WP

AN ACCESS SYSTEM FOR DATABASES DISTRIBUTED ON EARN

ASTRA Working Group

May 1989

Giuseppe Alberto Romano

CNUCE - CNR Via Santa Maria 36 56124 Pisa ITALY
E-mail ROMANO@ICNUCEVM.CNUCE.CNR.IT

ASTRA System

Abstract

We describe a system intended to access databases distributed on EARN network. Thus system has been developed at CNUCE by the ASTRA project team. ASTRA is a joint project between CNUCE - CNR and IBM Europe to implement a service to distribute, within the EARN network, public domain software produced by IBM study contracts in Europe. The project began in February 1988. The service started, in test mode, the january 1, 1989. The first version accessed the IBM study contracts database and other italian databases located on EARN network. The databases were managed an IBM information retrieval system of IBM named STAIRS/CMS. Afterwards we deciced to interface some other I.R.S systems and a DataBase Management system based on the relational model. Today the system interface uses the same query language as the following host IRS or DBMS system:

- STAIRS/CMS, produced and distributed by IBM
- ISIS, produced and distributed by LAD at UNESCO Paris.
- SQL/DS produced by IBM for the VM/370 operating system.

The SQL/DS relational DBMS is accessed via stored parametric queries in Standard SQL language.

The service is completely free of charge. The system is composed by several ASTRA Database Servers exchanging common requests and database requests of the users. The service is available to all networks which can communicate with EARN using the RFC822 mailer.

Table of Contents

The ASTRA Distributed Database System

General description

The ASTRA Distributed Database System is a system able to access different databases located on different host computers on the EARN network. The system is composed by several Astra DataBase Server (ADBS) connected by the EARN network.

We define an Astra DataBase Server (ADBS) as an IBM/VM370 account located on any IBM VM/370 node of EARN network, which is able to access the databases belonging to its environment, to recognize the location of the databases distibuted on EARN and to send the user's requests to the specific ADBS.

We define an ASTRA user as any user located on a host computer belonging to the EARN network or to other external networks connected to EARN via a RFC822 mailer.

From the previuos definition we may have two different ASTRA users.

1. An ASTRA-EARN user

2. An ASTRA-NON-EARN user

An ASTRA-EARN user can access all the databases of the system using a local interface named Astra User Interface [1] (AUI) which must be located on a read only disk of the host computer where the user is located.

An ASTRA-NON-EARN user can access the ASTRA service sending a mail to an ASTRA Database Server, using a batch language,[2] to query the databases.

The ASTRA batch language may be used also by a ASTRA-EARN user.

[1] See: ASTRA General description or ask to ASTRADB@ICNUCEVM the file ASTRA INFO

[2] See appendix A

The ASTRA user is not required to know the real location of the databases he/she wants to access. It is only necessary to know that the databases belong to the ASTRA system.

Astra Database Server

Since the installation, on a generic EARN node, of a RFC822 mailer is not mandatory, in solving the access to the service using mail feature we have classified the ADBS located in a EARN node as follows:

- The First level ASTRA Database Server is a server located in a EARN node where a RFC822 mailer is available. This server is able to receive the requests from local or remote networks in RFC822 mail format and to reply to them.
- The Second level ASTRA Database Server is a server located in an EARN node where a RFC822 mailer is not available.

A second level ADBS uses as ASTRA-mailer a first level ADBS. To do that, the communication beetween the two different level ADBS is done using the SENDFILE function. This means that when a ASTRA request is transferred using a mail function and requires the access to a database located on a second level ADBS, the request is mailed to a first level ADBS. Then, the first level ADBS transfers the request, in an internal format, to the second level ADBS. Because the second level ADBS cannot use a RFC822 mailer then the results of the request are transferred back to the first level ADBS. At the end, the first level ADBS reply to user's request using a RFC822 mailer.

The definition of a ADBS as a first or second level is done in the system definition table [3] stored in a CMS file named ASTRASYS DEFINE.

To save, if required, the host database system independence more then one Astra database server can be installed on the same node site.

A first or second level ASTRA database server can be defined, locally, as a **master** database server or a **slave** database server.

- A **MASTER** ASTRA database server is a server which receives from the networks the user's requests, is able to execute database accesses and to reply to the user's requests
- A **SLAVE** ASTRA database server is able only to execute database accesses and to reply to the user's requests. It is always ready to receive requests from a **master** ADBS.

[3] See appendix B

The definition of an ADBS as master or slave is done in the astra system definition table [4] defining the MasterAstraServer record.

The communication between a master ADBS and a slave ADBS is performed by the system using an internal format.

ADBS and AUI interaction

The interaction between any Astra Database Server (ADBS) and the Astra User Interface has been defined to simulate a pseudo-online access to the host database system owner.
There is no acknowlegement between any ADBS and the AUI, but the ADBS is always ready to accept the request of the users.
The database accesses are executed in batch mode by the host database system interface. When the access to the database ends, the results are modified into the standard form defined for ASTRA service results, and then are transferred to the user.
If the results are transferred to the user as messages, no acknowledgement is required by the ADBS, from AUI, to transmit data.
The messages received by any ADBS are stored and displayed in the message area of the AUI panel.

All the accesses to the databases are logged on a file located on every ADBS with the number of documents selected. If an ADBS is not a database owner, the logged informations contain the name of the EARN node where the owner ADBS is located.

ADBS and AUI communication

The communication between any Astra Database Server (ADBS) and the Astra User Interface (AUI) has been defined by the use of two different protocols:

1. MSG (message) protocol, using standard TELL command.

2. FTP (file transfer protocol), using standard SENDFILE command.

The MSG protocol is a very fast way of communication but, because the EARN characteristics, do not guarantee that the messages sent by the AUI are always received by the ADBS. If some nodes, between the AUI and ADBS, are down the messages are lost, and the ADBS sends to the user a warning message. But if all the messages will arrive to the ADBS, the following message is sent to the end user.

Your request has been received.

[4] See appendix B

The FTP protocol uses the store-and-forward function of the EARN network. The commands are stored in a standard internal ASTRA communication format and are transferred as a file, using SENDFILE command, to the ADBS. Using this protocol the user is sure that his request will be forwarded to the ADBS but the response time may be very long because the command file is queued.

The receiving ADBS replies to the user using the FTP protocol or using the MSG protocol according the way as the request has been sent. This means that the ADBS always replies to a MSG protocol request with several TELL command. The same happens for the FTP protocol request.

For more detail see appendix D

Appendix A

ASTRA batch query language

```
StartRequest: Username: <username> Host: <host>
Dbname: <dbname> Typeresult: <typeresult>
= <querytext>
= <querytext>
------
= <querytext>
Printformat: <format>
Dbname: <dbname> Typeresult: <typeresult>
= <querytext>
= <querytext>
------
= <querytext>
Printformat: <format>
------
------
------
Printformat: <format>
Dbname: <dbname> Typeresult: <typeresult>
= <querytext>
= <querytext>
------
= <querytext>
Printformat: <format>
EndRequest:
```

where:

- <**username**> is the name of the user.
- <**host**> is the computer name site.

- <**dbname**> is the name of one of the available databases on the ASTRA system.
- <**typeresult**> is the type of required result:
  - Numeric : the number of the selected documents.
  - Document : the documents selected.
  - Software : the software belonging to the selected documents.
- <**querytext**> is the query request in the STAIRS/CMS [5] query language or in the Host system query language (SQL stored query [6], ISIS query language[7])
- <**format**> is the format required :
  - STAIRS mail format specification
  - ISIS stored format name
  - SQL interface specification: H=Y/N,F=T/C
    - ▲ H=Y/N : if the heading is required or not
    - ▲ F=T/C : if a Tabular or Commadelimited format is required.

[5] See: STAIRS/CMS Information Retrieval Guide - IBM SH12-5366-2

[6] See: SQL/Data System Terminal User's Guide for VM/System product IBM SH24-5045-1

[7] See: Unesco Computerized System CDS/ISIS Terminal Operator's Manual

EXAMPLES

```
startrequest: username: ROMANO host: ICNUCEVM
dbname: star typeresult: numeric
= campus and network
printformat:
dbname: soft typeresult: document
= campus and network
printformat: all
endrequest:
```

The <**astra-database-request**> listed below is composed by two <**database-request**> [8]

- The first request looks for how many documents exist in the database STAR about **CAMPUS AND NETWORK.**
- The first request requires the documents contained in the database SOFT about **CAMPUS AND NETWORK.**

[8] See appendix C

Appendix B

ASTRA system definition table

StartAstraDefinition:
MailerDefinition: type: <**type**> mailer: <**mailer**> mailerhost: <**mailerhost**>
LogDefinition: type: <**logtype**> device: <**logdevice**>
MasterAstraDefinition: user: <**masteruser**> host: <**masterhost**>
MasterAstraServer: user: <**masterserver**> host: <**masterhostserver**>
EndAstraDefinition:

where:

1. MailerDefinition: define if a RFC822 mailer is locally installed.
   - <**type**> defines if the Astra Database Server is first or second level. It can be:
     - REMOTE: for a second level server
     - LOCAL: for a first level server
   - <**mailer**> defines the name of a ASTRA Database Server of first level to which the second level server must refer.
   - <**mailerhost**> defines the node identification of a first level server to which a second level server must refer.
2. LogDefinition: defines the log activity.
   - <**typelog**> defines how the log must be recorded. May be:
     - START: all the operation performed by the server and all the messages received by the users are logged.
     - STOP: only the most important operation are logged.

  - OFF: nothing is logged.

- <**logdevice**> defines the device on which write log information:
  - DISK: the log is registered on a CMS file named ASTRA LOG
  - CONSOLE: the log is registered on the vm console log

3. MasterAstraDefinition: define the user name and the node identification of responsible persons of local ASTRA Service.

   - <**masteruser**> is the user name of the resposible persons for the ASTRA Service
   - <**masterhost**> is the node identification where the responsible persons of local ASTRA Service are located.

4. MasterAstraServer: defines the name of the local master ASTRA database server. A slave ASTRA database server is a server which has in its system definition a <**masterserver**> and a <**masterhostserver**> different in the account name and node identification on which the server is running.

   - <**masterserver**> is the account name of the master ASTRA database server.
   - <**masterhostserver**> is the node identification where the master ASTRA database is located.

EXAMPLES

Today we have installed five Astra Database Servers with different configuration.

- **ASTRADB at ICNUCEVM**, located in Pisa, with the following system definition:

```
StartAstraDefinition:
MailerDefinition: type: LOCAL mailer: ASTRADB  mailerhost: ICNUCEVM
LogDefinition: type: STOP device: DISK
MasterAstraDefinition: user: ROMANO host: ICNUCEVM
MasterAstraServer: user: ASTRADB  host: ICNUCEVM
EndAstraDefinition:
```

  ASTRADB at ICNUCEVM is a master first level Astra database server.

- **ASTRASQL at ICNUCEVM**, located in Pisa, with the following system definition:

```
StartAstraDefinition:
MailerDefinition: type: LOCAL mailer: ASTRADB  mailerhost: ICNUCEVM
LogDefinition: type: STOP device: DISK
MasterAstraDefinition: user: ROMANO host: ICNUCEVM
MasterAstraServer: user: ASTRADB  host: ICNUCEVM
EndAstraDefinition:
```

  ASTRASQL at ICNUCEVM is a slave first level Astra database server because the use of IUCV functions to access SQL/DS databases.

- **ASTRADB at IFIBDP**, located in Florence, with the following system definition:

```
StartAstraDefinition:
MailerDefinition: type: REMOTE mailer: ASTRADB  mailerhost: ICNUCEVM
LogDefinition: type: STOP device: DISK
MasterAstraDefinition: user: ROMANO host: ICNUCEVM
MasterAstraServer: user: ASTRADB  host: IFIBDP
EndAstraDefinition:
```

  ASTRADB at IFIBDP is a master second level Astra database server and uses ASTRADB at ICNUCEVM as ASTRA-mailer.

- **ASTRADB at IFIIDG**, located in Florence, with the following system definition:

```
StartAstraDefinition:
MailerDefinition: type: LOCAL mailer: ASTRADB  mailerhost: IFIIDG
```

```
LogDefinition: type: STOP device: DISK
MasterAstraDefinition: user: ROMANO host: ICNUCEVM
MasterAstraServer: user: ASTRADB  host: IFIIDG
EndAstraDefinition:
```

ASTRADB at IFIIDG is a master first level Astra database server.

- **ASTRADB at IRMKANT**, located in Rome, with the following system definition:

```
StartAstraDefinition:
MailerDefinition: type: REMOTE mailer: ASTRADB  mailerhost: ICNUCEVM
LogDefinition: type: STOP device: DISK
MasterAstraDefinition: user: ROMANO host: ICNUCEVM
MasterAstraServer: user: ASTRADB  host: IRMKANT
EndAstraDefinition:
```

ASTRADB at IRMKANT is a master second level Astra database server and uses ASTRADB at ICNUCEVM as ASTRA-mailer.

Appendix C

ASTRA internal language format

We define a **<astra-database-request>** as a sequence of one or more **<database-request>** as follows:

<database-request>
<database-request>

<database-request>

Where:

<database-request> is a request addressed to any database belonging to ASTRA service to obtain:

The number of documents matching the user's request.

The documents matching the user's request.

The software and/or the report belonging to the documents matching the user's request.

Any **<database-request>** is structured according the following scheme

<start-request>
<search-request>

<search-request>
<print-request>

< stop-request >

1. **< start-request >**

   EXEC < procname > < dbname > ASTRA < vmid > < nodeid >

   where:

   EXEC define the beginning of the request for < dbname >

   < procname >: is the name of the ADBS procedure to be executed

   Today the procedures are:

   ASTRAQ: is used to obtain the number of documents or the documents matching user's request.

   ASTRAF: is used to obtain the software and/or the reports linked to the selected documents.

   < dbname >: is the name of the selected database. This name must exist in the system table named ASTRASYS DBTABLE where all the accessible databases and their EARN location are listed.

   ASTRA: is the name of a command file where the query language will be temporarely stored.

   < vmid >: is the name of the VM account of the user.

   < nodeid >: is the identification of the EARN node where the user is located.

2. **< search-request >**

   = < stairs-query >

   where:

   The symbol = defines the beginning of a **< stairs-query >**:

   < stairs-query >: is a request in the STAIRS/CMS query language [9]

3. **< print-request >**

   MAIL < qn > < format > TO DOCU

   where:

   MAIL: prints the documents

[9] See: STAIRS/CMS Information Retrieval Guide - IBM SH12-5366-2

<**qn**>: is the query number to be printed

If we have de following command:

```
EXEC ASTRAQ CARTOONS ASTRA ROMANO ICNUCEVM
= LINUS AND LUCY OR SNOOPY
= 1 AND (SNORKEL OR ZERO)
MAIL 2 A TO DOCU
OFF
```

The <**print-request**> is applied to the last query where we have the first and the second requests in logical ″and″.

<**format**>: is the print format according STAIRS/CMS print formatting language [10]

TO: sends the request on a print file.

DOCU: is the name of a print file where the results will be printed.

4. <**stop-request**>

OFF ends the access to the selected database.

[10] See: STAIRS/CMS Information Retrieval Guide - IBM SH12-5366-2

Appendix D

ASTRA communication protocol

The Astra User Interface, to communicate with any Astra Database Server using FTP protocol stores an <**astra-database-request**> into a file named ASTRA FILE, which will be sent to an ADBS using standard SENFILE command.

If the user chooses to communicate with ADBS using the MSG protocol the components of the <**astra-database-request**>[11]will be sent as a message. When all the messages will arrived to the ADBS, the <**astra-database-request**> will be rebuilt into the original format.

Any message has the following format:

<**command**> <**current-line**> <**total-line**> <**vmid**> <**nodeid**> <**request**>

where

<**command**>: is the receiving command. It may be:

<**start-command**>: starts the trasmission of the commands. Its value is: $>#

<**stop-command**>: ends the trasmission of the commands. Its value is: $<#

<**line-command**>: means that the current line contains a command. Its value is: $-#

[11] See Appendix C

< current-line >: is the current line command number.

< total-line >: is the total line commands to be transferred for the current **< astra-database-request >**

< vmid >: is the name of the VM account transferring the request.

< nodeid >: is the name of the EARN node where the user is located.

< request >: may bee one of the following definition:

- **< start-request >**
- **< search-request >**
- **< print-request >**
- **< stop-request >**

Because an **< astra-database-request >** may contain more then one **< database-request >**, we may have the communication performed in such way:

```
< start-command >  < first-line >  < total-line >  < vmid >  < nodeid >  < start-request >
< line-command >  < current-line >  < total-line >  < vmid >  < nodeid >  < search-request >
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
< line-command >  < current-line >  < total-line >  < vmid >  < nodeid >  < search-request >
< line-command >  < current-line >  < total-line >  < vmid >  < nodeid >  < print-request >
< line-command >  < current-line >  < total-line >  < vmid >  < nodeid >  < stop-request >
< line-command >  < current-line >  < total-line >  < vmid >  < nodeid >  < start-request >
< line-command >  < current-line >  < total-line >  < vmid >  < nodeid >  < search-request >
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
< line-command >  < current-line >  < total-line >  < vmid >  < nodeid >  < stop-request >
< line-command >  < current-line >  < total-line >  < vmid >  < nodeid >  < start-request >
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
< stop-command >  < last-line >  < total-line >  < vmid >  < nodeid >  < stop-request >
```

and in a real example:

```
$>#  1 10 ROMANO ICNUCEVM EXEC ASTRAQ CARTOONS ASTRA ROMANO ICNUCEVM
$-#  2 10 ROMANO ICNUCEVM = LINUS AND LUCY OR SNOOPY
$-#  3 10 ROMANO ICNUCEVM = 1 AND (SNORKEL OR ZERO)
$-#  4 10 ROMANO ICNUCEVM MAIL 2 A TO DOCU
$-#  5 10 ROMANO ICNUCEVM OFF
$-#  6 10 ROMANO ICNUCEVM EXEC ASTRAQ BOOKS ASTRA ROMANO ICNUCEVM
$-#  7 10 ROMANO ICNUCEVM = LINUS AND LUCY OR SNOOPY
$-#  8 10 ROMANO ICNUCEVM = 1 AND (SNORKEL OR ZERO)
$-#  9 10 ROMANO ICNUCEVM MAIL 2 A TO DOCU
$<# 10 10 ROMANO ICNUCEVM OFF
```