

FP7- Grant Agreement no. 283393 – RadioNet3

Project name: Advanced Radio Astronomy in Europe

Funding scheme: Combination of CP & CSA

Start date: 01 January 2012

Duration: 48 month



Deliverable D10.14

Final report integrating the Three Benchmark Studies and the Demonstrator Pipelines

Due date of deliverable: 2015-06-30

Actual submission date: 2015-12-11

Deliverable Leading Partner: THE CHANCELLOR, MASTERS AND SCHOLARS OF
THE UNIVERSITY OF OXFORD (UOXF), UK

1 Document information

| | |
|---------|----------------------------------------------------------------------------------------------------------|
| Type | Report |
| Title | Final report integrating the three benchmark studies and the demonstrator pipelines |
| WP | 10 (HILADO) |
| Authors | Ger van Diepen (ASTRON), Harm Munk (ASTRON), Dirk Petry (ESO), Stef Salvini (Oxford University) |

1.1 Dissemination Level

| Dissemination Level | | |
|---------------------|---------------------------------------------------------------------------------------|---|
| PU | Public | X |
| PP | Restricted to other programme participants (including the Commission Services) | |
| RE | Restricted to a group specified by the consortium (including the Commission Services) | |
| CO | Confidential, only for members of the consortium (including the Commission Services) | |

1.2 Content

| | |
|---------------------------------------------------------------|----|
| 1 Document information | 2 |
| 1.1 Dissemination Level | 2 |
| 1.2 Content | 3 |
| 2 Introduction | 4 |
| 2.1 Organisation | 4 |
| 3 Task 1: Optimisation of Casacore | 5 |
| 3.1 Access to native data-formats | 6 |
| 3.2 Measures classes/tables | 6 |
| 3.3 Ephemeris support in the MeasurementSet FIELD Table | 6 |
| 3.4 Table Query Language (TaQL) | 6 |
| 3.5 Support for Distributed File Systems | 7 |
| 3.6 Image format enhancements | 7 |
| 3.7 Software Aspects | 7 |
| 3.7.1 Parallelisation and thread-safety | 7 |
| 3.7.2 Casacore Repositories, Builds and Releases | 7 |
| 3.8 Future Work | 8 |
| 4 Task 2: Fast Transient Imager | 9 |
| 5 Task 3: Large Solvers | 11 |
| 5.1 Algorithm Development | 11 |
| 5.2 Applications Examples | 12 |
| 5.2.1 LOFAR | 12 |
| 5.2.2 MeqTrees | 13 |
| 5.2.3 Casacore | 13 |
| 6 Conclusion | 13 |
| 7 Deliverables | 14 |
| 8 Publications | 15 |
| 9 References | 15 |

2 Introduction

The scientific and technical goal for HILADO is to create optimized prototype software and demonstrator processing pipelines that improve the capabilities of currently planned software packages for existing and emerging radio telescopes. These developments are essential to increase the potential of the RadioNet user community in opening up those facilities for the more demanding scientific applications.

HILADO also explores a new development model for astronomical processing software. Expertise from a variety of disciplines is tapped: from mathematics for the foundations of new algorithms, from computer science for optimisations for high-performance computer platforms, and from industry to explore the space of novel architectures. Talented people around the world are enabled and encouraged to contribute. This collaborative project facilitates the adoption of the new developments, technologies and insights required by huge datasets, and tap expertise beyond the boundaries of the radio astronomy community. The success of this project depends critically on placing existing as well as novel HPC technologies at the centre of the development of algorithms, software and procedures. HILADO aims at optimized prototype software that can be re-used in a variety of contexts, including both automated pipelines and user-adaptable scripts. This brings maximum value to a broad group of RadioNet users by allowing them to balance between highly optimized automated processing and highly flexible interactive applications, without sacrificing efficiency and performance.

This report consists of a summary of the planning, activities and results of the HILADO Tasks.

2.1 Organisation

HILADO is subdivided into four tasks: Optimisation of Casacore, Fast Transient Imager, Large Solvers, and Bringing it to the User. This report covers the first three of these Tasks.

The following HILADO meetings were organised:

- Kick-off meeting on April 26 and 27 2012, in Zaandam, The Netherlands
- Second meeting on April 11 and 12 2013, in Oxford, UK
- Third meeting on January 28, 2014, in Dwingeloo, The Netherlands
- Fourth meeting on March 12 and13 2015, in Garching, Germany.

Monthly telephone conferences were organised to track and coordinate HILADO progress.

3 Task 1: Optimisation of Casacore

Casacore is a C++ library with essential functions for the CASA and other data reduction packages like those for LOFAR. These functions range from storage managers to solvers, from basic data-types to complex astronomical measures. Casacore is thus an essential and necessary component of CASA based data-reduction, and for proprietary packages used for e.g. LOFAR and ASKAP processing.

In the past the library was maintained in an informal consortium, which severely limited further optimisation. HILADO made substantial improvements “under the hood”, leading to significant efficiency improvements and making processing of very large datasets possible.

Originally, HILADO Task 1 consisted of three phases: goal setting (inventory), prototype development, and verification. This was changed early in the project into a yearly cycle of these three phases of goal setting, prototype development, and verification.

In the first cycle, access to native ALMA and LOFAR dataset was implemented (see D10.3). In the first year the Casacore and CASA system have been improved in several ways. An important step has been the creation of the ASDM storage manager to avoid having to copy the bulk data when importing data in ASDM format into the Casacore table format. The storage manager is quite general and can probably be applied to data stored in UVFITS or FITSIDI format as well. The program converting an ASDM dataset into a `MeasurementSet` can be changed such that instead of copying the data it just creates a small look-up table for on-the-fly conversion during data access.

The Measures are an important part of the Casacore package and are expected to be used in SKA software as well. In the HILADO project they have been made thread-safe, and have been optimised to improve performance when used in a multi-threaded program.

For the next cycle, two main areas were identified, relating to datasets and solvers. A third area related to the maintenance and release process.

On 25-26 April 2012 the key developers and users of Casacore (from ASTRON, ESO, NRAO, and ATNF) met in Dwingeloo and discussed how Casacore could be improved in terms of maintenance, releasing, functionality and performance. The outcome of the discussions drove the developments in Casacore, both within HILADO and by other parties.

Topics were:

- Improved documentation and test coverage
- Using STL/Boost classes instead of the homegrown, pre-STL classes like `Map`, `List`, and `Regex`.
- Adapt to C++11 (e.g., using the new move semantics to improve performance)
- Prepare for big data: where applicable use 8-byte instead of 4-byte integers
- Keep adapting to new compiler technology (e.g., `clang`) and keep the code free of warnings.
- The Lattice Expression Language (LEL) is a means to perform expressions on images, for example to add images, find the median, etc. Images will increasingly be stored in a distributed way and it might be worthwhile to make it support distributed processing in a Map-Reduce way of processing. Furthermore, on a single node it can be parallelised.

Early in the third year, a final inventory of improvement areas was made.

3.1 Access to native data-formats

The basic data-format in Casacore and the CASA system is the `MeasurementSet`, which sets out both the data-organisation and meta-data. Some radio telescopes generate some native data-format that can be converted to `MeasurementSet`. This operation, however, is data and compute intensive. Major improvements can be achieved by giving direct access to the native data-formats.

This has been implemented for ALMA through the creation of the ASDM storage manager to avoid having to copy the bulk data when importing data in ASDM format into the CASA Table format. This brings 30 – 60 % disk space savings and even accelerated data access.

The *AdiosStMan* developed by Jason Wang at ICRAR, Australia (<http://www.icrar.org/>), demonstrates the flexibility of the storage manager concept. This is a storage manager using *Adios* (developed at Oak Ridge) making parallel IO possible. An article on *AdiosStMan* has been submitted to *Astronomy and Computing*.

Support for very large images requires that the data are stored in different resolutions to have faster access in e.g. the viewer when showing the entire image. This requires an extra ‘resolution’ argument in functions accessing the data. For large images, the current tiling approach is not sufficiently efficient. The approach will be optimised for large image datasets. In the future, support for JPEG2000 will be added along the lines of the development. (see e.g. [7].)

3.2 Measures classes/tables

The Measures are an important part of the Casacore package and are expected to be used in SKA software as well. They contain definitions and conversions for a wide range of coordinates and units.

Some of these units require data tables (with e.g. observatory coordinates and leap seconds). Currently these have to be maintained locally at each site. A scheme to create two redundant public repositories was worked out which synchronise with each other.

3.3 Ephemeris support in the MeasurementSet FIELD Table

Using the already foreseen but not yet implemented `EPHEMERIS_ID` column in the `MS FIELD` Table, support for the attachment of an arbitrary tabulated ephemeris was added. The ephemerides are stored in the already established table format used in Casacore Measures. The sharing of a single ephemeris by the fields of a mosaic is possible. This is now generally used for handling ephemeris objects (typically solar system objects).

3.4 Table Query Language (TaQL)

TaQL is an SQL like interface to the CASA Tables. As part of the HILADO project, many improvements were added:

- Support for masked arrays
- Selection from more than one table (join on row number)
- Loading of user defined functions as shared libraries
- Support for GROUPBY, HAVING, and aggregation functions
- String formatting options (`printf`-like, angles, times)

3.5 Support for Distributed File Systems

The Casacore tables system was upgraded to support modern distributed file systems such as *Lustre*:

- *MultiFile* was developed to enable Casacore to combine multiple files virtually such that the number of files is reduced (controllable by an external resource file),
- A *NoLocking* option was added to prevent file locking from degrading the file systems performance,
- *TableTrace* was added to enable better debugging and optimisation of table operations.

3.6 Image format enhancements

The Casacore image format was given several new features:

- For spectral line cubes a separate beam can be defined per frequency and Stokes channel,
- Image metadata can now be better organised by assigning each attribute to an attribute group,
- Support for the storage of images in *HDF5* was added,
- Addition of arbitrary attributes to images.

3.7 Software Aspects

3.7.1 Parallelisation and thread-safety

Several basic Casacore algorithms have been parallelised, e.g. sorting and data conversion.

Some code in the *scimath* module was extended with code from packages such as *GSL*.

Parallelisation of higher-level algorithms takes place at the CASA system level.

Thread-safety is an important issue when writing parallel code. Casacore uses the same principle as the C++ Standard Library. The library takes care that global variables (such as statics) are thread-safe, but the user is responsible for using the Casacore objects in a thread-safe way. The performance penalty for making all Casacore code thread-safe would be much too high.

3.7.2 Casacore Repositories, Builds and Releases

In order to boost efficiency in further development, a single code repository was required for Casacore. It was been decided that a *github* repository should be used. The unified Casacore is now available at <https://github.com/casacore/casacore>.

While the Casacore teams at ASTRON and elsewhere already switched earlier, the CASA team fully switched to *github* after the release of CASA 4.5 end of October 2015.

github has its own Issues mechanism for bugs and enhancement request. NRAO uses JIRA. Both mechanisms are used in parallel. Commit messages will contain the issue or JIRA number unless the change is very small.

Some rules and guidelines were established which have to be obeyed when changing Casacore code. As an automated build and test verification system, the *github* repository uses the *Travis* service.

Each module in the Casacore package has been assigned a coordinator who understands the code, coordinates changes, maintains the code, makes sure it builds and tests well, and has sufficient test coverage.

For quite some time Casacore has been built on platforms such as Linux and Mac OS-X. Recently, it has also been built successfully on ARM platforms (e.g. Raspberry Pi) and IBM's Blue Gene supercomputer. By using the C++11 feature of explicit template instantiation, the build time could be reduced with 40% and the library sizes with 20%. Care has been taken to improve the code such that the code builds without warnings. Also the comments have been improved such that *doxygen* generates comprehensive documentation without hardly any warning.

In principle each site can make its own release, but it is expected that the NRAO release schedule will be leading and used by the other partners. In principle the Casacore release version does not need to be the same as the CASA version.

Reference [2] in the special issue on astronomical data formats of the Astronomy & Computing journal, discusses the Casacore Table Data System and several of the improvements made in this Joint Research Activity-HILADO of the RadioNet3 program, including performance improvements.

3.8 Future Work

Some new C++ 11 features have already been used in Casacore, but the code should be changed to use more of such features.

1. Standard STL classes have to be used as much as possible to increase robustness and performance.
 - Replace Map, HashMap, List, Link by their STL-counterparts. OrderedMap and SimpleOrderedMap can possibly be replaced too.
 - Remove cregex and base Regex on std::regex.
 - Replace PtrHolder by unique_ptr (note: auto_ptr is deprecated).
 - Use std::sort in GenSort.
 - Some classes use their own reference counting, which should be replaced by std::shared_ptr. Uncounted references as sometimes used in Tables and Measures, can be replaced by std::weak_ptr.

Some of these C++ STL classes are only available for newer compilers, so where needed (and if possible) the code has to be #ifdef-ed on C++11.

2. Code can be adapted to new C++11 features. However, that should only be done as a background job since its priority is quite low. For example, using the && (move) semantics in e.g. Array operators +, *, etc. could be implemented as it speeds up Array expressions because fewer temporaries are needed.

StEFCal has to be integrated in Casacore. First the unpolarised version can be added. Later, other versions (polarised, multi-directional, etc.) as discussed in section 5, can be added.

The JPEG2000 format is a very promising format to store large images in multiple resolutions. Support for JPEG2000 can be added to Casacore's *Images* module using a library such as *Kakadu* or *OpenJPEG*. In this way it can be shown if JPEG2000 can indeed be used well for multi-dimensional astronomical images.

Currently Casacore code is only checked using a dynamic checking tool such as valgrind. Static checking tools such as Coverity or QAC++ are also very valuable and it should be considered to use them. Furthermore, test coverage should be checked using e.g. gcov, and where needed tests should be extended to increase test coverage.

4 Task 2: Fast Transient Imager

With the advent of ever-larger data-streams, it becomes unavoidable to move data processing functions that have been traditionally carried out interactively into the domain of streaming processing. Through the development of a Fast Transient Imaging Pipeline (FTIP), HILADO both build a specific application (boosting LOFAR Transient Processing) and develop insights and components to ease further development in this area.

The FTIP procedure consist of four parts:

1. Sending the data from the real-time LOFAR control system to the pipeline task processing the data,
2. Run the de-mixing step of DPPP on the data to subtract the strong A-team sources from the data while solving for their gains,
3. Calibrate the data every 10 seconds,
4. Make an image of the data while applying direction-dependent effects like the beam.

A design was made consisting of the following steps to process the data in an efficient way:

1. Sending the data is done using the UDP protocol to prevent the real-time system from being hampered by the FTIP. We expected to use the Pelican framework developed in Oxford for this purpose, but the DPPP framework fulfilled all requirements. Also, a mechanism is developed to send the data from an existing MeasurementSet, which can be used to test the system. It can also be used to find transients in existing observations.
2. De-mixing can take quite some time due to the complexity of the source models and the many parameters to solve for. In the future, this will be addressed using StEFCal, developed in HILADO Task 3 as the solver. Furthermore, the FTIP checks for each time step which A-team sources need to be subtracted, because some may have a too low an apparent flux.
3. The calibration of the data also makes use of StEFCal to make it perform sufficiently well. Furthermore, this step is parallelised.
4. If possible, in the future the awimager program is used to make the images. It is enhanced in two ways:
 - a. Currently, it makes a single image per run. It must be made possible to make an image per second of data, without having to pay the overhead of calculating the convolution functions too often.
 - b. The data of multiple sub bands needs to be combined. This is being developed at the moment, but it has to be tested if it is fast enough.

The current implementation is too slow for the FTIP in line mode because of too much start-up overhead: it is only usable for continuum observations.

A demonstrator FTI pipeline is realised and described in a poster shown at the ADASS XXV conference in Sydney (2015). The paper associated with it will be published in the ADASS XXV Proceedings. (See [3].) The pipeline is based on the LOFAR DPPP framework and consists of the steps described below. The DPPP framework makes it possible to execute multiple steps on a chunk of data, where the data flows from step to step.

1. The data are read from the MeasurementSet being filled by the observation process. The Casacore Table System (HILADO Task 1) makes it possible to read data from a table

while it is being filled. In this way the FTI pipeline is fully independent of the observation process and can as easily be executed on an existing MeasurementSet.

2. An adapted version of the AOFlagger (see [6]) is used to flag RFI.
3. De-mixing performance has been improved, but StEFCal is not used yet because multi-directional StEFCal is not fully tested yet. It is expected to be added before the pipeline is used in the LOFAR production environment.
4. For calibration a GainCal step (using StEFCal) has been made available in DPPP.
5. Multiple imagers have been tested because the start-up overhead of imagers can be quite high. WSClean appeared to be the fastest, but needs a few additions to write all meta data needed for the TraP pipeline detecting the transients.
6. Some Casacore code has been implemented to be able to stream images to the TraP pipeline.

Recently the LOFAR Responsive Telescope project has been started in which the FTI pipeline will be used to detect transients and send out alerts if an important transient has been detected. In this project a latency of a few seconds is no problem, which makes the current pipeline quite feasible.

5 Task 3: Large Solvers

This HILADO Task studied the mathematical foundation and implementation of fast solvers for Radio Astronomy applications.

The calibration of the visibilities obtained by modern radio astronomy interferometers, such as LOFAR, have been a very intensive computational task, increasingly so for future instruments currently being planned such as the SKA (Square Kilometre Array). Any gains in computational efficiency and performance of the calibration computation, would lead to improvements in both scientific throughput and quality, the latter by allowing the study of different scenarios.

Methods were sought that had better performance and scalability properties than existing mainstream algorithms, such as the widely used Levenberg-Marquardt.

In HILADO a new iterative algorithm was developed, implemented and applied, generally known as StEFCal, with far better performance and scalability properties than existing ones and providing a rigorous mathematical assessment of its properties.

The algorithm has been published in refereed journals, and presented and discussed at conferences. (See [1] and [4].)

A StEFCal Workshop was also held on January 29, 2014 at ASTRON, attended by approximately 10 persons.

It has also been and is being applied in a number of radio astronomical environments

- LOFAR, both for the calibration of stations, and within DPPP,
- CASA / Casacore (G. Van Diepen, ASTRON) (under development),
- BBS (G. Van Diepen, ASTRON) (under development),
- AARTAAC (S. Wijnholds et al.),
- MeqTrees + MeerKAT (O. Smirnov et al., see [5]),
- DPPP (T.J.Dijkema, ASTRON).

Because of its nature and performance it is being studied within SKA and it is very likely that it will be used there also.

5.1 Algorithm Development

Current solvers were very time consuming and showed poor scalability with the size of the problem, i.e. with the number of baselines and it was felt in the radio astronomical community that a new generation of faster and more scalable algorithm was needed.

A modification of the Alternating Direction Implicit (ADI) method was therefore studied to provide a computationally efficient way to solve for antenna-based gains both in scalar (i.e. no polarization) and full-polarization modes. The algorithm was nicknamed in jest StEFCal by O.Smirnov and others from the first names of the authors – not by the authors themselves. A later acronym was coined (Statistically Efficient and Fast Calibration).

Salvini and Wijnholds (see [1]) showed that StEFCal was statistically efficient in low-SNR scenarios, which would make this approach particularly suitable for the calibration of large radio astronomical arrays like the Square Kilometre Array (SKA), proved the conditions for its convergence and demonstrated its performance and efficiency in both simulated and real cases.

The new algorithm scales linearly with the number of data points (baselines), i.e. as $\mathcal{O}(P^2)$ where P is the number of receivers (antennas), while currently used method such as Levenberg-

Marquardt scale at best as $\mathcal{O}(P^3)$. Also, because of its pattern of memory access, StEFCal has proved very efficient on modern systems, and it has also been ported experimentally to GPUs.

Salvini and Wijnholds (see [4]) extended the algorithm to the fully polarized case.

Further developments have been and are taking place:

- Multi-time step and/or frequency polynomial StEFCal to reduce the effect of noise. Complex gains are expanded on a set of appropriate basis functions and the minimization then takes place over this set (Salvini, Dijkema, and Wijnholds).
- Use of StEFCal within BDA (Baseline Dependent Averaging). This could possibly be used to average cross-correlator dumps to longer integration times for the shortest baselines (in collaboration with SKA).
- MeqTrees has implemented a form of Direction Dependent calibration, where directions are “peeled” one after the other.

It should also be noted that using StEFCal for SKA would make calibration computationally far cheaper than other tasks within the processing pipelines to such an extent as to render its contribution negligible to overall computational costs. It would be possible therefore to dedicate relatively low computational resources (e.g. CPUs) to calibration while dedicating more expensive and powerful components (e.g. GPUs, other many-cores, etc.) to more computationally intensive tasks such as gridding, FFTs, etc.

5.2 Applications Examples

5.2.1 LOFAR

Currently, LOFAR pipelines use StEFCal, as implemented in DPPP (default pre-processing pipelines). Several options have been added to StEFCal, which have been named after the equivalent option in comparable packages that use different solvers (NRAO CASA, BBS).

| | |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Solint | Find a constant solution for every x time slots, use every time slot as a separate data sample |
| mode | Full jones, diagonal (solve for xx / yy polarization only) or 'commonscalarpahse' (solve for xx / yy, force unit amplitude of the solutions) |
| detectstalling | A history of convergence is analysed as part of StEFCal. If no improvement is achieved for a number of iterations, a 'stall' is detected and the iteration is stopped. The solution of this time slot will later be marked as suboptimal, and possibly the data will be flagged. |

To make solving with StEFCal possible, significant effort had to be put into efficiently simulating visibilities from a sky model. This prediction is multithreaded, and can optionally apply the LOFAR beam model.

Astronomers currently using the DPPP implementation of StEFCal, report that it achieves speedups of at least a factor 10, and the solutions are at least as good as those achieved by BBS.

5.2.2 MeqTrees

StEFCal is now being used extensively in MeqTrees (Noordam and Smirnov, 2010). It should be noticed that both the polarised and unpolarised versions are used in a framework that allows the inclusion of Direction Dependent effects.

5.2.3 Casacore

Work is underway to provide a StEFCal interface to Casacore. That will make StEFCal accessible to those researchers using CASA and Casacore.

6 Conclusion

HILADO Tasks 1-3 results can be summarised as follows:

- The functionality of Casacore was extended through improved interfacing with existing telescopes (ALMA and LOFAR), through improved storage handling of datasets, and through speedups in the implementation. Many table query language extensions and improvements were added. These developments increase the potential of the CASA data reduction package, both for existing telescopes such as ALMA and LOFAR, and for the Square Kilometer Array.
- The development, maintainability and availability of Casacore was improved by the assignment of module coordinators, through the move from Googlecode to a github repository, clean alignment of the code with modern compilers and libraries, improved test coverage, and code clean-ups, ensuring the quality of the Casacore package
- A Fast Transient Imaging Pipeline was implemented and tested, and embedded in the TraP (university of Amsterdam) pipeline for detecting transients with LOFAR
- A solver for large datasets was implemented and tested, achieving a 10-fold speed improvement over existing solvers through the combined mathematical and computer architecture expertise of the HILADO partners.

All tasks resulted in scientific publications.

7 Deliverables

(The following table includes the Task 4 deliverables.)

| Del. No. | Deliverable Title | Lead beneficiary | Nature | Scheduled date | Submitted on |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|-----------|----------------|--------------|
| D10.1 | Detailed activities plan, including top-level architecture of libraries and final selection of benchmark platforms, confirming their availability (including compilers etc.) | ASTRON | Report | 12.2012 | 02.05.2013 |
| D10.2 | Report on the detailed activities plan, including top-level architecture of libraries and final selection of benchmark platforms, confirming their availability (including compilers etc.) | ESO | Report | 06.2013 | 26.06.2013 |
| D10.3 | Report on the comparison of data formats, specifying the key characteristics of optimal formats for various phases in the imaging chain, indicating where and how readily available solutions can be applied | UCAM | Report | 06.2013 | 28.06.2013 |
| D10.4 | Report specifying the requirements and architecture of the Fast Transient Imager, including identification of target hardware platforms | ASTRON | Report | 06.2013 | 26.08.2013 |
| D10.5 | Scientific publication on the application and adaptation of parallel solvers for large astronomical datasets, in particular detailing the application on new hardware platforms | UOXF | Report | 06.2013 | 31.10.2013 |
| D10.6 | Prototype code of improved ParselTongue library, to form a scripting interface to the prototype code | JIVE | Prototype | 01.2014 | 11.04.2014 |
| D10.7 | Optimised prototype software (in the repository), showing the actual performance gain from the optimum | ESO | Prototype | 06.2014 | 02.07.2014 |
| D10.8 | Prototype software for the demonstration of the solvers on a variety of hardware platforms | UOXF | Prototype | 06.2014 | 02.07.2014 |
| D10.9 | Prototype FTI application (code in repository) for use on the target hardware platforms | ASTRON | Prototype | 06.2014 | 02.07.2014 |
| D10.10 | Scientific publication on the results of the demonstrator, and the overall performance gains | ESO | Report | 02.2015 | 01.07.2015 |
| D10.11 | Scientific publication with the results from the FTI demonstrator application on the selected hardware | ASTRON | Report | 02.2015 | 20.11.2015 |
| D10.12 | Scientific publication of the results of demonstrator of large parallel solvers for huge astronomical datasets | UOXF | Report | 02.2015 | 22.09.2015 |
| D10.13 | A framework for minimal re-computation: Demonstrator pipelines for the selected applications | JIVE | Prototype | 02.2015 | 02.06.2015 |
| D10.14 | Final report integrating the three benchmark studies and the demonstrator pipelines | All | Report | 06.2015 | 11.12.2015 |

8 Publications

- [1] Stefano Salvini and Stefan J. Wijnholds, "*Fast gain calibration in radio astronomy using alternating direction implicit methods: Analysis and applications*," *Astronomy & Astrophysics*, V571, no. A97, pp1-14, November 2014.
[DOI: 10.1051/0004-6361/201424487](https://doi.org/10.1051/0004-6361/201424487)
 (Note: this paper was published in *Astronomy & Astrophysics*, and not in IEEE Transactions on DSP as stated in D10.5.)
- [2] G.N.J. van Diepen, "*Casacore Table Data System and its use in the MeasurementSet*," *Astronomy and Computing*, Volume 12, September 2015, Pages 174–180, [DOI: 10.1016/j.ascom.2015.06.002](https://doi.org/10.1016/j.ascom.2015.06.002)
- [3] Ger van Diepen, "*LOFAR Transient Imaging Pipeline*," Proceedings of the Astronomical Data Analysis Software and Systems conference XXV, October 2015, ASP Conference Series Vol. ?, Astronomical Society of the Pacific (To be published)
- [4] Stefano Salvini and Stefan J. Wijnholds, "*StEFCal - An Alternating Direction Implicit Method for Fast Full Polarization Array Calibration*," XXXIth General Assembly and Scientific Symposium, Beijing (China), 16-23 August 2014, [DOI: 10.1199/URSIGASS.2014.6930038](https://doi.org/10.1199/URSIGASS.2014.6930038)

9 References

- [5] J.E.Noordam, O.Smirnov, 2010, "*The MeqTrees software system and its use for third-generation calibration of radio interferometer*," *Astronomy & Astrophysics*, V524, no. A61, December 2010, [DOI: 10.1051/0004-6361/201015013](https://doi.org/10.1051/0004-6361/201015013)
- [6] A.R. Offringa, "[Algorithms for Radio Interference Detection and Removal](#)", June 2012, University of Groningen, dissertation.
- [7] V.V. Kitaeffa, A. Cannon, A. Wiceneca, D. Taubmanb, "Astronomical imagery: Considerations for a contemporary approach with JPEG2000", V12, September 2015, pages 229-239, [DOI: 10.1016/j.ascom.2014.06.002](https://doi.org/10.1016/j.ascom.2014.06.002)

Copyright

© Copyright 2015 RadioNet3

This document has been produced within the scope of the RadioNet3 Projects.

The utilization and release of this document is subject to the conditions of the contract within the 7th Framework Programme, contract no, 283393