Express Production Real-time e-VLBI Service

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

E-VLBI System final report

Poznan Supercomputing and Networking Center

Title:	E-VLBI System – final report
Sub-title:	
Date:	2009/09/10
Version:	1.1
Filename:	e-vlbi_system_final_report_v1.1.doc
Author:	D. Stokłosa
Co-Authors	N.Meyer, M. Pabis, L. Dolata, M. Lawenda

Table of contents

Table of figures

1 Introduction

This report summarizes the work conducted by Poznan Supercomputing and Networking Center (PSNC) together with JIVE within FABRIC activity in the following tasks:

- WP 2.1.1. Grid – VLBI collaboration
- WP 2.1.2. Grid Workflow management
- WP 2.1.3 Grid routing

The main objective was to design and implement a software-based distributed correlation embedded in a Grid Computing environment.

We have prepared two releases of a distributed e-VLBI System. First, a working prototype with the limited functionality has been designed, implemented and deployed.

The second prototype of the system allowed us to achieve all the objectives. We have successfully conducted a series of VLBI experiments. The entire process has been controlled by the e-VLBI System. The following paragraphs describe the e-VLBI System in more details, as well as system tests and results.

2 The e-VLBI system

2.1 The e –VLBI system components

The e-VLBI System has been constructed to provide tools for managing the distributed software correlation. The system consists of the following components: a Workflow Manager

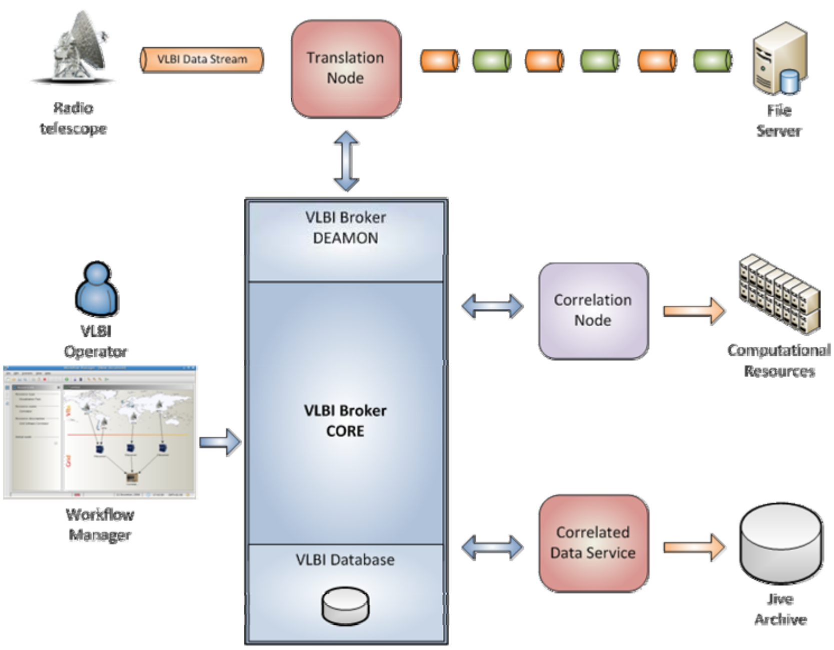


Figure 1 The general architecture of e-VLBI System

(WFM), Translation Node (TN), VLBI Broker (VB), Correlation Node (CN) and Correlated Data Service (CDS).

a) ***The Workflow Manager** (WFM)* **–** the e-VLBI System allows astronomers to plan, execute and monitor their observations in the form of so called workflows. However, the e-VLBI experiment is not only the observation itself. The e-VLBI experiment consists of a definition of storage elements, a definition of data flows or a definition of computation resources, etc. Such an e-VLBI workflow has to be created for each observation. The Workflow Manager Application (WFM) has been created to allow users to design and execute their observation workflows easily.

b) ***The VLBI Broker** (VB)* **–** central element of the e-VLBI System providing the centralized control of the entire experiment. The VB module processes experiment definition submitted from the WFM application and forwards tasks' description to telescope sites (Translation Nodes) and correlation sites (Correlation Nodes). This module is also responsible for coordination and submission of computational jobs with the distributed correlation of chunked data.

c) **Translation Node** (TN) – responsible for handling data from radio telescopes and preparing data for correlation. The data stream from radio telescope is buffered and divided into a smaller chunks. Chunk sets (every corresponding chunk from each data source) are temporarily stored within *File Server* disk space until they are correlated. 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 each new experiment by VLBI Broker.

d) **Correlation Node** (CN) – responsible for managing and executing correlation jobs on the cluster/grid.

e) **Correlated Data Service** (CDS) – service responsible for storing correlated data in the Jive Archive.

2.2 The communication protocol

The following figure presents the interaction between all system components. As you can see the VLBI Broke module is the central module. The rest of the components communicate with VB with a constant update of their current state.

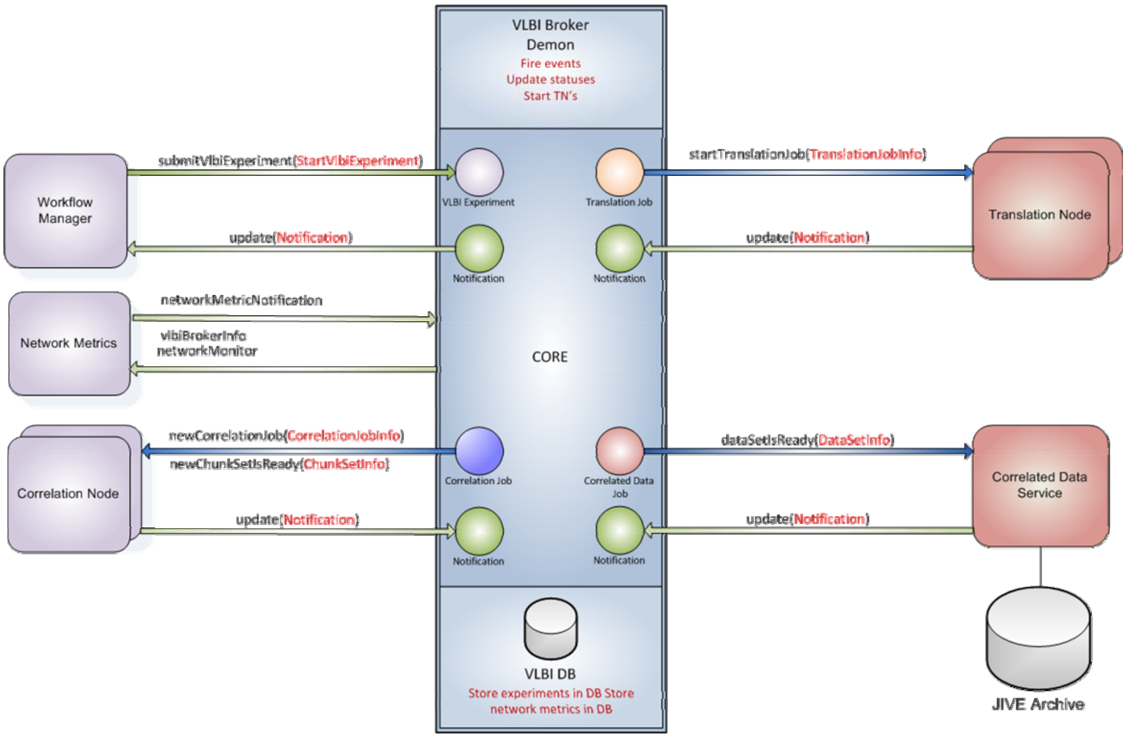


Figure 2 E-VLBI system components

2.2.1 MessageHeader structure

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

```java
/**
 * 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.

```java
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

2.2.2 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

2.2.3 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"),

        /** The experiment done */
        EXPERIMENT_DONE("state.exp.done", "state.exp.done.desc"),

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

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

Figure 6 Notification type

2.3 Network monitoring

After analysis of available network monitoring systems we have chosen perfSONAR to be used within the e-VLBI System. The perfSONAR monitoring services were deployed across GÉANT2 network and in selected NRENS (in pilot phase). An analysis showed that the best solution for measurement and other network management activities should be a dedicated server installations . The Network Monitor Module has been installed at JIVE, at PSNC and finally in Torun.

The Expres Network Monitor Module is responsible for providing optimal connection between the given Expres resources that should assure the fastest possible and reliable data transfer within e-VLBI GRID environment. The module performs on-demand tests, analyses network performance along transporting path, and provides the information about end point pairs for e-VLBI workflow management system. The overall Expres Network Monitor architecture is shown in Figure 7.

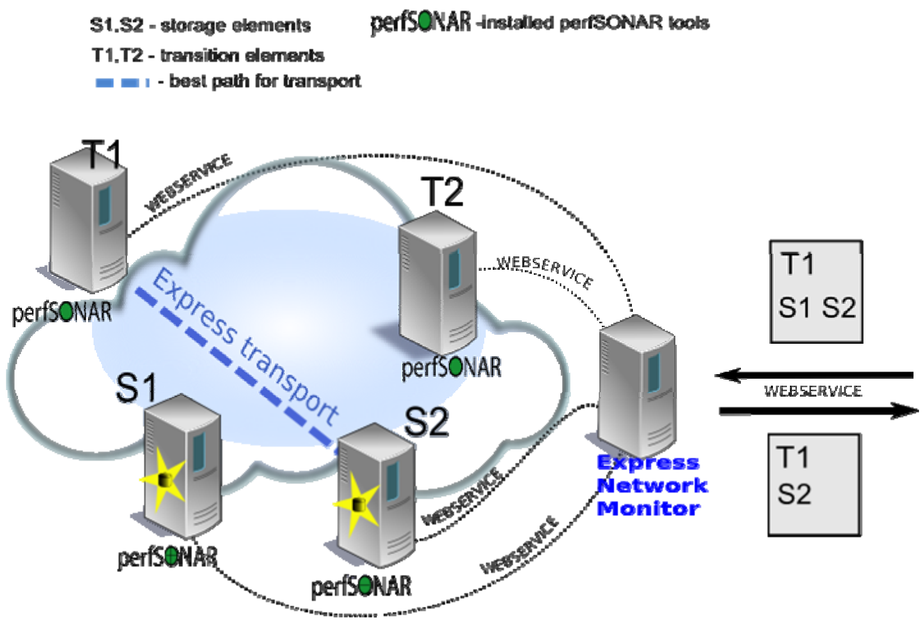


Figure 7 The Express Network Monitor architecture

The Expres Network Monitor application works as a request point for the e-VLBI workflow management system. It controls all configured measurements nodes, requests all needed on-demand tests, analyses the results and provide an ordered list of resource pairs.

A communication is based on web services technology and JSON protocol. The Expres Network Monitor waits for requests from e-VLBI workflow management system. Each time a transport session is scheduled, system creates a list of translation nodes and storage elements related with this session. The unordered list is then sent to the Expres Network Monitor together with value of the requested bandwidth for transport. The Expres Network Monitor creates measurement session and starts to engage the on-demand tests. At first stage system performs a number of parallel round trip times (RTT) measurements between all translation nodes and each storage element. A list of results is sorted in ascending order. The system performs single bandwidth utilization test to storage nodes in order resulting from the earlier list of measured RTTs. If measured achievable bandwidth is higher than requested bandwidth the transition node and storage element pair is created. The e-VLBI workflow management system checks simultaneously the measurement session progress and when it is completed updates information which translation node should be paired with which storage element.

2.4 The interaction 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

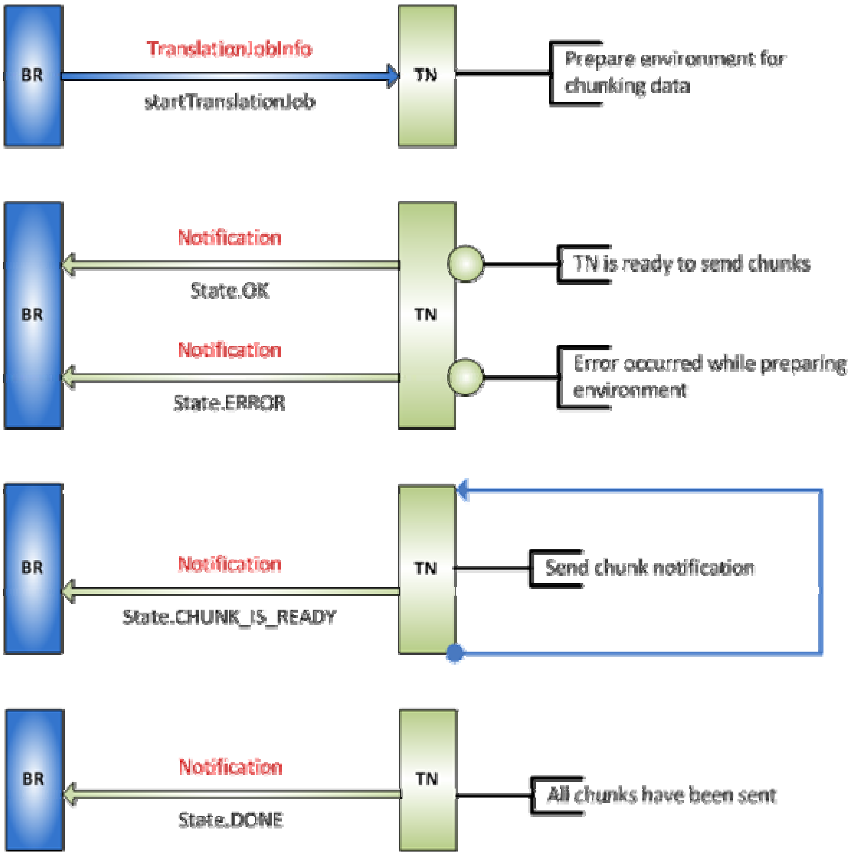


Figure 8 Interaction with Translation Node

preparing it for correlation. The process can be described as follows (see Figure 8):

- 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

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.

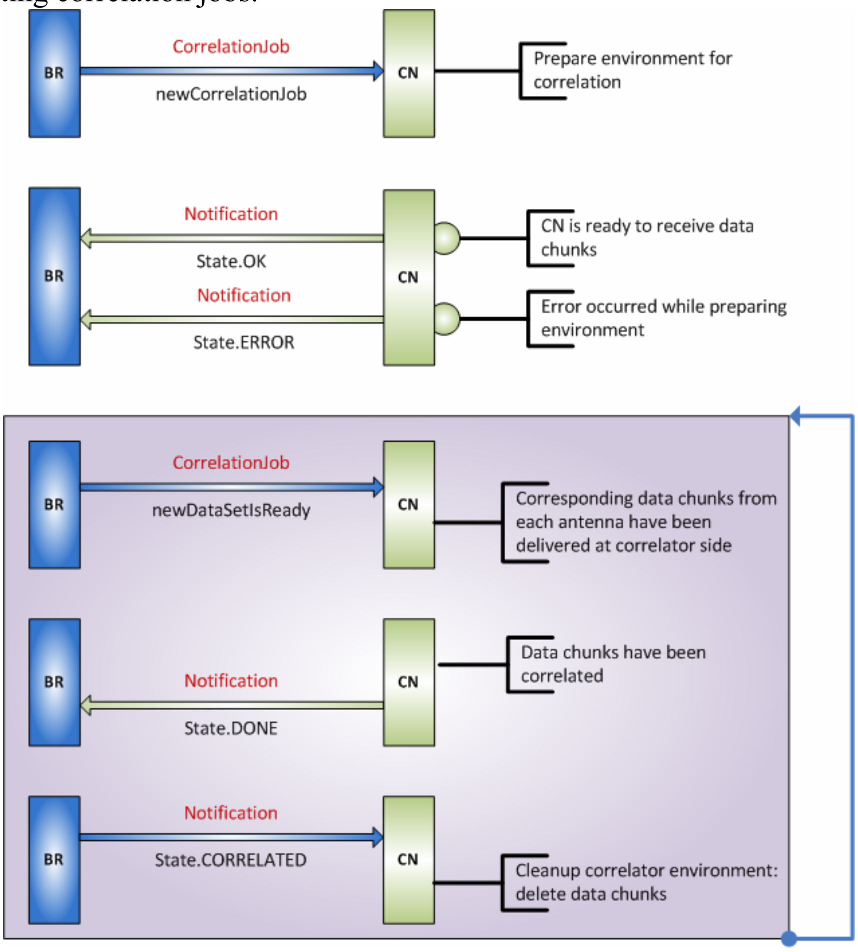


Figure 9 Interaction with Correlation Node

The process can be described as follows (see Figure 9):

- 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

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 10):

- 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

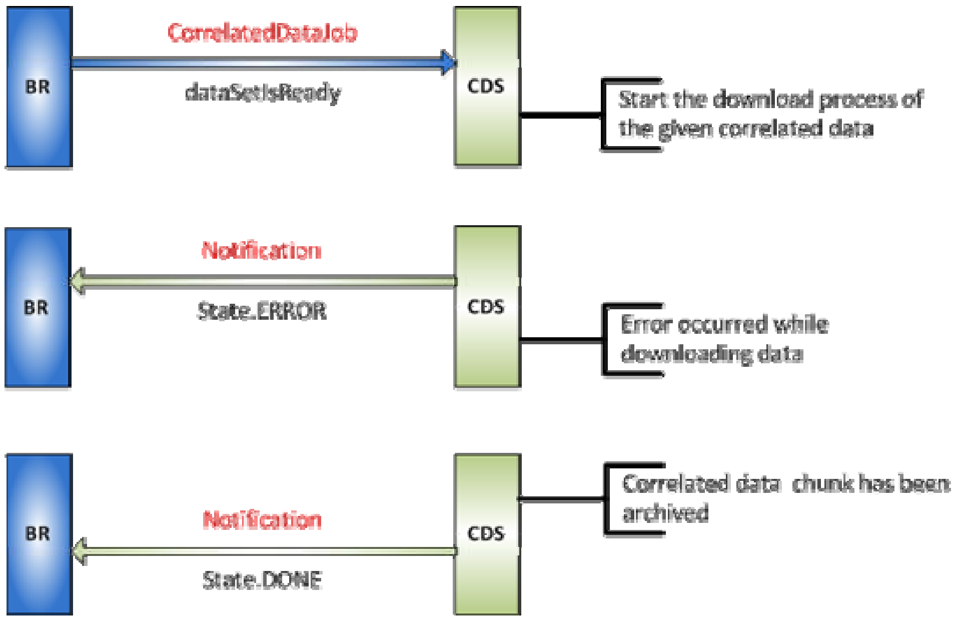


Figure 10 Interaction with Correlated Data Service

3 The e-VLBI System tests

3.1 The system testbed

The e-VLBI System working environment has been prepared where all system components have been deployed. The architecture of the e-VLBI system allows to add new components to the system testebed which will increase the overall performance. It is possible to add more Translation Nodes, as well as Correlation Nodes. We have prepared the following components:

a) VLBI Broker (VB)
   - deployed at Jive (huygens.jive.nl)
   - deployed at PSNC(melisa.man.poznan.pl) – backup module

b) Translation Node (TN)

- deployed at Jive (huygens.jive.nl)
- deployed at Torun (perf.astro.uni.torun.pl)

c) Correlation Node (CN)
- expres.reef.man.poznan.pl (PSNC)
- clusia.man.poznan.pl (PSNC)
- adam.astron.nl (Jive)

d) Expres Network Monitor (NM)
- Jive
- Torun
- Poznan

Locations of all the system components, which were used during test phase are presented in the figure below:

Figure 11 The e-VLBI System testbed

3.2 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.

- o Sometimes the SFXC does not release all resources and does not clean the environment properly/
  - o 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.
  - o There is a list of chunks which never correlate. We were not able to find real reason of this problem.
- By the time e-VLBI system has been tested we had experienced some problems with the NFS system on the reef cluster in Poznan.

3.3 Conducting an experiment

A process of conducting VLBI experiment has been divided into four phases: creating an observation workflow, submitting workflow for execution, workflow execution (the actual correlation) and finally monitoring the correlation process. Each phase is described below.

1. **Creating an observation workflow using a WFM application** An e-VLBI user creates an experiment description (VEX file) using SCHED application. This process is no different than in regular VLBI observations. The VEX is processed in the Workflow Manager Application (WFM) by an e-VLBI

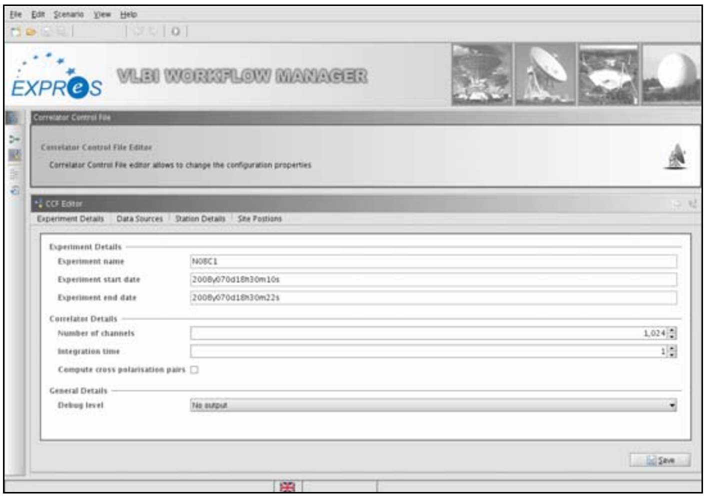


Figure 12 CCF editor

   operator. Experiment control parameters can be verified and modified if necessary. A central e-VLBI operator will be able to associate file servers with radio telescopes. This tells the VLBI System where to stream data from radio telescopes. The last step is to add computational resources which will be used to correlate data. Each computational resource represents a separate, independent distributed software correlator module which can be used to correlate data chunks.
2. **Submitting the observation workflow for execution in the Grid environment** A completed observation workflow with file servers defined and assigned to radio telescopes, with correlation nodes defined and other correlation parameters set up can be submitted for execution using the Workflow Manager Application. The next step – actual data chunks correlation is done behind the scene by the e-VLBI System.

3. **The workflow execution**
The observation workflow, together with observation parameters is sent to the VLBI Broker. The broker module informs all Translation Nodes specified in the

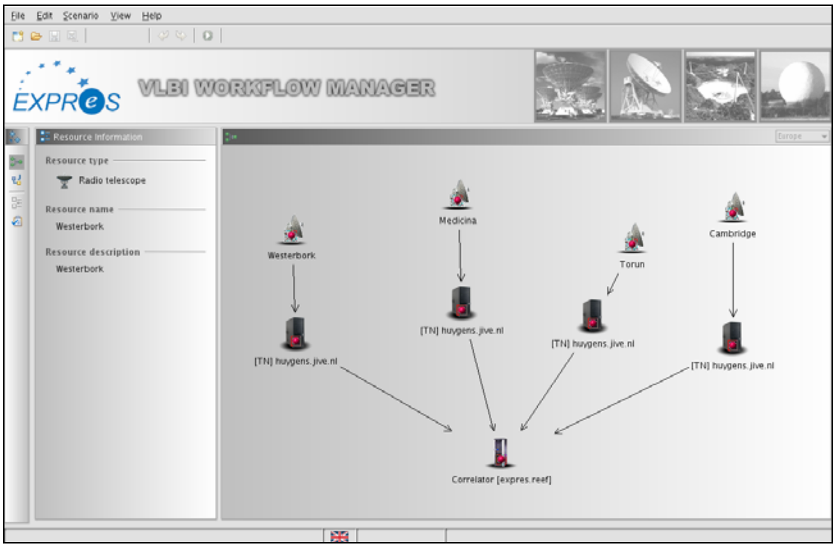


Figure 13 The VLBI workflow

observation workflow about each new experiment. Each TN is responsible for preparing the environment for retrieving data from data source and dividing it to smaller chunks. In the mean time VB informs also each Correlation Node about each new experiment. Each CN prepares environment for chunks correlation. The process of delay table generation is scheduled by CN, before first chunks are ready for correlation. This happens only once, at the beginning of the correlation process.

Whenever each new data chunk is ready for correlation TN module informs the VLBI Broker about its location. The VB broker stores information about chunk locations from all data sources. When a complete data set, within the same time boundaries is collected, it is submitted for correlation. The VB module contacts one of the Correlation Nodes, which handles the correlation process. The decision on which Correlation Node should be used is taken based on internal algorithm. The actual load of each CN is taken into consideration. CN generates a special script which starts software correlator. Please note that a number of CN is configurable and is set up by VLBI operator during stage 1: *workflow creation*. The status of the correlation job is monitored by CN module and propagated back to the VB module. The status can be displayed in the Workflow Manager Application. When a data set has been correlated, CN sends a notification to the VB module with a result location. The VB informs the Correlated Data Service that new data chunk has been correlated. The CDS service is responsible for storing this correlated data product in the Jive archive. When correlated data is stored in the database it is removed, together with data chunks from the computation resources. This process is repeated until all data chunks are correlated.

4. **Monitoring the VLBI experiment**
After the VLBI workflow has been submitted, the correlation progress can be

viewed in the Workflow Manager Application. It is possible to see the total number of data chunks, number of correlated data chunks and finally the number of data chunks, which still need to be correlated.

3.4 Results of sample experiment

The first successful experiment which has been conducted using the e-VLBI System was N08C1 experiment. We have correlated a few seconds of data just to compare the results of the software correlation with the result from the hardware correlator. In our test cases we were correlating data from four different sources: Westerbork, Medicina, Torun and Cambridge. The figure below contains scan details for scan number 33 for our test experiment: N08C1.

```
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 14 Scan details of sample experiment

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

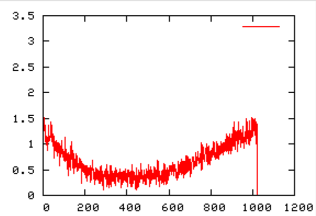
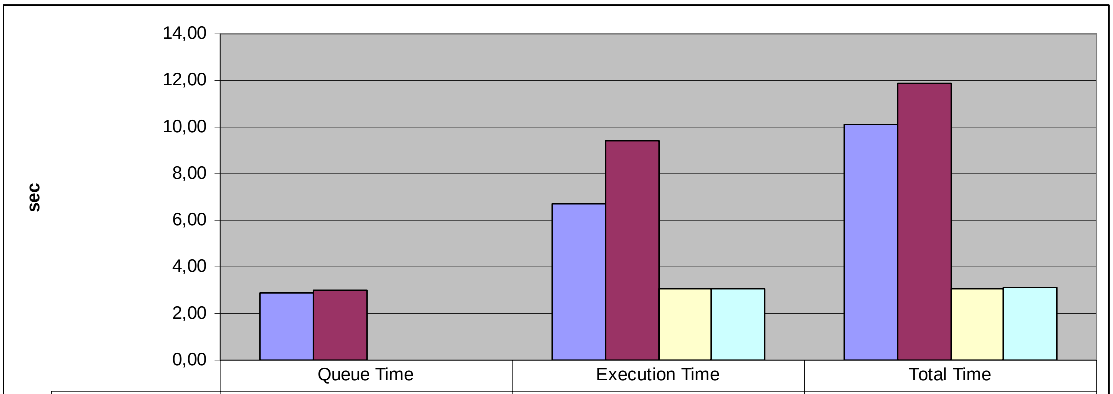


Figure 15 Sample results

The next step was to correlate 25 minutes of data from the same experiment. We have conducted lots of experiments with the same data set, but with different settings of VLBI environment, as well as cluster environment.



	Queue Time	Execution Time	Total Time
Chunk = 1s	2,87	6,69	10,11
Chunk = 2s	3,00	9,40	11,86
MIn - Chiunk= 1s	0,00	3,05	3,06
Min - Chunk = 2s	0,00	3,06	3,14

Figure 16 Chunk execution time

After performing quite a huge number of different experiments it turned out that the best chunk size is one second of data (which in our case is 32 MB). The figure above presents the comparison between different chunk sizes: 1 second and 2 seconds. The chart presents the average and minimum values. As you can see, data chunk is correlated very fast – within a few seconds. The bottle neck of the correlation is the data transfer. The system is idle for long period of time waiting for chunks to be delivered.

We have also noticed not proper work of translation nodes. Please note that in our testbed only one translation node served all the stations. Data chunks for one station were always late comparing to other stations. The average number depending on data chunk was 50 up to 120 chunks. This also slowed down the entire correlation process. The first conclusion is that the e-VLBI system should be equipped with some kind of a mechanism to pause some translation nodes, to let the slowest one deliver the data.

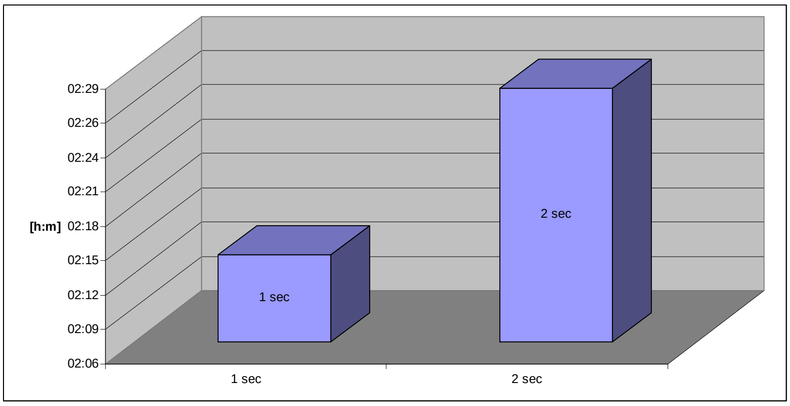


Figure 17 The total execution time

The figure above presents an average, total execution time required to correlate 25 minutes of data for 4 stations. For our use case scenario more than two hours were required to correlate 25 minutes of data. However, in our testbed we had only one sufficient translation node. It is suggested that each station has one unique translation node assigned.

4 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) and Torun (Poland).

The prototype system which has been constructed proved that distributed, software correlation with Grid resources is possible. However, further work is required to create a production based service which could serve constant VLBI observations.