Express Production Real-time e-VLBI Service

EXPReS is funded by the European Commission (DG-INFSO),
Sixth Framework Programme, Contract #026642

Report on integration of E-VLBI System

Title:	Report on integration of E-VLBI System
Sub-title:	
Date:	2009/06/10
Version:	1.0
Filename:	report
Author:	D. Stokłosa
Co-Authors	
Summary:	

Table of contents

Table of figures

1. Status before

The following figure presents the status of the e-VLBI System before integration at Jive.

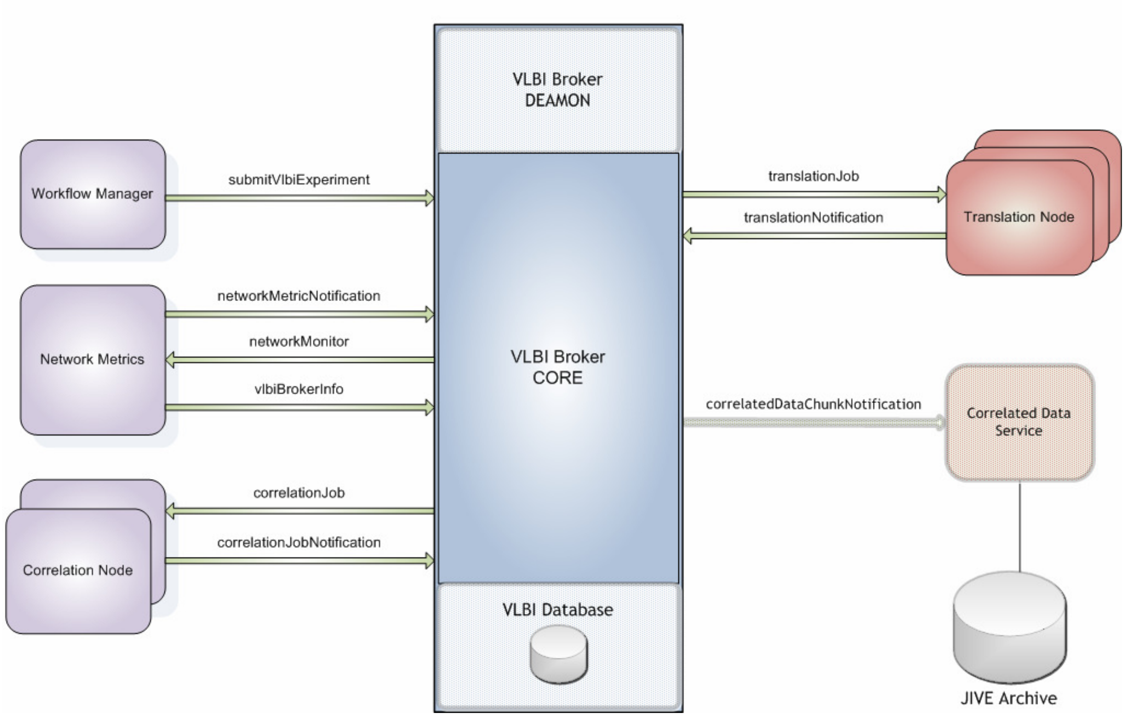


Figure 1 E-VLBI System - status before integration

- *VLBI Broke*r module – single threaded module core implemented. The broker core and persistence layer is missing. The module is not integrated with other system components. *VLBI Broker Demon* is responsible for triggering all events depending on the system state i.e. starting new experiments, updating statuses or sending notifications. *VLBI Database* – internal storage for VLBI experiments
- *Correlated Data Service* (CDS) - service responsible for storing correlated data in the Jive Archive. The CDS service is not implemented.
- *Translation Node* (TN) – responsible for handling data from radio telescopes and preparing data for correlation. There can be many TNs involved in VLBI experiment. However it is required that each radio telescope has one Translation Node assigned. The TN module is informed about new experiment by VLBI Broker. The TN core has been implemented but not integrated with VLBI Broker.
- *Correlation Node* (CN) – responsible for managing and executing correlation jobs on the cluster/grid. The module core has been implemented. The integration with VLBI Broker is required.

2. E-VLBI system testbed

The e-VLBI System working environment has been prepared where all system components have been deployed. The actual system testbed description can be found at: http://www.jive.nl/dokuwiki/doku.php/expres:fabric:evlbisystem:internal:testbed.[1]

1. VLBI Broker – deployed at Jive (huygens.jive.nl)
2. Translation Node – deployed at Jive (huygens.jive.nl)
3. Correlation Node –three correlation nodes have been deployed:
   - expres.reef.man.poznan.pl (PSNC)
   - clusia.man.poznan.pl (PSNC)
   - adam.astron.nl (Jive)

3. e –VLBI system components

The first objective was to integrate all system components with VLBI Broker. First of all, old communication schema has been analysed. Since VLBI Broker is a central module, which communicates with other components it was decided that common communication interface for all modules should be introduced.
The following figure presents interactions between system modules.

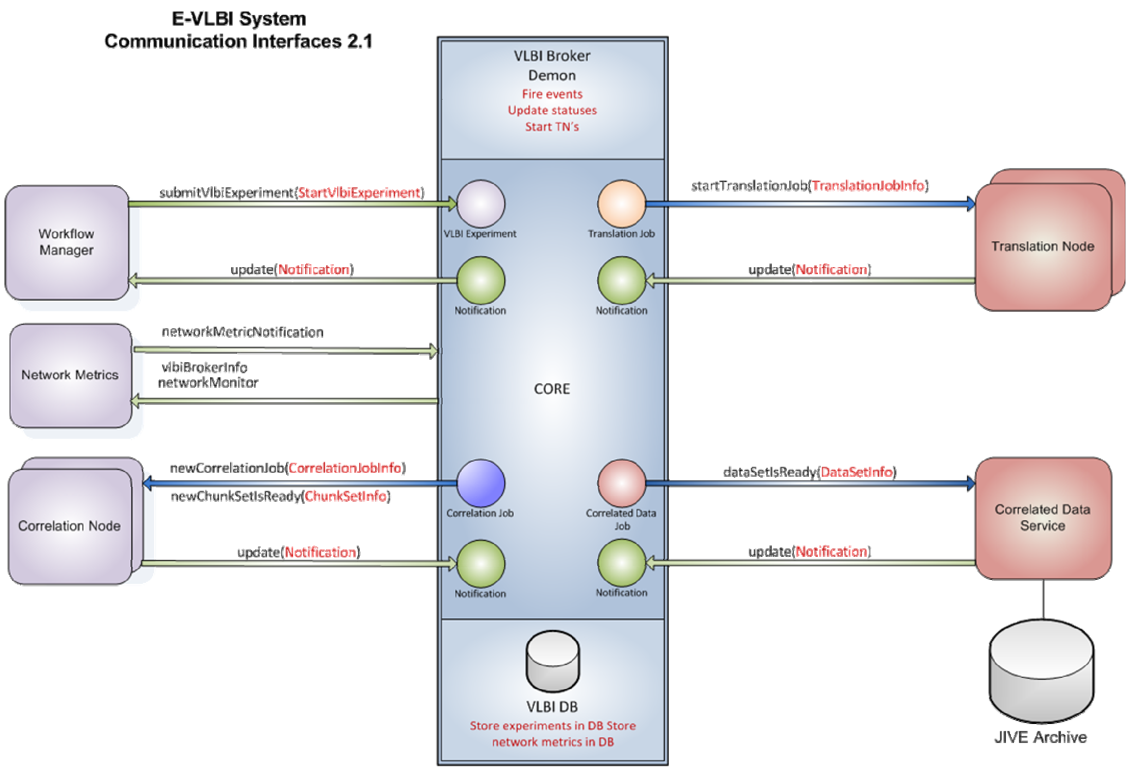


Figure 2 E-VLBI system components

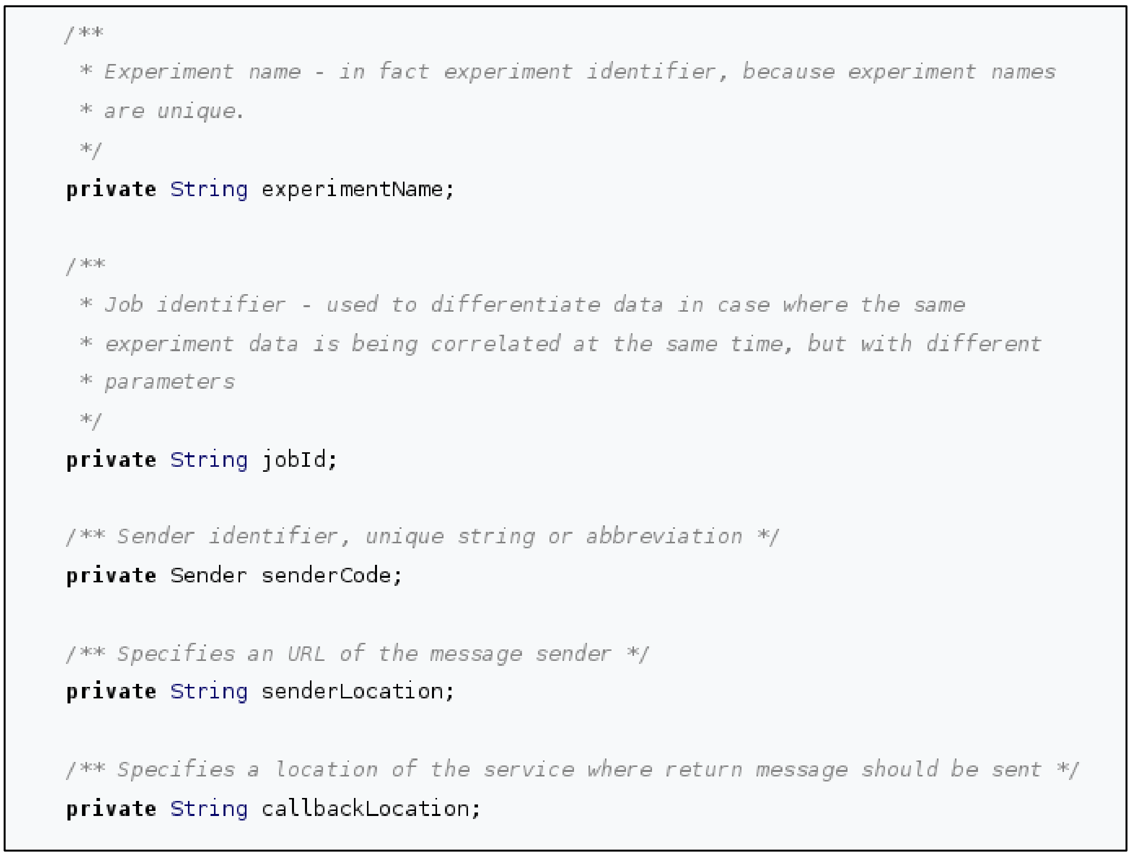
[1] Wiki account is required to view this page

Changes in the common communication interface:

- unused fields removed
- added missing items
- the communication schema has been unified. Each message consists of MessageHeader and ChunkInfo structures (see 3.1 and 3.2)
- common Notification interface for exchanging information introduced

3.1. MessageHeader structure

Each message send between system components contains a structure called *Message Header* which stores general information about message sender.



```
/**
 * Experiment name - in fact experiment identifier, because experiment names
 * are unique.
 */
private String experimentName;

/**
 * Job identifier - used to differentiate data in case where the same
 * experiment data is being correlated at the same time, but with different
 * parameters
 */
private String jobId;

/** Sender identifier, unique string or abbreviation */
private Sender senderCode;

/** Specifies an URL of the message sender */
private String senderLocation;

/** Specifies a location of the service where return message should be sent */
private String callbackLocation;
```

Figure 3 Message Header

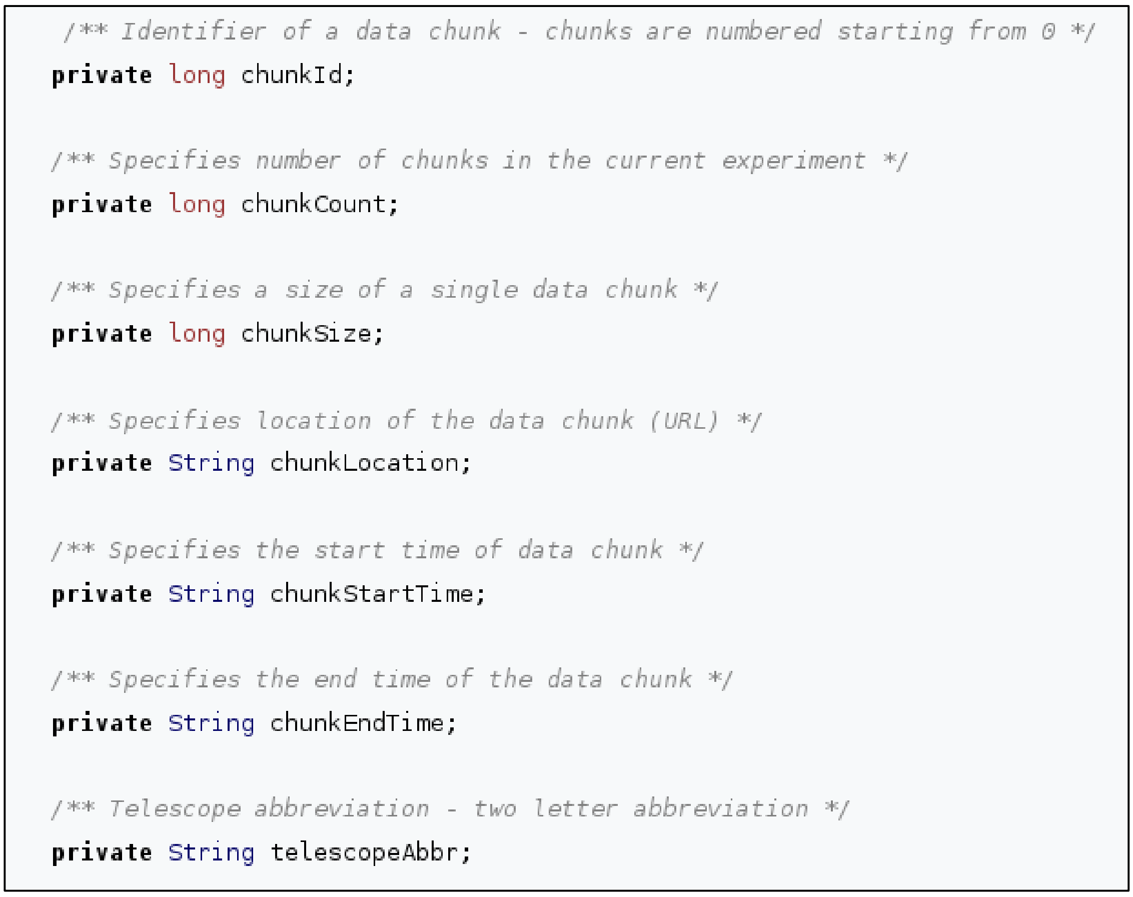
We have introduced a *jobId* which is used to distinguish the same experiments correlated with different parameters. The *callbackLocation* property is used by the message receiver for sending a reply message. The *senderCode* determines the message sender. The list of valid message senders is presented in the figure below.

```
public enum Sender {

        /** Sender code for Vlbi Broker module */
        VLBI_BROKER("broker"),

        /** Sender code for Translation Node module */
        TRANSLATION_NODE("tn"),

        /** Sender code for Correlatin Node module */
        CORRELATION_NODE("cn"),

        /** Sender code for Correlated Data Serrvice module */
        CORRELATION_DATA_SERVICE("cds");
```

Figure 4 List of valid message senders

3.2. ChunkInfo structure

Each message send between system components has a structure called *Chunk Info* which contains general information about a data chunk from a given radio telescope. The chunk info structure is also used to inform VLBI Broker and Correlation Data Service about a location of a correlated data. The detailed description of the Chunk Info structure is presented in the figure below.

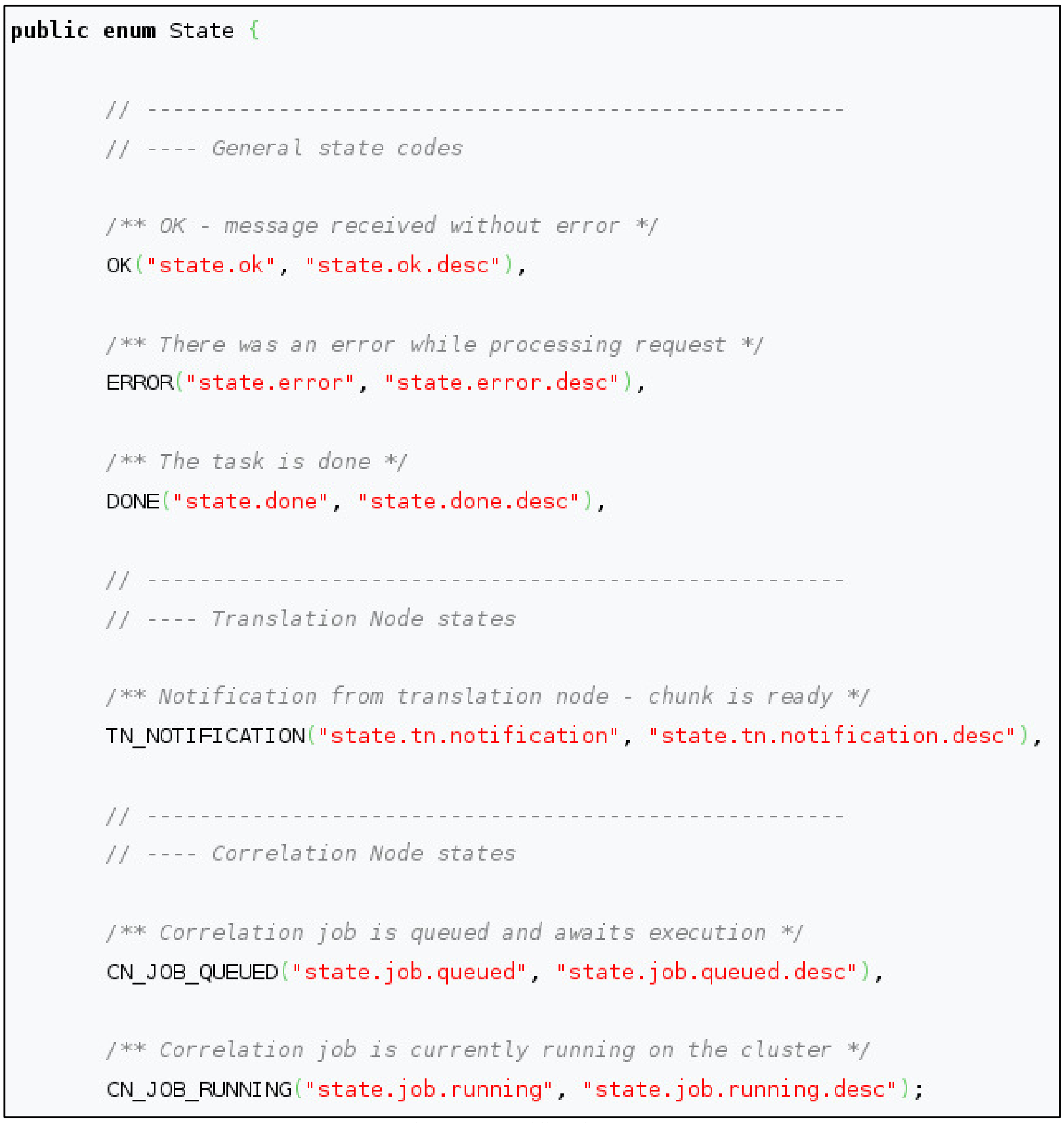


```
 /** Identifier of a data chunk - chunks are numbered starting from 0 */
private long chunkId;

/** Specifies number of chunks in the current experiment */
private long chunkCount;

/** Specifies a size of a single data chunk */
private long chunkSize;

/** Specifies location of the data chunk (URL) */
private String chunkLocation;

/** Specifies the start time of data chunk */
private String chunkStartTime;

/** Specifies the end time of the data chunk */
private String chunkEndTime;

/** Telescope abbreviation - two letter abbreviation */
private String telescopeAbbr;
```

Figure 5 Chunk Info structure

3.3. Notification service

The *Notification* service is used by system modules to notify other components about changes in the correlation state or errors. Sample notifications: new data chunk is ready, a chunk set has been correlated. The notification message consists of message header, chunk info structure (if this is relevant), state and message. The type of the notification event is encoded in the *state* field. The list of possible states is presented in the figure below.



```
public enum State {

        // ----------------------------------------------------
        // ---- General state codes

        /** OK - message received without error */
        OK("state.ok", "state.ok.desc"),

        /** There was an error while processing request */
        ERROR("state.error", "state.error.desc"),

        /** The task is done */
        DONE("state.done", "state.done.desc"),

        // ----------------------------------------------------
        // ---- Translation Node states

        /** Notification from translation node - chunk is ready */
        TN_NOTIFICATION("state.tn.notification", "state.tn.notification.desc"),

        // ----------------------------------------------------
        // ---- Correlation Node states

        /** Correlation job is queued and awaits execution */
        CN_JOB_QUEUED("state.job.queued", "state.job.queued.desc"),

        /** Correlation job is currently running on the cluster */
        CN_JOB_RUNNING("state.job.running", "state.job.running.desc");
```

Figure 6 Notification type

3.4. Interactions between system components

This chapter presents interactions between VLBI Broker and other system components. Each figure presents a collaboration between VLBI broker and other component. The entire interaction between system components can be divided into the following phases:

- Chunking data
- Correlating chunked data
- Storing correlated data

a. Chunking data

Before data can be sent for correlation it has to be divided into a smaller chunks. Translation Node is a module responsible for handling data from radio telescopes and preparing it for correlation. The process can be described as follows (see Figure 7):

- Translation Node (TN) is informed by VLBI Broker about new experiment. TN prepares environment for chunking data.
- Confirmation message or error message is sent to the VLBI Broker
- TN starts chunking process, stores chunks and finally notifies broker about theirs locations
- When last chunk is transmitted TN informs broker that all chunks have been transmitted

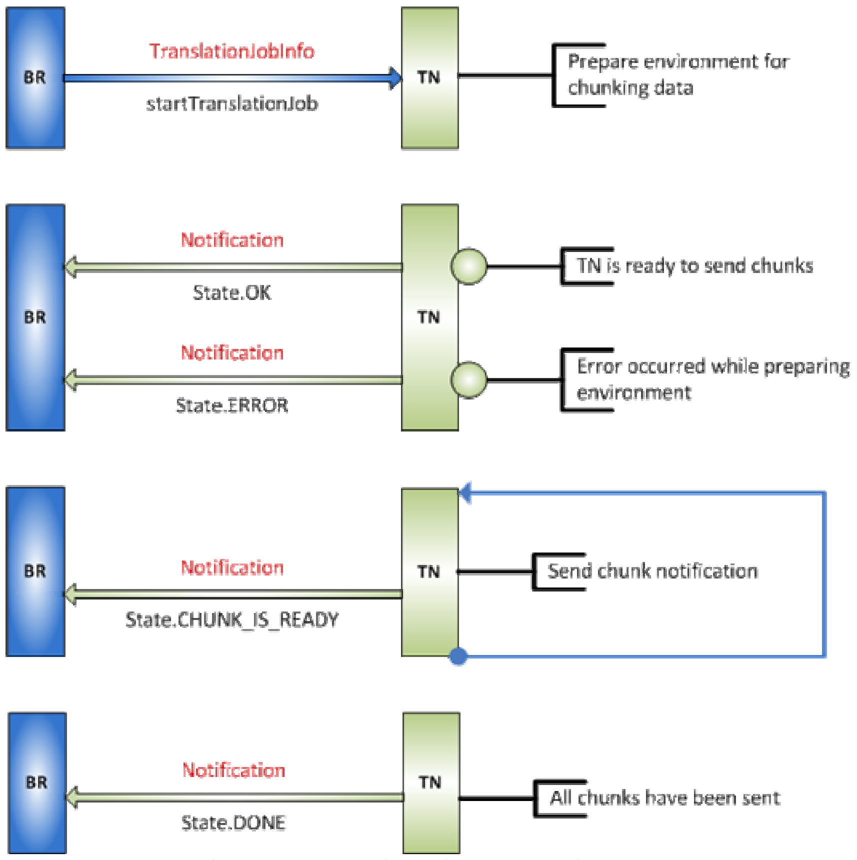


Figure 7 Interaction with Translation Node

b. Correlating chunked data

When first data chunks have been delivered the correlation process can be started. Correlation Node is a module responsible for generating a correlation scripts and submitting correlation jobs. The process can be described as follows (see Figure 8):

- Correlation Node (CN) is informed by VLBI Broker about new experiment. CN prepares environment for correlation
- Confirmation message or error message is sent to the VLBI Broker
- CN starts submitting correlation jobs in the Grid
- When data set is correlated the notification is sent back to the VLBI Broker

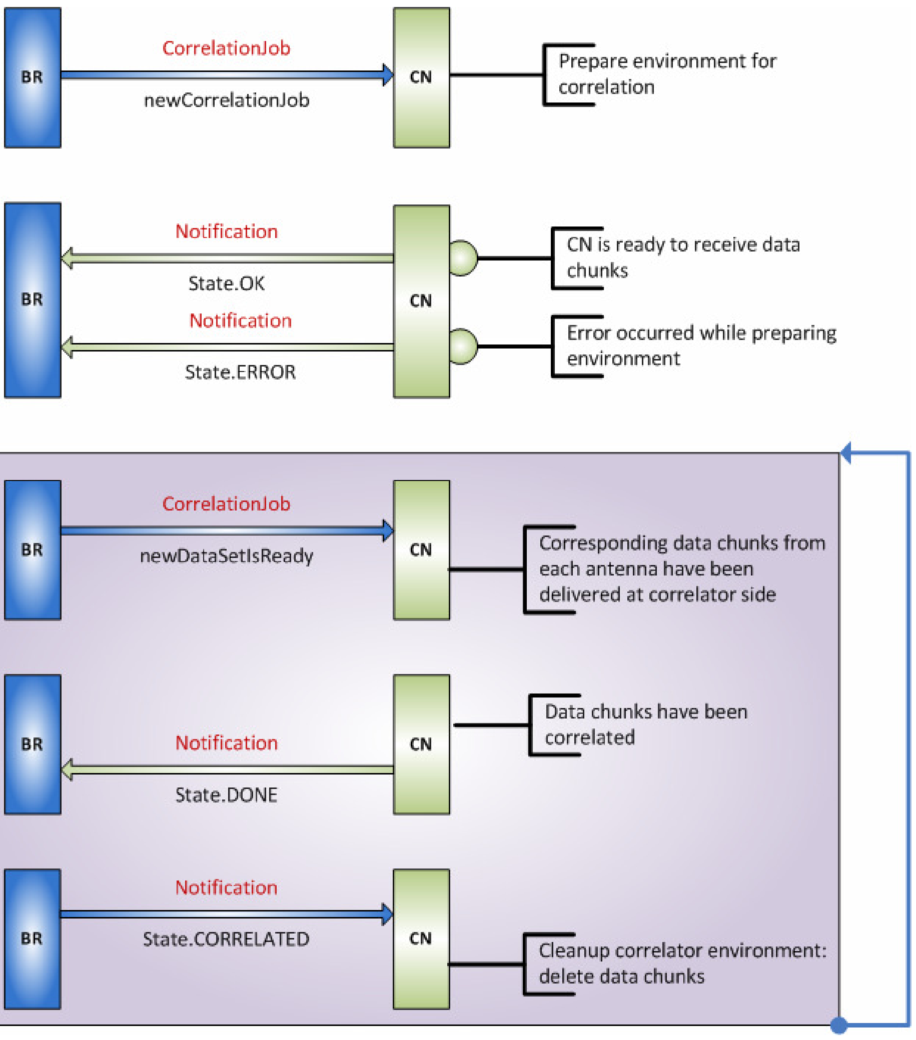

BR
CorrelationJob
newCorrelationJob
CN
Prepare environment for correlation
BR
Notification
State.OK
Notification
State.ERROR
CN
CN is ready to receive data chunks
Error occurred while preparing environment
BR
CorrelationJob
newDataSetIsReady
CN
Corresponding data chunks from each antenna have been delivered at correlator side
BR
Notification
State.DONE
CN
Data chunks have been correlated
BR
Notification
State.CORRELATED
CN
Cleanup correlator environment: delete data chunks

Figure 8 Interaction with Correlation Node

c. Storing correlated data

When first data chunk have been correlated they have to be stored in the VLBI Database. The Correlated Data Service is responsible for downloading and storing correlated data products. The process can be described as follows (see Figure 9):

- VLBI Broker informs Correlated Data Service (CDS) that new correlated data product is ready for download
- CDS downloads the correlated data set and sends confirmation message or error message
- The correlated data product, as well as corresponding data chunks are removed form Grid environment

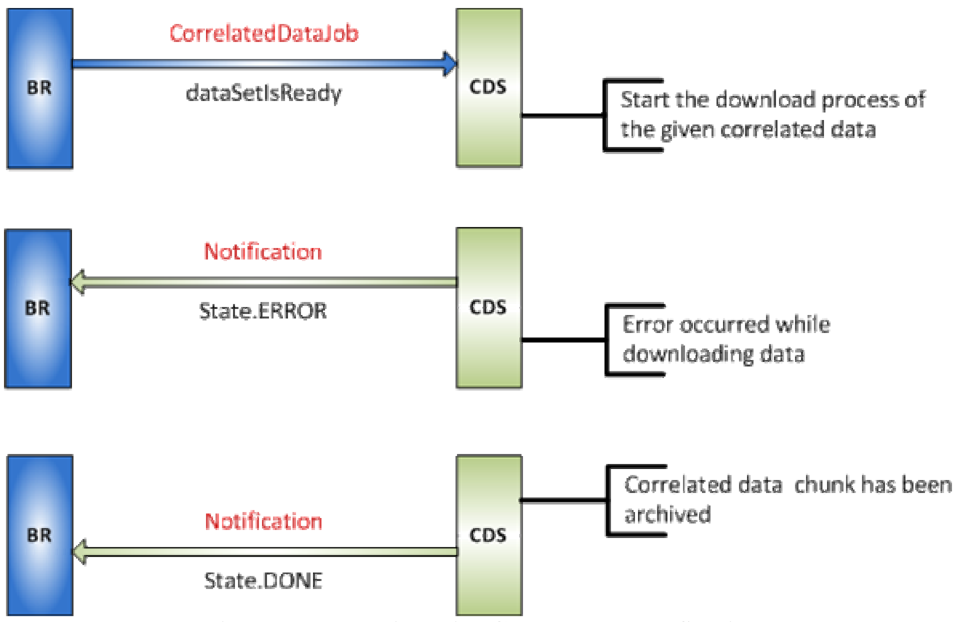

BR
CorrelatedDataJob
dataSetIsReady
CDS
Start the download process of the given correlated data
BR
Notification
State.ERROR
CDS
Error occurred while downloading data
BR
Notification
State.DONE
CDS
Correlated data chunk has been archived

Figure 9 Interaction with Correlated Data Service

4. Problems

During work on the e-VLBI system we have faced several problems.

- Problem with delay table generation: if the delay table is not present, the SFXC exits after generating it. It does not correlate the first chunk of data. It was decided that a script will be created to generate delay table. The Correlation Node Service will be responsible for calling this script before actually starting to submit correlation jobs. After the delay table is created and stored in the experiment directory (which is experiment.name_job.id) the Correlation Node can start submitting jobs.
- We have also faced several different problems with the software correlator deployed on reef cluster at PSNC. The real source of all these problems was difficult to find because they were only present on the reef cluster. The list of issues is presented below:
    - The SFXC ends up in an endless loop when number of nodes specified in the mpi script is greater than SFXC can possibly use.
    - Sometimes the SFXC does not release all resources and does not clean the environment properly
    - We have also experienced not satisfactory speed of the software correlator. This is currently being analysed. One possible factor could be the NFS system of the reef cluster.
- By the time e-VLBI system has been tested we had experienced some problems with the NFS system on the reef cluster in Poznan.

5. E-VLBI system – test results

All system components have been deployed within the system testbed. We have managed to conduct several experiments using distributed software correlation:

1. self correlation
2. distributed correlation with one software correlator and four radio telescopes: Torun, Medicina, Westerbork and Cambridge.
3. distributed correlation with two software correlators and four radio telescopes: Torun, Medicina, Westerbork and Cambridge.

5.1. Sample experiment description

This section describes briefly the process of conducting sample VLBI experiment. Firstly, the experiment workflow has to be constructed based on the specified vex file using Workflow Manager Application (see Figure 10)

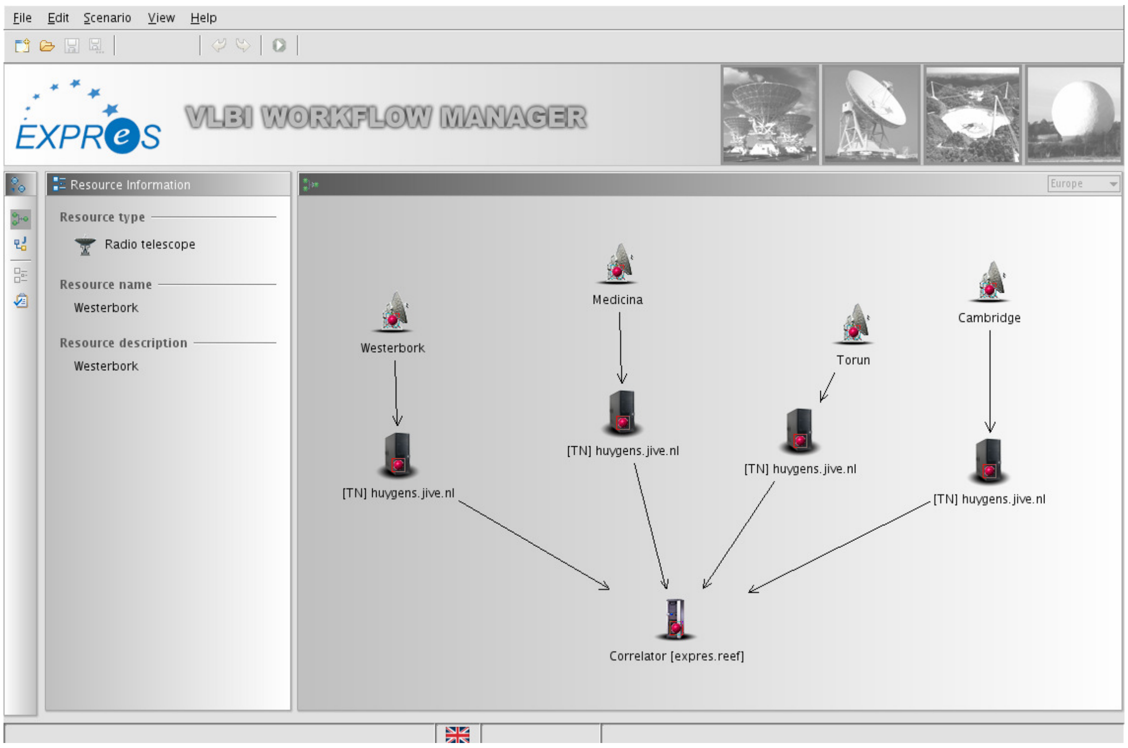
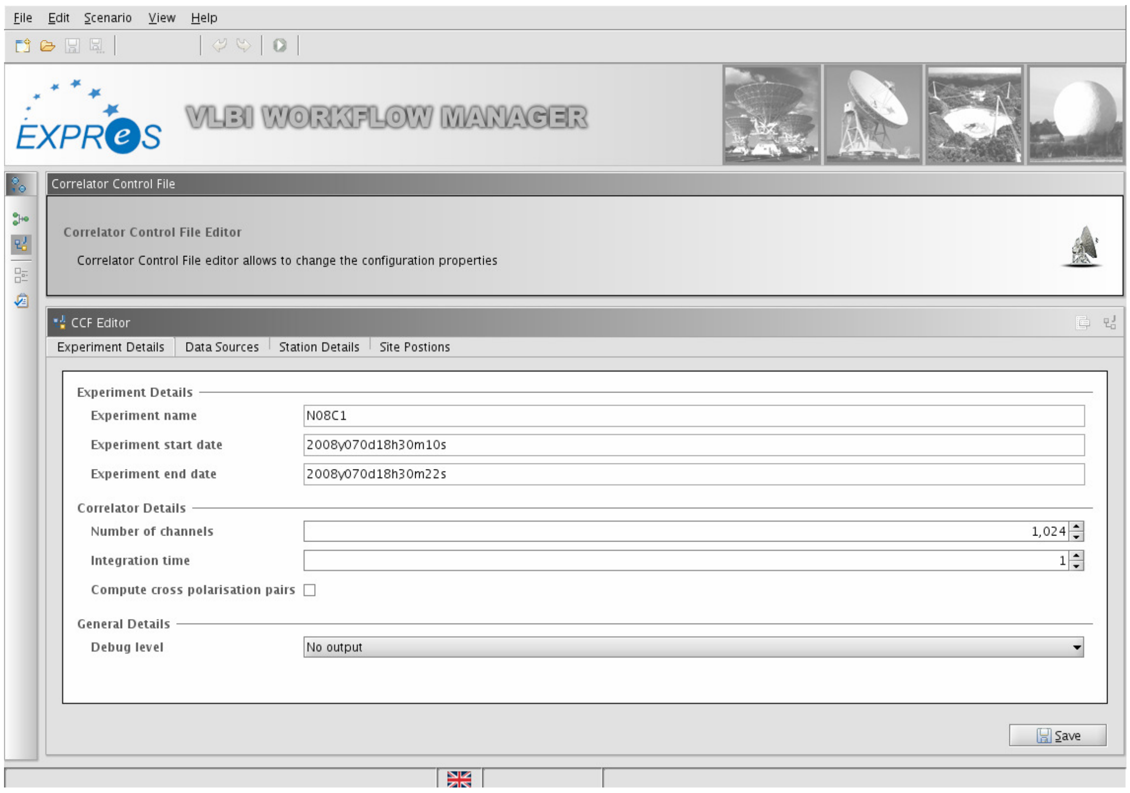


Figure 10 Sample VLBI experiment

The experiment parameters has to be set up using CCF editor:



Correlator Control File Editor

Correlator Control File editor allows to change the configuration properties

Experiment Details
Experiment name: N08C1
Experiment start date: 2008y070d18h30m10s
Experiment end date: 2008y070d18h30m22s

Correlator Details
Number of channels: 1,024
Integration time: 1
Compute cross polarisation pairs: ☐

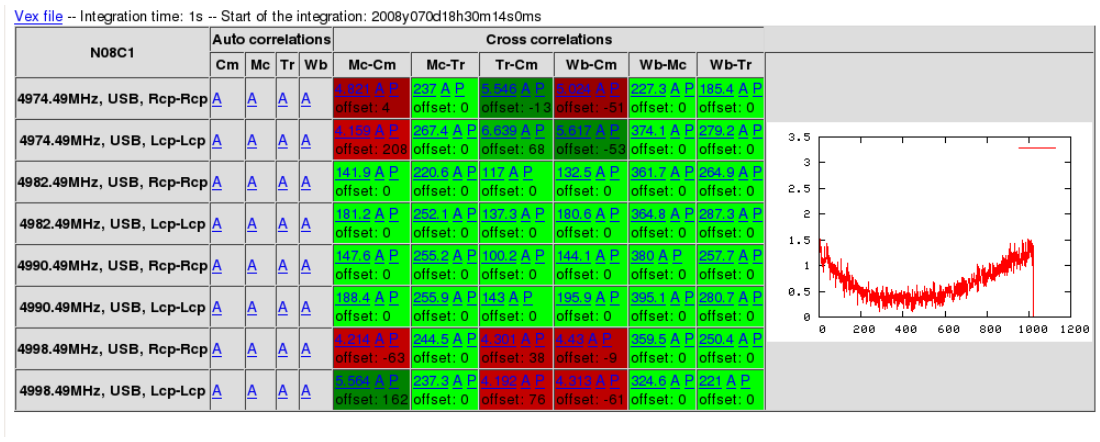
General Details
Debug level: No output

Figure 11 CCF editor

The figure below contains scan details. There are four antennas: Westerbork, Medicina, Torun and Cambrigde.

```
scan No0033;
      start=2008y070d18h30m00s; mode=NME.6CM; source=4C39.25;
      station=Wb:    0 sec:    180 sec:   470.189872312 GB :    :   : 1 ;
      station=Mc:    0 sec:    180 sec:   469.856467208 GB :    :   : 1 ;
      station=Tr:    3 sec:    180 sec:   467.909433728 GB :    :   : 1 ;
      station=Cm:    0 sec:    180 sec:   287.683837032 GB :    :   : 1 ;
```
Figure 12 Scan details

Finally, when distributed correlation has been finished we can see the fringes - integration start: 2008y070d18h30m14s0ms.


Vex file -- Integration time: 1s -- Start of the integration: 2008y070d18h30m14s0ms

N08C1	Auto correlations				Cross correlations					
	Cm	Mc	Tr	Wb	Mc-Cm	Mc-Tr	Tr-Cm	Wb-Cm	Wb-Mc	Wb-Tr
4974.49MHz, USB, Rcp-Rcp	A	A	A	A	4.821 A P offset: 4	237 A P offset: 0	5.546 A P offset: -13	5.024 A P offset: -51	227.3 A P offset: 0	185.4 A P offset: 0
4974.49MHz, USB, Lcp-Lcp	A	A	A	A	4.159 A P offset: 208	267.4 A P offset: 0	6.639 A P offset: 68	5.617 A P offset: -53	374.1 A P offset: 0	279.2 A P offset: 0
4982.49MHz, USB, Rcp-Rcp	A	A	A	A	141.9 A P offset: 0	220.6 A P offset: 0	117 A P offset: 0	132.5 A P offset: 0	361.7 A P offset: 0	264.9 A P offset: 0
4982.49MHz, USB, Lcp-Lcp	A	A	A	A	181.2 A P offset: 0	252.1 A P offset: 0	137.3 A P offset: 0	180.6 A P offset: 0	364.8 A P offset: 0	287.3 A P offset: 0
4990.49MHz, USB, Rcp-Rcp	A	A	A	A	147.6 A P offset: 0	255.2 A P offset: 0	100.2 A P offset: 0	144.1 A P offset: 0	380 A P offset: 0	257.7 A P offset: 0
4990.49MHz, USB, Lcp-Lcp	A	A	A	A	188.4 A P offset: 0	255.9 A P offset: 0	143 A P offset: 0	195.9 A P offset: 0	395.1 A P offset: 0	280.7 A P offset: 0
4998.49MHz, USB, Rcp-Rcp	A	A	A	A	4.214 A P offset: -63	244.5 A P offset: 0	4.301 A P offset: 38	4.43 A P offset: -9	359.5 A P offset: 0	250.4 A P offset: 0
4998.49MHz, USB, Lcp-Lcp	A	A	A	A	5.564 A P offset: 162	237.3 A P offset: 0	4.192 A P offset: 76	4.313 A P offset: -61	324.6 A P offset: 0	221 A P offset: 0

Figure 13 Sample results

6. Summary

The main objective of the Fabric activity, which PSNC and JIVE were involved in, was to design and implement a distributed version of a software correlation. This objective has been fulfilled. We have conducted an experiment with a distributed software correlation modules running on two separate clusters at PSNC (Poznan, Poland). Data has been served by translation node deployed at Jive (Dwingeloo, Netherlands).

The prototype system which has been constructed proved that distributed, software correlation with Grid resources is possible.

Currently the work is focused on integration of network modules with e-VLBI System. The network modules will be responsible for network measurements and providing tips for VLBI operator during design phase of an observation workflow.

The e-VLBI System will be tested with a sample data set: four stations in two different modes: with two and three software correlation modules. The test results will be included in the final report of the EXPReS project.