and defining dependences between properties.

Each definition already includes suitable defaults for the pipeline parameters, such as configuration settings for the processing steps and the necessary configuration for the data retrieval and the choice of a computing site. Once set correctly, the users does not need to adjust these. Suitable defaults are provided for the current PROCESS testbed, which will retrieve data from the LTA and use the HPC facilities at Cyfronet as the compute site. If needed, these values can be adjusted in the *staging* and *hpc* sections of the pipeline configuration shown in Figure 8.

As part of the PROCESS testbed, the web portal comes bundled with a single predefined pipeline, but it is easily extended. It allows users to develop additional pipelines and integrate them in the Web application. For this purpose a pipeline template is provided in addition to a step-by-step guide on how to integrate new pipelines into the service [17]. The procedure consists of implementing a *run* function, defining the pipeline configuration parameters in *JSON schema* format, and registering the pipeline in the pipeline administrator of the IEE. After installation of the new pipeline based on these steps, a new pipeline appears in the list of available pipelines.

### 4.2 Pipeline Submission to the IEE

Once the pipeline is selected (and optionally configured), the user can submit it to the IEE for execution. The integration between the web portal and the IEE consists of REST API calls for retrieving the pipeline configuration parameters and submitting pipeline computations to the IEE based on the expected parameters.

When a pipeline is submitted, the LTA identifiers of the required target and calibrator observations are provided to the IEE as parameters of the pipeline. Before actual the actual pipeline execution begins, the IEE requests LOBCDER to fulfill the data requirements of the pipeline and ensure the data of both observations are available on the compute site. LOBCDER is described in more detail in the next section. This step may be skipped if the data already resides on the compute site.

Once the data is available, the pipeline job will be scheduled on the compute infrastructure. This job consists of a containerized version of the pipeline (described in Section 4.4 to ensure portability. This job submision is performed through RimRock, which provides a REST API to the underlying scheduling system of the HPC compute cluster.

Once running, the web portal will retrieve the pipeline status from the IEE using the REST API calls. Once the pipeline has completed, the IEE will provide a link to where the results can be retrieved.

### 4.3 LOBCDER Data Services

Before the pipeline can be executed, the observation data needs to be available on the compute site. Consequently, the very first action of the IEE when receiving

a pipeline submission is to call the LOBCDER data services to request that the target and calibration observations are retrieved from the LTA archive. LOBCDER in turn contacts the LTA to request that the observational data is retrieved from the tapes and stored at a temporary location (a process referred to as "staging"). Once this staging is complete, LOBCDER will transfer the data to the selected compute site.

Therefore, to satisfy the data service requirements of our use case, several endpoints have been created by the LOBCDER team. One for issuing a staging command and one to check its status. Additionally, another one to transfer the staged in data from the temporary location to a HPC cluster for processing, along with its corresponding status check command. These are incorporated into the IEE portal using a Python module. The parameters for configuring these service endpoints are exposed by the JSON schema for the pipeline configuration.

Once the transfer is completed, LOBCDER notifies IEE which then submits the job which will run the pipeline to the workload management system on the selected computing resource(s).

### 4.4 Containerized Analysis Pipeline

To ensure portability, we have created a containerized version of the LTA imaging pipeline based on CWL [25] and Singularity[4]. This container essentially implements the *FACTOR* pipeline shown in Figure 4. The first two steps (*calibrator* and *target*) are taken care of by *prefactor* which provides parsets for each of them. They provide direction independent calibration. *prefactor* also provides a parset for the third step, *subtract*, which is specific to *FACTOR*. This step images the field at medium and low resolution to make initial models of the sources and subtracts these models from the *uv* data. The last step is *FACTOR* itself performing direction dependent calibration and imaging of HBA data. It divides the field into facets based on bright direction-dependent calibrators. It then cycles over the facets to self calibrate the calibrator sources (*facetselfcal*) and to improve the subtraction with new model and calibration (*facetsub*). The facets are then imaged (*facetimage*). Finally, *FACTOR* makes a mosaic of all facets and corrects for the primary beam attenuation (*fieldmosaic*).

For the integration of the container with IEE, we need to provide an appropriate *run script* within the container. This script passes the expected input parameter values provided to the container by the IEE to the workflow runner (*cwl-runner*) running inside the container. These parameters are provided by the user via the frontend shown in Figure 8. They are passed to IEE through the REST API calls described above. Currently, the container is stored online at a private location, from where it is currently manually downloaded and installed at the computing site by the IEE team. We aim to automate this process by storing the container in a registry so it can be automatically retrieved, as shown in Figure 6.

[4] `https://www.sylabs.io`

Once the processing pipeline has completed, the output is made available for download by the IEE via a download link. This output consists of the output images in FITS format[5], which is commonly used in astronomy, plus inspection plots (examined by the astronomers to check the normal functioning of various parts of the instrument) and various log files produced by the processing steps in the pipeline. The full resolution output images can be downloaded by the user. For convenience, downscaled JPG thumbnails of the images are presented in the frontend.

## 5 RESULTS

With our use case fully implemented using the PROCESS services, generating images from the LTA data has never been so easy: the astronomer has just to launch the frontend, choose its calibrator, target and pipeline, and click a button. The PROCESS platform takes care of all the processing. The user can then use his/her valuable time for more analytical tasks while waiting for the images. Sample output images generated by this pipeline are shown in Figure 9.

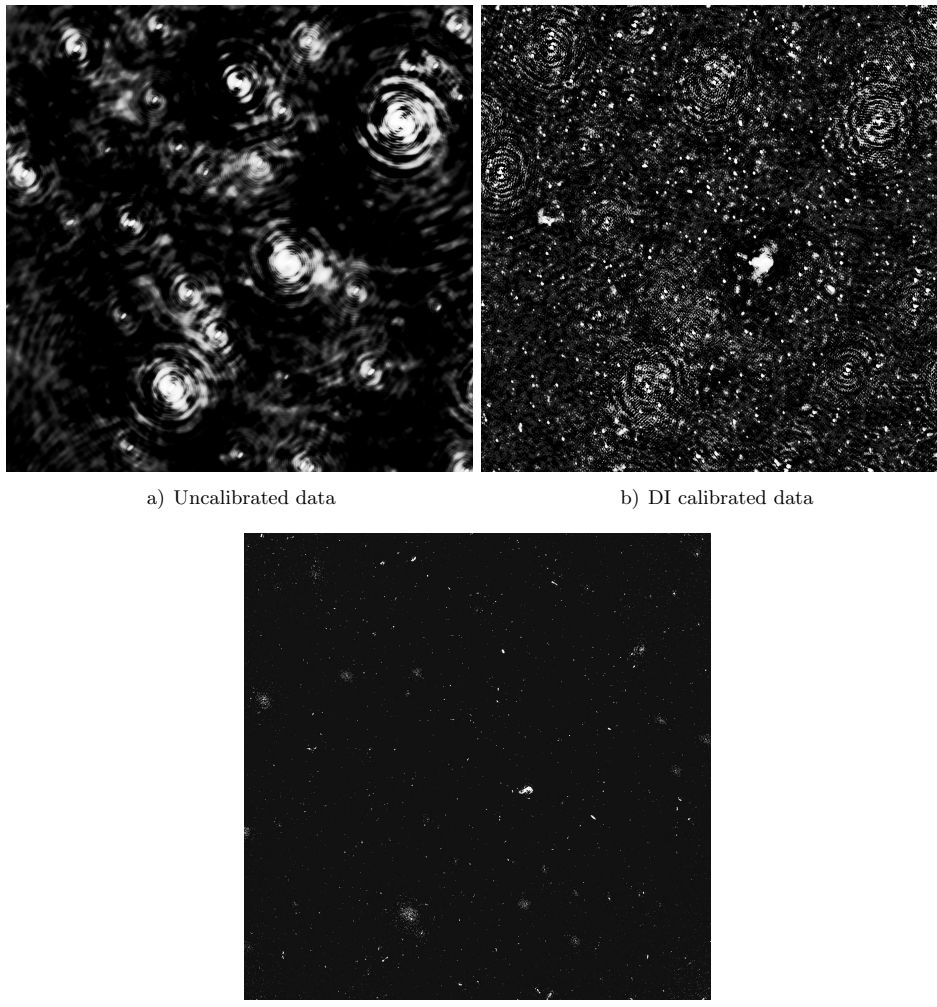
Figure 9 a) shows the image that will be produced when directly using the data from the LTA archive. This data is only roughly calibrated, and contains significant distortions caused by ionospheric interferences and instrument effects. Figure 9 b) shows the data after the initial processing, demixing of bright sources, and direction independent calibration, which already improves the image quality significantly. Figure 9 c) shows the final result after direction dependent calibration, which further improves the image quality to the level required for astronomy research.

| Step | Data Size [GB] | Run Time [s] |
|---|---|---|
| calibrator | 25 | 8 534 |
| target | 433 | 11 902 |
| subtract | 76 | 37 212 |
| FACTOR | 76 | 464 400 (∼ 5d9h) |

Table 1. Breakdown of a test run of the LTA imaging pipeline using the PROCESS services and infrastructure

In Table 1, we show a breakdown of the average execution time of each step in the *FACTOR* pipeline. This pipeline is running on a test dataset consisting of twenty sub-bands (out of 144) of both the calibrator and target data, and processes about 450 GB of data. The end-to-end execution time of the pipeline is about 6 days. As the table shows, the processing time is dominated by the direction dependent calibration performed by *FACTOR*, which takes about 89 % of the overall time required.

[5] `https://fits.gsfc.nasa.gov/fits_standard.html`



a) Uncalibrated data

b) DI calibrated data

c) DD calibrated data

Figure 9. Results of imaging uncalibrated, DI calibrated and DD calibrated data

## 6 CONCLUSION AND FUTURE WORK

In this paper, we discussed our efforts in "unlocking" the LOFAR LTA using the software ecosystem developed in the PROCESS project. We described the motivation for our use case and analysed its requirements. We succinctly described the science case behind it and briefly presented the PROCESS project services and tools. Then we showed that the solution for our use case can be straightforwardly implemented using the PROCESS services and tools. Finally, we showed an example

of the sky maps generated using that solution and the time needed to reach those results.

One of the nonfunctional requirements identified for our use case is horizontal scalability that allows the processing of several observations in parallel, and potentially on different compute sites. This feature would be very useful for the Surveys KSP for instance, as they typically require a large amount of processing. Although, theoretically, IEE can submit to several computing sites concurrently, in practice, it can currently only submit to its local computing site, Prometheus. As our future work, we hope to extend this to multiple sites.

In addition, all processing is currently performed on a single node. Parallelism is limited to the number of cores available within this node. It would be interesting to revisit these processing steps and reimplement them using more scalable approaches. As the Direction Dependent calibration step is the most compute-intensive, it would be beneficial to add alternative approaches to *FACTOR* such as the DDF pipeline [4] or SAGECal [24], which may support better parallelisation schemes.

**Acknowledgements**

This work is supported by the "PROviding Computing solutions for ExaScale ChallengeS" (PROCESS) project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 777533.

## REFERENCES

[1] AA-ALERT: Access and Acceleration of the Apertif Legacy Exploration of the Radio Transient Sky. `https://www.esciencecenter.nl/project/aa-alert`, accessed: 2020-09-17.

[2] Apache Airflow. `https://airflow.apache.org`, accessed: 2020-09-09.

[3] Cloudify: Multi-Cloud Orchestration. `https://cloudify.co/`, accessed: 2020-09-10.

[4] Ddf-Pipeline (GitHub). `https://github.com/mhardcastle/ddf-pipeline`, accessed: 2020-09-09.

[5] The Default Pre-Processing Pipeline (DPPP). `https://support.astron.nl/LOFARImagingCookbook/dppp.html`, accessed: 2020-09-09.

[6] The European Open Science Cloud for Research Pilot Project. `https://eoscpilot.eu`, accessed: 2020-09-09.

[7] Factor: Facet Calibration for LOFAR. `https://www.astron.nl/citt/facet-doc/index.html`, accessed: 2020-09-09.

[8] FRB Catalogue. `http://www.frbcat.org`, accessed: 2020-09-17.

[9] Generic Pipeline. `https://www.astron.nl/citt/genericpipeline/`, accessed: 2020-09-09.

[10] The Interactive Execution Environment (IEE). https://gitlab.com/cyfronet/iee, accessed: 2020-09-10.

[11] JSON Schema. https://json-schema.org/, accessed: 2020-09-09.

[12] The LOBCDER Data Services. https://github.com/micro-infrastructure/mini-lobcder, accessed: 2020-09-10.

[13] LOFAR Key Science Projects. http://www.lofar.org/astronomy/key-science/lofar-key-science-projects.html, accessed: 2020-09-09.

[14] LOFAR Long Term Archive Pipeline Orchestrate Web Application. https://github.com/EOSC-LOFAR/ltacat, accessed: 2020-09-09.

[15] LOFAR LTA Pipeline Orchestrate Web Application. https://github.com/process-project/ltacat_UC2, accessed: 2020-09-09.

[16] LOFAR Wiki: Standard Imaging Pipeline. https://www.astron.nl/lofarwiki/doku.php?id=public:user_software:documentation:standard_imaging_pipeline, accessed: 2020-09-09.

[17] LTA Processing Pipeline Template and Guide. https://github.com/process-project/UC2_pipeline, accessed: 2020-09-09.

[18] Netherlands Institute for Radio Astronomy. https://www.astron.nl, accessed: 2020-09-09.

[19] Picas Overview: http://doc.grid.surfsara.nl/en/latest/Pages/Practices/picas/picas_overview.html, accessed: 2020-09-09.

[20] Prefactor: Preprocessing for Facet Calibration for LOFAR. https://www.astron.nl/citt/prefactor/, accessed: 2020-09-09.

[21] PROCESS: Providing Computing Solutions for Exascale Challenges. https://www.process-project.eu, accessed: 2020-09-09.

[22] PROCESS Use Case Descriptions. https://www.process-project.eu/use-cases, accessed: 2020-09-09.

[23] Rimrock – Robust Remote Process Controller. http://dice.cyfronet.pl/products/rimrock, accessed: 2020-09-10.

[24] SAGECal (GitHub). https://github.com/nlesc-dirac/sagecal, accessed: 2020-09-09.

[25] Amstutz, P.—Crusoe, M. R.—Tijanić, N.—Chapman, B.—Chilton, J.—Heuer, M.—Kartashov, A.—Leehr, D.—Ménager, H.—Nedeljkovich, M.—Scales, M.—Soiland-Reyes, S.—Stojanovic, L.: Common Workflow Language, v1.0. Figshare, 2016, doi: 10.6084/m9.figshare.3115156.v2.

[26] Bobák, M.—Hluchy, L.—Belloum, A. S. Z.—Cushing, R.—Meizner, J.—Nowakowski, P.—Tran, V.—Habala, O.—Maassen, J.—Somosköi, B.—Graziani, M.—Heikkurinen, M.—Höb, M.—Schmidt, J.: Reference Exascale Architecture. 2019 15th International Conference on eScience (eScience), San Diego, CA, USA, 2019, pp. 479–487, doi: 10.1109/eScience.2019.00063.

[27] Bubak, M.—Meizner, J.—Nowakowski, P.—Bobák, M.—Habala, O.—Hluchý, L.—Tran, V.—Belloum, A. S. Z.—Cushing, R.—Höb, M.—Kranzlmüller, D.—Schmidt, J.: A Hybrid HPC and Cloud Platform for Multidisciplinary Scientific Application. 2020 Super Computing Frontiers Europe, Virtual Global Conference, March 2020.

[28] Cohen, A. S.—Lane, W. M.—Cotton, W. D.—Kassim, N. E.—Lazio, T. J. W.—Perley, R. A.—Condon, J. J.—Erickson, W. C.: The VLA Low-Frequency Sky Survey. The Astronomical Journal, Vol. 134, 2007, No. 3, pp. 1245–1262, doi: 10.1086/520719.

[29] Cornwell, T. J.—Golap, K.—Bhatnagar, S.: W Projection: A New Algorithm for Wide Field Imaging with Radio Synthesis Arrays. In: Shopbell, P., Britton, M., Ebert, R. (Eds.): Astronomical Data Analysis Software and Systems XIV. Astronomical Society of the Pacific, San Francisco, ASP Conference Series, Vol. 347, 2005, pp. 86–90.

[30] de Bruyn, A. G.: The Westerbork Northern Sky Survey. In: Ekers, R., Fanti, C., Padrielli, L. (Eds.): Extragalactic Radio Sources. Springer, Dordrecht, International Astronomical Union, Vol. 175, 1996, pp. 495–498, doi: 10.1007/978-94-009-0295-4_180.

[31] Intema, H. T.—Jagannathan, P.—Mooley, K. P.—Frail, D. A.: The GMRT 150 MHz All-Sky Radio Survey – First Alternative Data Release TGSS ADR1. Astronomy and Astrophysics, Vol. 598, 2017, Art. No. A78, 28 pp., doi: 10.1051/0004-6361/201628536.

[32] Kurtzer, G. M.—Sochat, V.—Bauer, M. W.: Singularity: Scientific Containers for Mobility of Compute. PLoS ONE, Vol. 12, 2017, No. 5, Art. No. e0177459, 20 pp., doi: 10.1371/journal.pone.0177459.

[33] Mechev, A. P.—Oonk, J. B. R.—Danezi, A.—Shimwell, T. W.—Schrijvers, C.—Intema, H. T.—Plaat, A.—Röttgering, H. J. A.: An Automated Scalable Framework for Distributing Radio Astronomy Processing Across Clusters and Clouds. International Symposium on Grids and Clouds 2017 (ISGC 2017), Academia Sinica, Taipei, Taiwan, 2017, Art. No. 002. https://pos.sissa.it/293/002/pdf.

[34] Mechev, A. P.—Oonk, J. B. R.—Shimwell, T.—Plaat, A.—Intema, H. T.—Röttgering, H. J. A.: Fast and Reproducible LOFAR Workflows with AGLOW. 2018 IEEE 14th International Conference on e-Science (eScience), Amsterdam, Netherlands, 2018, Vol. 1, pp. 136–144, doi: 10.1109/eScience.2018.00029.

[35] Oonk, J.—Mechev, A.—Danezi, A.—Schrijvers, C.—Shimwell, T.: Radio Astronomy on a Distributed Shared Computing Platform: The LOFAR Case. 2017.

[36] Renting, G. A.—Holties, H. A.: LOFAR Long Term Archive. In: Evans, I. N., Accomazzi, A., Mink, D. J., Rots, A. H. (Eds.): Astronomical Data Analysis Software and Systems XX. Astronomical Society of the Pacific, San Francisco, ASP Conference Series, Vol. 442, 2011, pp. 49–52.

[37] Sabater, J.—Sánchez-Expósito, S.—Best, P.—Garrido, J.—Verdes-Montenegro, L.—Lezzi, D.: Calibration of LOFAR Data on the Cloud. Astronomy and Computing, Vol. 19, 2017, pp. 75–89, doi: 10.1016/j.ascom.2017.04.001.

[38] Shimwell, T. W.—Tasse, C.—Hardcastle, M. J.—Mechev, A. P.—Williams, W. L.—Best, P. N.—Röttgering, H. J. A.—Callingham, J. R.—Dijkema, T. J.—de Gasperin, F. et al.: The LOFAR Two-Metre Sky Survey. II. First Data Release. Astronomy and Astrophysics, Vol. 622, 2019, Art. No. A1, 21 pp., doi: 10.1051/0004-6361/201833559.

[39] Tasse, C.—van der Tol, S.—van Zwieten, J.—van Diepen, G.—Bhatnagar, S.: Applying Full Polarization A-Projection to Very Wide Field of View Instruments: An Imager for LOFAR. Astronomy and Astrophysics, Vol. 553, 2013, Art. No. A105, 13 pp., doi: 10.1051/0004-6361/201220882.

[40] van Haarlem, M. P.—Wise, M. W.—Gunst, A. W.—Heald, G.—McKean, J. P.—Hessels, J. W. T.—de Bruyn, A. G.—Nijboer, R.—Swinbank, J.—Fallows, R. et al.: LOFAR: The LOw-Frequency ARray. Astronomy and Astrophysics, Vol. 556, 2013, Art. No. A2, 53 pp., doi: 10.1051/0004-6361/201220873.

[41] van Weeren, R. J.—Williams, W. L.—Hardcastle, M. J.—Shimwell, T. W.—Rafferty, D. A.—Sabater, J.—Heald, G.—Sridhar, S. S.—Dijkema, T. J.—Brunetti, G. et al.: LOFAR Facet Calibration. The Astrophysical Journal Supplement Series, Vol. 223, 2016, No. 1, Art. No. 2, 16 pp., doi: 10.3847/0067-0049/223/1/2.

**Souley Madougou** is eScience Engineer at the Netherlands eScience Centre since December 2018. He is mainly involved in the PROCESS project in which he contributes to the implementation of the LOFAR use case and the development and analysis of PROCESS performance models. He previously worked in several eScience projects in the Netherlands. His research interests include performance modelling on many-core architectures, parallel programming and provenance.

**Hanno Spreeuw** is an eScience Research Engineer at the Netherlands eScience Center since February 2015. His Ph.D. research paved the way for the detection of transient radio sources with LOFAR. During his subsequent postdoc position at the Netherlands Cancer Institute he accelerated CPU code for 3D dose reconstruction from radiotherapy treatments in real time. At the Netherlands eScience Center, his projects mostly involve astronomy or physics with a focus on accelerated computing using GPUs.

**Jason Maassen** is Technology Lead at the Netherlands eScience Center. He is involved in many of the projects at the center that apply parallel and distributed programming to scientific applications, ranging from high-resolution climate modeling to digital forensics. In addition, he guides internal software development at the center and scouts for new software technology that can be used in projects. In the past, he participated in many research projects, such as EU FP5 GridLab, the Dutch Virtual Labs for eScience, StarPlane, PROMM-GRID, COMMIT, and H2020 PROCESS, where he worked on a range of topics related to large scale distributed computing.