EMS

Electronic Mail System
Version 0.0

by Davide Massarenti

1 Introduction

The purpose of EMS is made clear by the meaning of its name: to manage in an easy and uniform way an Electronic Mail System. The word "Mail", here, has a wider meaning than that of "Messages", since it also includes files, much the same way conventional mail also dispatches parcels.

In an uniform way, I said. In fact, unlike other products, EMS can be used with FIDONET technology networks AND with USENET-kind network, with the same kind of interaction. The software takes on itself all the work required to overcome the differences between different networks.

EMS is not a single, huge executable file, but rather a set of modules, configured as run-time libraries, each dealing with a different task. The whole thing only runs with Kickstart 2.0 or newer: it's both a choice (Kick 2.0 is a major step onwards if compared with earlier versions of the O.S), and a necessity, since BOOPSI only comes with the new operating system.

Now what has BOOPSI ("Basic Object-Oriented Programming System for Intuition") to do with us? Isn't it something that has to do with graphics, and not with electronic mail...? False, we need it. First, because the software relies, for its graphic part, on a library (odd, isn't it?!?) called ObjectiveGadTools.library, which makes possible to get very 2.0-compliant interfaces, automatically adapting to the selected font, be it fixed-width or proportional: never more topaz 8 on 1000x1000 screens! Second, because BOOPSI is much like a philosophy, the philosophy of object-oriented programming with dynamic binding.

"Sounds nice. But what is the advantage for me, earthly user??"

What you get is a more reliable piece of software, able to perform extremely complicated tasks in an easy way. Here's a quick example. EMS deals not only with messages, but also with files. From within the editor, you can read the text of a message, or switch to the files database and read the descriptions of the files in the BBS, add a new file, delete it, or send it to someone. All ARexx commands that can affect a message can also affect a file in the same way, and this is only made possible by the use of BOOPSI. Of course you do not get only advantages; there is a disadvantage too: less speed than with a traditionally written program, dealing with the same task. In other word, if a traditional software and one written using BOOPSI can perform the same task, the traditional software *may* be faster. But with a closer look, you would soon notice that the faster program chokes on tasks that the other software can perform easily.

2 The System

EMS is distributed on many subdirectories of a single directory, assigned as ‘EMS:’

```
EMS:
 |
 + bin          --> EMS_Editor ...
 |
 + db           --> BASIC_db
 |
 + libs         --> EMS.library ...
 |
 + config       --> AKAs
 |
 + configCustom --> Origins ...
 |
 + rexx         --> Import.ems WriteMsg.ems ...
```

We already pointed out that EMS is not a single program, but rather a set of libraries. The only actions needed to boot it up are:

```
Assign EMS:  .
Assign LIBS: EMS:libs REMOVE >nil:
Assign LIBS: EMS:libs ADD    >nil:
```

On the other hand, to shut everything down, you only need to launch a script called ‘EMS:rexx/ShutDown.ems It is very important to remember to launch the script before each reset or switching off, to be sure that files are properly updated and closed. Libraries are closed, but not immediately expunged from memory, which happens when you execute Avail Flush or when the available memory pool is getting small.

If you are a registered user, you must put the 'EMS.key'[1] file in the 'EMS:' directory, for the program to properly recognize it.

3 Address Format

EMS supports 3 kinds of addressing: FIDONET, UUCP, USENET.

FIDONET addressing has the form:

```
<zone>:<net>/<node>[.<point>]@<domain>
```

Beware of the difference between 2:332/505@fidonet.org and 2:332/505.0@fidonet.org: the first is a node, the second not. The software tries, wherever possible, to figure out the missing part(s) of an address: for example, when writing a message, 505.3 will be expanded to 2:332/505.3, provided that the address set for that area is e.g. 2:332/505@fidonet.org. Please note that addresses should always be written in their complete form in the configuration files.

UUCP addressing has the form:

```
<name>! ... !<name>!<user name>
```

where <name> are the net names of the machines at a different hierarchy. If it's an address of yours, the rightmost names are the names of the nearer systems, and the last on the right is the name of your system. When you need to specify a specific machine, such as your own, you only write <name>.UUCP.

USENET addressing has the form:

```
[<user name>@]<subdomain>. ... .<domain>
```

[1] GCCHOST V4.0 users should rename this way their keyfile.

For example, you can convert a FIDONET address in USENET format, as in

```
2:332/505.3@fidonet.org  -> Davide@p2.f505.n332.z2.fidonet.org
```

All these formats are fixed, i.e. they only specify a single address. There is a situation in which you need a format to specify whole groups with a single stroke, and that's `Routing`. `From` addresses in the `Routing` configuration, and only they, can have a slightly different format, called **wild**, where you can add in some modifiers:

`*` Stands for any number of characters, including zero. For example, `2:33*/*@fidonet.org` is equivalent to *All nodes in fidonet zone 2, whose net numbers begin with 33.*

`?` Stands for one and only one character. For example, `2:332/505.?*@fidonet.org` is equivalent to *All points of node 2:332/505@fidonet.org*, because it states that there must be at least a character after the point, and it only happens with point addresses.

`~` Can only be put in as the first character of a field, and its purpose is to invert the pattern: in other words, it acts as a logical *NOT*. For example, `2:332/~5??@fidonet.org` is equivalent to *All nodes which are NOT in nets beginning with 5.*

Note that `*:*/*.*@fidonet.org` designates all systems in fidonet.

The **wild** format for UUCP and USENET addresses is a bit different. For UUCP addresses, if the pattern is made up by only one word (that is, it's the name of a particular machine), it is compared with the names of the known machines, from right to left.

For example, let us take an address, `cbmehq!cbmita!ipisa!esmae`, and a few pattern, pointing out which of them include the given address:

```
ipisa.UUCP     -> yes
cbmita.UUCP    -> yes
pippo.UUCP     -> no
cbm*           -> yes
cbmita!ipisa!* -> yes
cbmehq!ipisa!* -> no
*@ipisa.*     -> yes
```

As we will see, this is exactly what we need to route mail through different nodes. In this way, if you communicate with UUCP systems called 'ipisa' and 'gear', a message directed to a node which is accessible through 'ipisa' (as in our example) will be sent through 'ipisa', and all the others will be sent through 'gear'.

4 Configuration

The configuration is made up by a set of ASCII files, located in the '`EMS:Config/`' subdirectory. The format is very simple: a series of `<field name>=<field value>`, spread over one or more lines. For each section in the configuration there is a set of so-called 'marker' keywords, which mark the beginning of a series of values that are related to one another; for example, in the configuration of the nodes your system usually communicates with,

```
ADDRESS=2:332/505@fidonet.org
```

means that the following values are related to node 2:332/505 of fidonet. A blank line is used to separate the settings for two different entities (variables, nodes, etc.).

It's not required to enter all the keywords available for each configuration file: all uninitialized fields will assume a reasonable default value, usually a null string or zero.

Besides, there are a few cases of keywords without values (see Section 4.6 [AREAs], page 19), and values without keywords (see Section 4.7 [LINKs], page 22, Section 4.9 [USERLIST], page 25).

`<field value>` is considered to end at the first blank space: if you need to include a space in it you have two ways to do it. You can enclose the whole value with quotes ("), or use the '\' (backslash plus space) sequence instead of a space. Similar sequences (backslash plus character), must always be used if you need to insert quote symbols, or backslashes:

```
Due Parole     -> "Due Parole"

Due Parole     -> Due\ Parole

"Due" Parole   -> "\"Due\" Parole"
```

When you alter one of the configuration files, there is no warranty that changes will be immediately used by the software; the configuration is read only once at startup. Before every change of configuration, then, run the '`EMS:rexx/ShutDown.ems`' script to properly terminate execution.

The user is not the only one who can modify the configuration; the software itself can do it, as e.g. when a new area is created and configured during an import. The software always performs a backup copy of the file(s) it is going to modify: subsequent copies have their index incremented, as in (`<cfg>`, `<cfg>.1`, `<cfg>.2`, etc.).

GCCHOST V4.0 users can rescue a large part of their existing configuration using the GCC2EMS utility. This program allows both to convert the old configuration files, and simply to display the data contained in them. It is possible to convert separately each of the three files that made up the configuration for GCChost ('`GCC.cfg`', '`AREAS.BBS`' and '`AREAS.BBS.EXTRA`'), but it's better to work simultaneously on all the files, because information is often spread between different files.

An advice: make a backup copy of the example configuration you found in the EMS archive, delete all files in '`EMS:config/`' and launch GCC2EMS. Then check the result by hand, comparing the generated files with those given as example, with particular attention to externals and routing, which are the topics that underwent the most significative changes (you should really consider to put the example externals back after performing the conversion...).

4.1 SWITCHes

The file we are dealing with is '`EMS:Config/Switches`'. It contains a series of pairs of names, followed by a status keyword, indicating which of the two is active. These are the keywords:

On=<name> *marker*

The name of the active status.

Off=<name>

The name of the inactive status.

Status=<boolean>

Indicates the status of the defined pair, and can assume `ON`[2] or `OFF` values. It must always be put after the pair, and not before it.

[2] Acceptable values for all <boolean> fields are: ON, OFF, TRUE, FALSE, YES, NO

Configuration Example:

```
Off=NO_RENUM_AFTER_IMPORT On=RENUM_AFTER_IMPORT     Status=ON

Off=RENUM_BY_ARRIVAL_TIME On=RENUM_BY_CREATION_TIME Status=ON

Off=SEARCH_ON_TO          On=IGNORE_TO              Status=ON
```

Blank lines, as we have already seen, are used to separate different parts of the file.

The system uses a particular set of pairs, listed below, to control its behaviour. All pairs are ordered so that the `Off` item, which is the default if `Status` is not specificed, is in the first place.

NO_RENUM_AFTER_IMPORT
RENUM_AFTER_IMPORT

> If this is ON, it forces a reorganization of all areas that received new messages after an import. It is useful to avoid 'holes' in the message numbering, due for example to messages that have been deleted.

RENUM_BY_ARRIVAL_TIME
RENUM_BY_CREATION_TIME

> Except for import, all renumbering operations are controlled by the status of this pair. The default, `RENUM_BY_ARRIVAL_TIME`, sorts messages (or files) by import date, while `RENUM_BY_CREATION_TIME` sorts them by creation date.

SEARCH_ALL
SEARCH_NEW

> The status of this pair affects the `EMS_Database_Search` command, which scans the database(s) for particular messages. `SEARCH_NEW` only considers the ones which have just been imported.

IGNORE_TO
SEARCH_ON_TO
IGNORE_FROM
SEARCH_ON_FROM
IGNORE_SUBJECT
SEARCH_ON_SUBJECT
IGNORE_DATE
SEARCH_ON_DATE
IGNORE_TEXT
SEARCH_ON_TEXT

When one of these pairs is ON, the corresponding field will be considered in a search operation. You may set only `SEARCH_ON_TO` to scan the database for messages to you, or `SEARCH_ON_TEXT` to locate messages with a particular word in their text.

USE_KILL_FLAG
IGNORE_KILL_FLAG

Normally, if a message is marked as `KILLSENT`, it will be deleted immediately after being exported. With this switch, you can force EMS to ignore the `KILLSENT` flag.

USE_SENT_FLAG
IGNORE_SENT_FLAG

When you export an area, not all messages get processed. Each area maintains a number, called *High Water Mark*, corresponding to the last exported message. Messages numbered below or equal to the HWM will not, and cannot, be processed in an export; the opposite is not always true, and for a message higher than HWM to be exported, two more conditions must be verified:

1. (only valid for areas not marked as `MAIL`): the message must not have been already seen by the system we are exporting to. This can happen, for example, when that message originally came from that system; in this case, there's no way to export the message without altering it.

2. The `SENT` flag, which is normally set in each processed message, must not be set. This is meant to avoid messages being processed twice, but can be overcome by `IGNORE_SENT_FLAG`.

DO_BACKUP
NO_BACKUP

Controls the automatic backup of the inbound mail. Every time you perform an import, if this switch is set on DO_BACKUP, all files are backed up in a directory set in the `BACKUP_DIR` variable before being processed. Please note that the backup is now done a file at a time, in order to reduce the amount of disk space needed.

NO_KEEP_MATRIX
KEEP_MATRIX

Only useful for nodes. Allows you to keep matrix mail in transit, beside those addressed to your system.

AREAFIX_INTERNAL
AREAFIX_EXTERNAL

Enables the use of the EMS-specific AreaFix script.

KEEP_AREAFIX
NO_KEEP_AREAFIX

Used by the AreaFix script to decide whether processed AreaFix requests should be deleted or not.

KILL_DUPES
KEEP_DUPES

When a message is recognized as a duplicate, the normal behaviour is to delete it. With KEEP_DUPES, a copy of it gets kicked in the `BAD_MSGS` area. The duplicate-recognition scheme is based upon the identifying data *^aEID:*, *^aMSGID:* e *Message-ID:*. It is a very reliable method, and in normal conditions you can leave `KILL_DUPES` without worrying.

NO_ROUTING
USE_ROUTING

Useful for points with particular mailing needs, who can activate custom routing for their messages. The default is automatic routing; each message is exported to one of your bosses, provided that it does not have optional flags set (see below)

CRASH_NORMAL
CRASH_TO_BOSS

When automatic routing is in effect, `CRASH`-flavoured messages can be packed directly for the addressee's node, or be routed through one of your bosses. `CRASH_TO_BOSS` is not used often, since if your boss is not willing to make a phone call for you, your message will be treated as a normal message.

HOLD_NORMAL
HOLD_TO_BOSS

When automatic routing is in effect, `HOLD`-flavoured messages can remain in your outbound directory, waiting to be picked up by the addressee, or being routed through one of your bosses. The `HOLD_TO_BOSS` setting is more common, because a point seldom gets called, and in most cases it's not a trouble for the boss to keep a file waiting for someone to pick it up.

FILES_STOP
FILES_FORWARD

Allows or disallows file attach capability in `MAIL` areas.

4.2 VARIABLEs

The file we are dealing with is '`EMS:Config/Variables`'. It contains a series of name/value pairs, which are used to choose e.g. the editor for message entering, the directory to put backup copies in, etc. There are only two keywords:

Name=<name> *marker*

The name of the variable.

Value=<string>

Its value.

Configuration Example:

```
Name=TEMP_DIR  Value=T:

Name=LOG_FILE  Value=EMS:EMS.log

Name=LOG_LEVEL Value=0
```

As with switches, there is a set of variables that have a particular meaning to EMS:

SYSOP_NAME

The name of the system operator. For registered users, this is read from the keyfile.

LOG_FILE

The name of the log file, i.e. the file in which EMS writes a summary of its actions. A log is always written, unless this variable is deleted.

LOG_LEVEL

This is the verbosity level for the log. Error reporting is always included, while ARexx commands never generate output. Therefore, only the tosser and scripts will have their behaviour affected.

At the lowest level, 0, there are messages such as:

```
Backing up <file> to <dir>
Export msg <num> ...
Message from <address> for ...
Post import msg <num> ...
Rescan msg <num> ...
```

The following level, 1, includes origin declarations: i.e, where do things come from:

```
Packet from <address> to <address>
Tick from <address> for area <name>
Import file <file> from <address>
```

At level 2 we find, let's say, statistical information:

```
Imported <num> msgs from <pkt>
Exporting area <name>
Post Importing area <name>
Rescanning area <name>
Arcing for <address> (<num> > <num> - <num>%)
Unarcing <archive>
```

At level 5 you only get very general information about what is happening in the system:

```
Importing <archive>
Importing <pkt>
Importing <tick>
```

Note that the log level is not limited between 0 and 5, but these are the only meaningful values for the system. If you do not want a report, except for error messages, you can set the log level to something quite high, for example `LOG_LEVEL=100`.

TEMP_DIR

This is the directory where mail archive are unpacked during an import, and a place where to put temporary files in general.

BACKUP_DIR

This is the directory where files are backed up before being processed in an import.

UNARC This is the default dearchiver's name. Unlike GCChost, this is not the complete command to be executed, but only the symbolic name that has been given to that particular (un)packer. For more information, see Section 4.8 [ARCHIVERs], page 23.

ARC As `UNARC`, but this is related to the packing process.

DB_MSG_NAME

This is the message database name to be used by default when creating new areas. See the script @ref ListDBnames.ems .

DB_FILE_NAME

This is the file database name to be used by default when creating new areas. See the script @ref ListDBnames.ems .

NEWAREAS_MSG_PATH

If EMS, during an import, finds a message for an area that doesn't exist, and the system that message comes from has been given the ability to create new areas, this variable is employed as a pattern to get the directory name that will be used for the new area's messages. `%n` or `%N` sequences will be filled in with the name of the area; `%d` or `%D` will become a random number, and EMS will take care that the resulting directory name is not already in use on the disk.

NEWAREAS_FILE_PATH

This variable has a purpose very similar to NEWAREAS_MSG_PATH, but it is related to file areas, i.e. areas marked as `TICK` or `FILE`.

NODELIST_PATH

For FIDONET technology networks, this is the directory where the nodelist can be found. Such nodelist must be managed with TRAPLIST.

ORIGIN_NAME

This is the default origin for messages entered in non-`MAIL` areas.

AREAFIX_NAME

This is the name AreaFix messages for your system should be addressed to.

DUPECHECKING_SIZE

Default size, expressed in messages, of the databases used for duplicate-checking data. You can set this value also on an area by area basis with the `NumOfDupes` parameter (see Section 4.6 [AREAs], page 19).

MIN_FREEMEM_SIZE

When processing an area, EMS must read the contents of the messages in it. To do so, it uses, about 250 bytes for each message. It's easy to see that a large system, with dozens of areas and thousands of messages, needs a large amount of memory for an import, because it needs to read the data for all areas in which even a single message must be imported. If the software were given full freedom of memory allocation, this would soon bring to the exhaustion of the free memory pool, with errors both from EMS and from other programs (such as an archiver, launched by EMS at the end of the import). It is of basic importance to put a limitation to the program's voracity. It's never required to have data for all areas in memory at the same time; yet, this can prove to be really practical, because you don't have to wait when you switch from one area to another. You can go for a compromise with `MIN_FREEMEM_SIZE`, representing the minimum amount of memory that should be free in the system in a single block; if, at any given moment, EMS realizes that free memory is less than asked, it will begin to release memory taken up by the least used areas. It's not advisable to use a value less than 200000.

MAX_PACKET_SIZE

If your system processes a large amount of mail, you can take advantage of this value, which stands for the largest size that mail packets will have. Raising the number of packets enlarges the resulting archive by only a few bytes, but allows both you and the receiving system to process mail more safely (less chance of running out of memory and disk space). A good value is 200000.

There are other variables, whose values don't affect the software directly, but are needed by ARexx scripts:

SCRIPT_REXX_DIR

The directory in which ARexx scripts are put.

SCRIPT_TEXT_DIR

The directory where marked messages should be archived in text form.

SCRIPT_MSG_FILE

The complete name, including path, of the temporary file used to pass the message text to an external editor, then back into EMS.

SCRIPT_EDITOR_NAME

This is the command to be executed to launch the text editor. The name of the file to edit is appended by the scripts.

SCRIPT_EDITOR_ARGS

Some editors need arguments AFTER the file name; such arguments must be put in this variable.

SCRIPT_DELETE_NAME

This is the name of the command used to delete files.

4.3 AKAs

The file we are dealing with is '`EMS:Config/AKAs`'. It contains all the addresses of the system in the different nets. Here are the keywords:

<address> *marker*

An address, in FIDONET, UUCP or USENET format:

```
2:332/505@fidonet.org
ipisa
mother.gear.sublink.org
```

This is also an example of a field without keyword: every line not beginning with `Fake` is considered to be a valid address, and will generate an error message if it is not. Registered users with a node licence can freely mix FIDONET node and point addresses; you can have some areas as a point, and others as a node, on the same system.

Fake= <fake net>

When communicating with old FIDONET software, you need to set the so-called *Fake Net*. It is a number that will be used to send and receive mail to and from old-fashioned points: an address such as `332/505.3` will become `<fakenet>/3`.

Configuration example:

```
2:332/505@fidonet.org     Fake=22505

39:102/511.3@amiganet.ftn Fake=22511

mother.gear.sublink.org
```

4.4 DOMAINs

The file we are dealing with is '`EMS:Config/Domains`'. It contains information about where to put outbound mail, and where to find inbound packets. The keywords are:

Name=<name> *marker*

The name of the net, also called *Domain*. It's important that the names of the domains be consistent with those used in your AKAs, because the tosser will extract the domain from the address of the addressee, and look for a matching outbound directory: if it cannot be found, it will abort. The matching process is different for FIDONET and UUCP addresses. In the first case, only the part of the domain on the left of the first dot will be considered: for example, `fidonet.org`, `fidonet.org.earth` and `fidonet` are considered equivalent. In case of an UUCP address the rightmost part will be taken into account; but since the case of more than one UUCP feeder is very unlikely, the name of the net will most probably be `UUCP`. As an advice, try to get accustomed to only one form of writing a domain's name; don't type `fidonet.org` here and `fidonet` there. It does no harm, but it's indeed a bad habit.

Inbound=<inbound dir>

This is the directory where the inbound mail can be found.

Outbound=<outbound dir>

This is the directory where the outbound mail should be put.

AltOutbound=<secondary outbound dir>

Secondary directory where the outbound packets should be put. If set up, EMS moves here the mail packets, so to thin the main outbound on big systems, where you can find hundreds of files. Having less files in the main outbound means a smaller response time from the mailer.

Type=<choice>

This is the kind of mailer dealing with this network. The available settings are:

Trapdoor

Flow

UUCICO

At the moment, the only not supported setting is `Flow`. About `UUCICO`, if you use AmigaUUCP it's necessary to add the line `MungeCase N` into the '`UULIB:config file`' file, otherwise EMS could not recognize mail packets.

Configuration Example:

```
Name=fidonet
 Inbound=mail:inbound Outbound=mail:outbound Type=TrapDoor

Name=amiganet
 Inbound=mail:inbound Outbound=mail:outbound Type=flow
```

4.5 NODEs

The file we are dealing with is '`EMS:Config/Nodes`'. It contains the attributes of the systems we communicate with, such as passwords, levels, mail exchange format, etc. The keywords are:

Address=<address> *marker*

This specifies the address of the system to which the following settings are related.

Archiver=<archiver name>

This is the symbolic name of the archiver used to send mail towards this system.

PacketPWD_in=<password>
PacketPWD_out=<password>

This is used to set up a packet password. The `OUT` password will be put inside packets generated by you, the `IN` password will be expected in packets coming from that particular node.

PacketType=<type>

This is the format for the outbound packets that will be generated.

Fido_2

Fido_2+

Fido_2.2

USENET

`USENET` is not currently implemented.

AreafixName=<name>

This is the name of the Areafix robot on the remote system.

AreafixPWD_in=<password>
AreafixPWD_out=<password>

These passwords are expected/used for message to AreaFix from/for that particular node.

AreafixEchoList=<file>
AreafixFileList=<file>

These are the names of two text files, containing the tag names of the areas on the remote system. When a system requests an area you don't have, and its level is high enough, these lists will be used to locate a node who has that area, and an areafix message will be automatically generated to link it.

AreafixNotify=<boolean>

Activate this if the remote system should receive a reply for each AreaFix request.

AreafixAlert=<boolean>

Activate this if you want the remote system to know about all actions taken by your own AreaFix. Useful for remote sysop points.

GroupName=<name> *multi*
GroupLevel=<level>
GroupFlags=<flags>

With these fields you can grant a privileged level to the remote system. `GroupName` is the name of the group the system belongs to (see Section 4.6 [AREAs], page 19), `GroupLevel` is the level of that system within the group, and `GroupFlags`[3] specify what the system can do inside the group:

MSG_read The system can read messages.

MSG_write
The system can write messages.

FILE_read The system can receive files.

FILE_write
The system can send files.

Groups are useful to separate the areas belonging to different nets, and have the sysop areas at a higher level than normal ones; this way, a node which has a high level in the 'fidonet' group will have access to the fidonet sysop conferences, but not to similar areas in the 'amiganet' group. Each system can belong to any number of groups, with a different level and set of flags for each. Please note that:

```
GroupName=fidonet GroupLevel=10 GroupLevel=11 GroupFlags="MSG_read"
```

is equivalent to:

```
GroupName=fidonet GroupLevel=11 GroupFlags="MSG_read"
```

while:

```
GroupName=fidonet GroupName=amiganet
GroupLevel=11 GroupFlags="MSG_read"
```

is equivalent to:

```
GroupName=fidonet  GroupLevel=0  GroupFlags=""
GroupName=amiganet GroupLevel=11 GroupFlags="MSG_read"
```

SeenPathDim=<choice>

This field is used to filter addresses that will be listed in the `SeenBy` lines in messages exported to this system, and the way they are written too:

Any All FIDONET, UUCP and USENET addresses/formats.

Fido_2D
Fido_3D FIDONET addresses only, same zone and net.

Fido_4D FIDONET addresses only, same net.

[3] This form of protection has yet to be implemented.

Fido_5D FIDONET addresses only.

UUCP UUCP or USENET addresses only.

Flavor=<choice>

This selects the standard way to communicate with a system:

NORMAL

DIRECT

HOLD

CRASH

VIRTUAL

All non-`MAIL` messages are packed in this flavour. If a system is marked as `VIRTUAL`, no packets or flow files will ever be generated for it.

CanCrash=<boolean>

When the tosser meets a message marked as `CRASH`, it looks at this field for the system which originated the message, NOT for the one we received it from. If this field is FALSE or the system is not found in our list, the message is changed to `NORMAL`. This option is a fundamental protection against unwanted (and expensive) uses of your system as a worldwide caller.

CanCreateAreas=<boolean>

This system can create new areas automatically.

PackSmallFiles=<boolean>

It's possible to send to this system file attached to messages or TIC in a compressed way, like mail packets. See Section 4.2 [VARIABLEs], page 10.

AutoLink=<boolean>

Not implemented yet. It should be used to have every new area automatically linked to this system, when the group settings allow it.

Configuration Example:

```
Address=2:332/505@fidonet.org
 Archiver=lha
 PacketType=Fido_2_2     PacketPWD_in=aaa  PacketPWD_out=bbb
 AreafixName=echorobot   AreafixPWD_in=aaa AreafixPWD_out=bbb
 AreafixNotify=NO        AreafixAlert=NO
 SeenPathDim=Fido_5      Flavor=NORMAL
 CanCrash=NO             CanCreateAreas=YES AutoLink=NO
 GroupName=fidonet       GroupLevel=10
 GroupFlags=MSG_read
 GroupName=sysop_fidonet GroupLevel=1
 GroupFlags="MSG_read MSG_write FILE_read FILE_write"

Address=2:333/107.3@fidonet.org
 Archiver=lha        PacketType=Fido_2
 AreafixNotify=NO    AreafixAlert=NO
 SeenPathDim=Fido_2D Flavor=NORMAL
 CanCrash=NO         CanCreateAreas=YES AutoLink=NO
```

4.6 AREAs

The file we are dealing with is ‘`EMS:Config/Areas`’. It contains information about the areas in your system. These are the keywords:

Area *marker*

Used to mark the beginning of a new area definition.

Name=<name> *multi*

These are the names by which this area is known. Useful for areas spread across different domains, that takes different names on different networks.

Address=<address>

This is the main address used when processing this area.

Type=<choice>

Kind of area, one of the following:

Mail Area reserved to person-to-person messages and files.

Echo Area reserved to messages that are spread among all linked systems.

File Like ‘Echo’, but you can also post files.

Tick File-only area.

All kinds of area are implemented in the current version.

MsgPath=<path>

This is the main path used for the message database. Data related to messages are always put here.

MsgAltPath=<alternate path>

This is the secondary path for the message database: it's used to store files attached to messages. If this path is not specified files are put in `MsgPath`, unless a file database has been defined.

MsgDBname=<type>

This is the name of the kind of database to be used for messages. To find out what databases are available, you can use the script '`EMS:rexx/ListDBnames.ems`'[4].

MsgDescription=<string>

The description string for this area.

FilePath=<path>

The main path for the file database; it contains the descriptions, lengths, etc.

FileAltPath=<alternate path>

Secondary path for the file database, used to store files which are put in the database. If this path is not specified, files are stored in `FilePath`.

FileDBname=<type>

This is the name of the kind of database to be used for files. To find out what databases are available, you can use the script '`EMS:rexx/ListDBnames.ems`'[5].

FileDescription=<string>

The description string for this area.

GroupName=<name>

The group this area belongs to.

GroupLevel=<level>

The access level required for this area within its group.

MaintMode=<choice>

MaintLimit=<num>

These fields control how maintenance operations are performed:

ByNum Limits the number of messages to <num>.

ByDay Limits the number of messages by deleting those more than <num> days old.

ByNumRenum

Like ByNum, with added area renumbering.

[4] *fido_db_msg* and *quick_db_msg* in the current release.

[5] *DLG_db_file* only in the current release.

ByDayRenum

Like ByDay, with added area renumbering.

The maintenance is actually performed by the command `EMS_Database_AutoMaint`. If `MaintLimit` is set to 0, the command will not affect this area.

BBS_msg_name=<BBS name>
BBS_msg_number=<number>

Type of BBS controlling the message area, and ID #. Not Implemented Yet.

BBS_file_name=<BBS name>
BBS_file_number=<number>

Tipo del BBS che controlla l'area file e numero identificativo. Non implementato.

Language=<string>

Language. Not Implemented Yet.

IsPrivate=<boolean>

This must be activated if you want to allow private messages in a non-MAIL area.

IsLocal=<boolean>

This must be activated if you don't want an area to be ever exported.

TwoPassImport=<boolean>

Set this to activate two-pass import. The ARexx command `EMS_Tosser_Import` will import messages as usual, but will stop before re-exporting them towards other systems. The command `EMS_Tosser_PostImport` will take care of re-exporting. Useful to implement a filter or a msgtracker.

PassThrough=<boolean>

Set this if you don't want the messages and files of this area to be stored in your local message/file base.

KeepDupes=<boolean>

If this option is active, the system keeps track of meaningful data needed for duplicate checking. Active as default.

KeepStats=<boolean>

Not Implemented Yet.

NumOfDupes=<number>

If this is different from 0, it allows you to set the maximum size for the dupe-database. If it's 0 or missing, the value of the `DUPECHECKING_SIZE` variable will be used.

UnsafeTick=<boolean>

The standard behaviour of a TICK area is to check the CRC of all the imported files and to consider bad any '`.tic`' file without the CRC field. If activated, this flag allows the import of those tics that don't have any CRC data.

Configuration Example:

```
Area Name=MAIL_DIR
 Address=2:332/505@fidonet.org   Type=Mail
 MsgPath=mail:mail_dir           MsgDBname=fido_db_msg
 MaintMode=ByNum                 MaintLimit=30
 IsPrivate=YES                   IsLocal=NO
 TwoPassImport=NO PassThrough=NO KeepDupes=NO KeepStats=NO

Area Name=FirstName Name=OtherName
 Address=2:333/107@fidonet.org Type=File
 MsgPath=mail:test             MsgAltPath=mail:test_Alt
 MsgDBname=quick_db_msg        MsgDescription="...esempio..."
 GroupName=fidonet             GroupLevel=2
 MaintMode=ByDayRenum          MaintLimit=7 Language=Italiano
 BBS_msg_name=DLG              BBS_msg_number=103
 IsPrivate=YES                 IsLocal=YES
 TwoPassImport=NO              PassThrough=NO
 KeepDupes=YES                 KeepStats=NO

Area Name=NODEDIFF
 Address=2:333/107@fidonet.org   Type=Tick
 MsgPath=mail:new_msg/NODEDIFF   MsgDBname=quick_db_msg
 FilePath=mail:new_file/NODEDIFF FileDBname=DLG_db_file
 MaintMode=ByNum
 IsPrivate=NO                    IsLocal=NO
 TwoPassImport=NO                PassThrough=NO
 KeepDupes=YES                   KeepStats=NO

Area Name=BAD_MSGS
 Address=2:333/107@fidonet.org Type=Echo
 MsgPath=mail:BAD_MSGS         MsgDBname=fido_db_msg
 MaintMode=ByNum
 IsPrivate=YES                 IsLocal=YES
 TwoPassImport=NO              PassThrough=NO
 KeepDupes=NO                  KeepStats=NO
```

4.7 LINKs

The file we are dealing with is ‘EMS:Config/Links’. It contains the list of the systems attached to non-MAIL areas. These are the keywords that can appear in it:

Name=<name> *marker*

This is the name of an area. It is important to remember that an area may have more than one name, but a system always sees it with the name by which it first linked the area.

<address> *multi*

This is the address of a system that will have access to this area, to read and write files and messages. In normal situations, you should not put your own AKAs in this list: this is only necessary for particular setups, such as zonegates. Anyway, the tosser will never generate a packet addressed to itself.

Disabled=<address> *multi*

This is the address of a system that is attached to the area, but is not active at the moment and doesn't want to receive messages. The system will automatically switch to the active state when it first writes a message in this area. Useful in AreaFix operation.

Configuration Example:

```
Name=test
 2:333/107@fidonet.org 2:333/107.3@fidonet.org
 2:333/111@fidonet.org

Name=NETSYSOP.333
 Disabled=2:333/111@fidonet.org

Name=GAMES.ITA
 2:333/110@fidonet.org

Name=MAC.ITA
 2:333/107.3@fidonet.org 2:333/111@fidonet.org
 2:332/111@fidonet.org   2:331/111@fidonet.org
```

4.8 ARCHIVERs

The file we are dealing with is '`EMS:Config/Archivers`'. It includes information about the archiving programs used by the system. It is very important to correctly set the `Arc` and `Unarc` fields, otherwise you will not be able to import or export mail. These are the keywords you can put in this configuration file:

Name=<name> *marker*

This is the symbolic name associated to an archiver. All the rest of the system will reference the archiver by this name only.

Arc=<arc>

This is the command to be executed to pack mail. The tosser will automatically append the name of the archive and the name of the file to compress, in this order.

Unarc=<unarc>

This is the command to be executed to unpack mail. The tosser will automatically appnd the name of the archive to process.

ID=<id string>
IDpos=<id pos>

These two fields are used during import, to identify the kind of archiver that created a file. The identification is based on a particular sequence of characters in the first bytes of the file. `IDpos` is the number of characters to skip at the beginning of the file; `ID` is the matching sequence of bytes. If this sequence must include unprintable codes, as is the case with ARC, these should be entered in the hexadecimal form `\0xx`: for example, decimal 14 would become `\00E`.

Configuration Example:

```
Name=LHA
 Arc="Archivers:lha140 -q a" Unarc="Archivers:lha140 -q e"
 ID=-lh                        IDpos=2

Name=ZIP
 Arc=archivers:zip Unarc=Archivers:unzip
 ID=PK

Name=ZOO
 Arc="Archivers:zoo a" Unarc="Archivers:zoo x"
 ID=ZOO

Name=ARC
 ARC="archivers:arc a" UNARC="archivers:arc x"
 ID="\\01A\\008"
```

4.9 USERLIST

The file we are dealing with is ‘EMS:Config/Userlist’. It contains a list of user names, both local and external to your system, complete with address, nickname and comments. Its main purpose is to make message entering easier, but it is also used for address remapping. When a message addressed to your system or to one of your points is tossed, the userlist is scanned for the name of the addressee. If it is found, the original address of destination is replaced with the one taken from the userlist, and the message is re-exported towards its correct destination. Now remapping is safe also for users with more than one address, since the tosser picks the address which is most similar to the original (and wrong) one. These are the keywords that can be included in this file:

Name=<name> *marker*

Name of the user.

Nick=<nick name>

The nickname of the user. This can be used when entering a message: the system will replace it with the correct full name.

Note=<note text>

A short comment.

<address> *multi*

One or more addresses at which the user can be found.

Configuration Example:

```
Name="Mario Mure’"
 2:332/505@fidonet.org

Name="Maurizio Fabiani" Nick=Maui
 2:335/602@fidonet.org

Name="Robert Hofmann"
 2:2400/24@fidonet.org 2:247/737@fidonet.org
 2:2400/737@fidonet.org

Name="Magnus Thelander"
 2:255/35.18@fidonet.org 2:203/602.18@fidonet.org
```

4.10 EXTERNALs

The file we are dealing with is '`EMS:Config/Externals`'. It contains the definitions for the user menu in EMS_EDITOR, and can include these keywords:

Title=<title> *marker*

Text to be shown in the menu.

ShortCut=<short-cut key>

Shortcut key associated with this command in the menu, for quick selection.

IsCommand=<boolean>

Set this if the action associated with this menu item is an executable command; otherwise, it is assumed to be an ARexx script.

Command=<command>

This is the command or script which is executed when the menu item is selected. For scripts, you don't always need to specify a full path, since by default they are searched for in `SCRIPT_REXX_DIR`.

Among the parameters passed to the command, you can insert a few modifiers regarding the editor's status:

`%s`
`%S` Number of the current editor session.

`%#` Number of the current message.

`%a`
`%A` Name of the current area.

`%p`
`%P` Main path of the current area.

`%d`
`%D` Type of database currently displayed: can be `MSG` or `FILE`.

ScreePos=<choice>

Only useful when using EMS_EDITOR with the `CUSTOMSCREEN` option, to control the depth arrangement of the editor's screen:

DoNothing

Does absolutely nothing.

Back Moves the screen behind BEFORE executing the command.

Front Takes the editor screen in front AFTER the command has terminated the execution.

Toggle Equivalent to Back + Front.

Configuration Example:

```
Title="Edit A Msg" ShortCut=E
   Command="EditMsg.ems %s %a %#"
   ScreenPos=Toggle

Title="View Log" ShortCut=L IsCommand=TRUE
   Command="editors:ced/ed point:mail/trapdoor.log -sticky"
```

4.11 ROUTING

The file we are dealing with is ‘`EMS:Config/Routing`’. It contains the informations needed to route netmails toward the desired destinations.

Routing in EMS is different from that implemented in `GCCHost`, it has been in some way simplified and made more automatical.

Link in `GCChost` points don’t have any need of routing, because they send all their mail to their boss by default. But under strange situations points could feel the need for manual routing, so see Section 4.1 [SWITCHes], page 6.

EMS first checks whether a message is addressed to one of your points. In this case, routing is completed by sending that message to the point, as `HOLD` or with the flavor you specified for that point.

Then, `CRASH` permissions are checked for and applied to messages. Since allowing everybody to send `CRASH` messages can be painfully expensive, and controlling this priviledge through routing can make it less clear, EMS adopts a very radical position: nobody can `CRASH` messages, unless explicitly authorized to do so. In other words, only nodes listed in the ‘`EMS:Config/Nodes`’ file, with `CanCrash=TRUE` are enabled to crash messages; in all mail coming from other nodes, the `CRASH` flag is stripped and messages are degraded to `NORMAL`.

After this, EMS executes the routing commands specified in ‘`EMS:Config/Routing`’, which can be of four kinds:

Route&Send

Route

Send

Convert

All commands are executed on the same basis, i.e. only on messages whose **destination** address and flavor match those specified in the `From` and `FromFlavor` fields. Remember that they are not necessarily the same as in the original message header, since both address and flavor can change due to the execution of a command.

For example, `2:332/505.3@fidonet.org (NORMAL)` after the execution of

```
Action=Route To=2:332/500@fidonet.org ToFlavor=HOLD FromFlavor=NORMAL
   From=2:332/50?.*@fidonet.org
```

will become `HOLD` and addressed to `2:332/500@fidonet.org`. Always remember that these destination addresses and flavors (stored in the packet header) only affect the node messages will be packed for, and not the ultimate address of destination of a message.

A message for `2:332/405.3@fidonet.org (NORMAL)`, would not be affected by the above command, since the destination address doesn't match the one specified in `FromAddress`;

A message for `2:332/505.3@fidonet.org (HOLD)` would not be affected by the above command, since the flavor is different from the one specified in `FromFlavor`.

The simplest command of all is `Convert`, that modifies only the flavor of a message. `Route` and `Route&Send` can also modify the destination address.

`Route` and `Route&Send` commands are very important, because they terminate routing for a message which has been addressed to the right destination. Shouldn't we terminate routing, in such cases, the message could be altered by following commands, and be packed for an improper destination.

After the execution of all commands in the routing sequence, EMS performs two more operations. If the resulting destination system is a point, and it is not listed in '`EMS:Config/Nodes`', the message is automatically routed through his boss.

Finally, EMS checks if the resulting destination system is known and has an archiver set in '`EMS:Config/Nodes`'; in that case, the packet is compressed according to the preferences that have been set. However USENET mails are never compressed.

Back to the configuration file. the keywords that can be included in this file are:

Action=<choice> *marker*

Controls the action to perform, that could be:

`Route&Send`

`Route`

`Send`

`Convert`

To=<address>

Address to use together the commands `Route&Send` and `Route`.

ToFlavor=<choice>

The new flavor that will be used for the message, that could be:

`NORMAL`

`CRASH`

`HOLD`

`DIRECT`

From=<wild address> *multi*

A list of patterns that must be satisfied in order to execute the command in `Action` (see Chapter 3 [Address Format], page 3).

FromFlavor=<choice>

Kind of flavor that the message must own in order to execute the command. Both conditions, the one on the address and this one, should be satisfied:

`ALL`

`NORMAL`

`CRASH`

`HOLD`

`DIRECT`

Configuration Example:

```
Action=Send ToFlavor=ALL FromFlavor=CRASH
 From=*:*/*.*@fidonet.org

Action=Route&Send To=2:332/607@fidonet.org
 ToFlavor=HOLD FromFlavor=NORMAL
 From=2:332/60?.*@fidonet.org

Action=Route&Send To=2:332/500@fidonet.org
 ToFlavor=DIRECT FromFlavor=ALL
 From=2:*/400.*@fidonet.org From=2:332/503.1?@fidonet.org

Action=Route&Send To=2:332/504@fidonet.org
 ToFlavor=NORMAL FromFlavor=ALL
 From=*:*/*.*@fidonet.org
```

5 EMS_config

This chapter is still under translation. Please refer to the italian one.

6 EMS_editor

This chapter is still under translation. Please refer to the italian one.

7 ARexx Commands

All ARexx commands are included in EMS_REXX.LIBRARY, which should be opened at the beginning of each EMS script with the following instructions:

```
signal on error
signal on syntax

if( ~show( ’l’, "ems_rexx.library" ) ) then
do
   if( ~addlib( "ems_rexx.library", 0, -30, 0 ) )then
   do
      say "Could not open ems_rexx.library"
      exit 10
   end
end
```

To properly manage variables, switches and local messages, EMS stores a few data for each script that asks for its service. Unfortunately, there’s no way for EMS to know when a script ends, so it cannot automatically free the memory it allocated for that script. Therefore, you must do it manually (see Section 7.1.3 [EMS˙FreeScriptData], page 32), these instructions at the end of your script:

```
call EMS_FreeScriptData()
return 0
```

Special care should be taken for the case of a premature and unexpected script termination due to an error; your script should include this error-trapping routine:

```
error:
syntax:

error_text = EMS_LastError()

if error_text = ’’ then error_text = rc ErrorText( rc )

say ’| ***BREAK: error at’ sigl error_text

call EMS_FreeScriptData()
exit rc
```

In the command descriptions a few conventions apply:

- For all commands, the arguments and (possibly) a result are listed. When a command returns a result, you should use this syntax:

  ```
  <result> = <comando>( <argomento>, ... , <argomento> )
  ```

  or, in case you do not care at all about the result,

  ```
  CALL <comando>( <argomento>, ... , <argomento> )
  ```

- If an argument is enclosed in '[]' brackets, it is optional.
- If an argument is not enclosed by '`<>`', the only possible values for that argument are the different strings that, separated by a slash ('/'), make up its name. For example, `MSG/FILE` can be either `MSG` or `FILE`.
- If an argument includes the '`var name`' or '`stem`' specification, it means it is an indirect argument; you should not pass EMS its value, but rather the name of a variable or a stem that holds it.
- If the argument's name is related to something plural, it means you can put in several values, separated by a space. If the argument is an area name, you may also use standard AmigaDOS wildcards to select a bunch of areas at a time.

7.1 General

7.1.1 EMS_Help

Arguments: [<command name pattern>]

7.1.2 EMS_LastError

Result : <string>

7.1.3 EMS_FreeScriptData

Arguments: [ALL]

7.1.4 EMS_ShutDown

7.1.5 EMS_Version

Arguments: [<version var name>], [<registration number var name>]

7.1.6 EMS_Cfg_Read

Arguments: <part name>

7.1.7 EMS_Cfg_Write

Arguments: <part name>, [FORCE]

7.1.8 EMS_Log_Close

7.1.9 EMS_Log_Level

Arguments: [<new level>]
Result : <old level>

7.1.10 EMS_Log_Print

Arguments: <level>, <string to print>, [TIME]

7.1.11 EMS_FullAddress

Arguments: <area name>, <partial address>
Result : <result>

7.1.12 EMS_Address_IsFIDO

Arguments: <address>
Result : <result>

7.1.13 EMS_Address_IsPOINT

Arguments: <address>
Result : <result>

7.1.14 EMS_Address_IsUUCP

Arguments: <address>
Result : <result>

7.1.15 EMS_Address_IsUSENET

Arguments: <address>
Result : <result>

7.1.16 EMS_Address_GetDomain

Arguments: <address>
Result : <domain>

7.1.17 EMS_Address_Compare

Arguments: <address>, <address to compare>
Result : <result> [FALSE|TRUE]

7.1.18 EMS_Address_Compare_Wild

Arguments: <address>, <wild address to compare>
Result : <result> [FALSE|TRUE]

7.2 Lists

7.2.1 EMS_Akas

Arguments: <stem name>, [<domain filter>]

7.2.2 EMS_Domains

Arguments: <stem name>

7.2.3 EMS_Nodes

Arguments: <stem name>

7.2.4 EMS_Areas

Arguments: <stem name>, [SORT]

7.2.5 EMS_Areas_Imported

Arguments: <stem name>, [SORT]

7.2.6 EMS_Areas_New

Arguments: <stem name>, [SORT]

7.2.7 EMS_Archivers

Arguments: <stem name>

7.3 Variable Information

7.3.1 EMS_Var_Local

Arguments: <var name>, [<new value>]
Result : <value>

7.3.2 EMS_Var_Global

Arguments: <var name>, [<new value>]
Result : <value>

7.3.3 EMS_Var_Delete

Arguments: <var name>

7.4 Switch Manipulation

7.4.1 EMS_Switch_Local

Arguments: <switch name>, [<new status>]
Result : <status>

7.4.2 EMS_Switch_Global

Arguments: <switch name>, [<new status>]
Result : <status>

7.4.3 EMS_Switch_Delete

Arguments: <switch name>

7.5 Node Information

7.5.1 EMS_Node_Archiver

Arguments: <node address>, [<new name>]
Result : <name>

7.5.2 EMS_Node_Packet_Pwd_In

Arguments: <node address>, [<new password>]
Result : <password>

7.5.3 EMS_Node_Packet_Pwd_Out

Arguments: <node address>, [<new password>]
Result : <password>

7.5.4 EMS_Node_Packet_Type

Arguments: <node address>, [<new choice>]
Result : <choice>

7.5.5 EMS_Node_Areafix_Name

Arguments: <node address>, [<new name>]
Result : <name>

7.5.6 EMS_Node_Areafix_EchoList

Arguments: <node address>, [<new file>]
Result : <file>

7.5.7 EMS_Node_Areafix_FileList

Arguments: <node address>, [<new file>]
Result : <file>

7.5.8 EMS_Node_Areafix_Pwd_In

Arguments: <node address>, [<new password>]
Result : <password>

7.5.9 EMS_Node_Areafix_Pwd_Out

Arguments: <node address>, [<new password>]
Result : <password>

7.5.10 EMS_Node_Areafix_Notify

Arguments: <node address>, [<new bool>]
Result : <bool>

7.5.11 EMS_Node_Areafix_Alert

Arguments: <node address>, [<new bool>]
Result : <bool>

7.5.12 EMS_Node_Addressing

Arguments: <node address>, [<new choice>]
Result : <choice>

7.5.13 EMS_Node_Flavor

Arguments: <node address>, [<new choice>]
Result : <choice>

7.5.14 EMS_Node_CanCrash

Arguments: <node address>, [<new bool>]
Result : <bool>

7.5.15 EMS_Node_CanCreateAreas

Arguments: <node address>, [<new bool>]
Result : <bool>

7.5.16 EMS_Node_PackSmallFiles

Arguments: <node address>, [<new bool>]
Result : <bool>

7.5.17 EMS_Node_AutoLink

Arguments: <node address>, [<new bool>]
Result : <bool>

7.5.18 EMS_Node_Groups_Get

Arguments: <node address>, <stem name>

7.5.19 EMS_Node_Groups_Set

Arguments: <node address>, <stem name>

7.5.20 EMS_Node_Delete

Arguments: <node address>

7.6 Area Information

7.6.1 EMS_Area_Address

Arguments: <area name>, [<new address>]
Result : <address>

7.6.2 EMS_Area_Type

Arguments: <area name>, [<new choice>]
Result : <choice>

7.6.3 EMS_Area_Msg_Path

Arguments: <area name>, [<new path>]
Result : <path>

7.6.4 EMS_Area_Msg_AltPath

Arguments: <area name>, [<new path>]
Result : <path>

7.6.5 EMS_Area_Msg_DBname

Arguments: <area name>, [<new name>]
Result : <name>

7.6.6 EMS_Area_Msg_Description

Arguments: <area name>, [<new string>]
Result : <string>

7.6.7 EMS_Area_File_Path

Arguments: <area name>, [<new path>]
Result : <path>

7.6.8 EMS_Area_File_AltPath

Arguments: <area name>, [<new path>]
Result : <path>

7.6.9 EMS_Area_File_DBname

Arguments: <area name>, [<new name>]
Result : <name>

7.6.10 EMS_Area_File_Description

Arguments: <area name>, [<new string>]
Result : <string>

7.6.11 EMS_Area_Group_Name

Arguments: <area name>, [<new name>]
Result : <name>

7.6.12 EMS_Area_Group_Level

Arguments: <area name>, [<new level>]
Result : <level>

7.6.13 EMS_Area_Maint_Mode

Arguments: <area name>, [<new number>]
Result : <number>

7.6.14 EMS_Area_Maint_Limit

Arguments: <area name>, [<new choice>]
Result : <choice>

7.6.15 EMS_Area_BBS_Msg_Name

Arguments: <area name>, [<new name>]
Result : <name>

7.6.16 EMS_Area_BBS_Msg_Number

Arguments: <area name>, [<new number>]
Result : <number>

7.6.17 EMS_Area_BBS_File_Name

Arguments: <area name>, [<new name>]
Result : <name>

7.6.18 EMS_Area_BBS_File_Number

Arguments: <area name>, [<new number>]
Result : <number>

7.6.19 EMS_Area_Language

Arguments: <area name>, [<new string>]
Result : <string>

7.6.20 EMS_Area_IsPrivate

Arguments: <area name>, [<new bool>]
Result : <bool>

7.6.21 EMS_Area_IsLocal

Arguments: <area name>, [<new bool>]
Result : <bool>

7.6.22 EMS_Area_TwoPassImport

Arguments: <area name>, [<new bool>]
Result : <bool>

7.6.23 EMS_Area_PassThrough

Arguments: <area name>, [<new bool>]
Result : <bool>

7.6.24 EMS_Area_KeepDupes

Arguments: <area name>, [<new bool>]
Result : <bool>

7.6.25 EMS_Area_KeepStats

Arguments: <area name>, [<new bool>]
Result : <bool>

7.6.26 EMS_Area_UnsafeTick

Arguments: <area name>, [<new bool>]
Result : <bool>

7.6.27 EMS_Area_NumOfDupes

Arguments: <area name>, [<new number>]
Result : <number>

7.6.28 EMS_Area_Create

Arguments: <area name>, <type>

7.6.29 EMS_Area_Delete

Arguments: <area name>, [EVENDB]

7.6.30 EMS_Area_NameFromPath

Arguments: <path>
Result : <area name>

7.6.31 EMS_Area_NameFromNumber

Arguments: <BBS number>, MSG|FILE
Result : <area name>

7.6.32 EMS_Area_FullFileName

Arguments: <file name>
Result : <area name>, <file name>

7.6.33 EMS_Area_HiMark

Arguments: <area name>, [<new hwm>]
Result : <old hwm>

7.7 Database Information

7.7.1 EMS_Area_Item_List

Arguments: <area name>, MSG|FILE, <stem name>

7.7.2 EMS_Area_Item_List_Marked

Arguments: <area name>, MSG|FILE, <stem name>

7.7.3 EMS_Area_Item_List_Flagged

Arguments: <area name>, MSG|FILE, <stem name>, <flag names>

7.7.4 EMS_Area_Item_Mark

Arguments: <area name>, MSG|FILE, <number>, [<new status>]
Result : <status>

7.7.5 EMS_Area_Item_Flag

Arguments: <area name>, MSG|FILE, <number>, <flag name>, [<new status>]
Result : <status>

7.7.6 EMS_Area_Item_Delete

Arguments: <area name>, MSG|FILE, <number>

7.7.7 EMS_Area_Item_Delete_Marked

Arguments: <area name>, MSG|FILE

7.7.8 EMS_Area_Item_Header_From

Arguments: <area name>, MSG|FILE, <number>, [<new value>]
Result : <value>

7.7.9 EMS_Area_Item_Header_To

Arguments: <area name>, MSG|FILE, <number>, [<new value>]
Result : <value>

7.7.10 EMS_Area_Item_Header_Subject

Arguments: <area name>, MSG|FILE, <number>, [<new value>]
Result : <value>

7.7.11 EMS_Area_Item_Header_AttachedFile

Arguments: <area name>, MSG|FILE, <number>, [<new value>]
Result : <value>

7.7.12 EMS_Area_Item_Header_Date_Created

Arguments: <area name>, MSG|FILE, <number>, [<new value>]
Result : <value>

7.7.13 EMS_Area_Item_Header_Date_Received

Arguments: <area name>, MSG|FILE, <number>, [<new value>]
Result : <value>

7.7.14 EMS_Area_Item_Header_Link_Down

Arguments: <area name>, MSG|FILE, <number>
Result : <value>

7.7.15 EMS_Area_Item_Header_Link_Up

Arguments: <area name>, MSG|FILE, <number>
Result : <value>

7.7.16 EMS_Area_Item_Header_Flag

Arguments: <area name>, MSG|FILE, <number>, <flag name>, [<new status>]
Result : <status>

7.8 Checking for Existence

7.8.1 EMS_Aka_Check

Arguments: <address>
Result : <bool>

7.8.2 EMS_Aka_Nearer

Arguments: <address>
Result : <aka address>

7.8.3 EMS_Node_Check

Arguments: <node address>
Result : <bool>

7.8.4 EMS_Area_Check

Arguments: <area name>
Result : <bool>

7.9 Tosser Control

7.9.1 EMS_Tosser_Import

Arguments: [<domain to import>]

7.9.2 EMS_Tosser_PostImport

Arguments: [<areas to export>]

7.9.3 EMS_Tosser_Export

Arguments: [<areas to export>]

7.9.4 EMS_Tosser_Rescue

Arguments: <areas to rescue>, <address to rescue for>, <mode>

7.9.5 EMS_Tosser_Rescan

Arguments: <areas to rescan>, <address to rescan for>, <mode>, [<from user>], [<to user>], [<subject>], [<num of days back>]

7.9.6 EMS_Tosser_TestRouting

Arguments: <from address>, <to address var name>, <flavor var name>

7.10 Database Control

7.10.1 EMS_Database_Caches_Update

Arguments: <areas>, MSG|FILE

7.10.2 EMS_Database_Caches_Load

Arguments: <areas>, MSG|FILE

7.10.3 EMS_Database_Caches_Free

Arguments: <areas>, MSG|FILE

7.10.4 EMS_Database_Maint_Internal

Arguments: <areas>, MSG|FILE

7.10.5 EMS_Database_AutoMaint

Arguments: <areas>, MSG|FILE

7.10.6 EMS_Database_Renum

Arguments: <areas>, MSG|FILE

7.10.7 EMS_Database_Delete_By_Num

Arguments: <areas>, MSG|FILE, <max number of messages>

7.10.8 EMS_Database_Delete_By_Day

Arguments: <areas>, MSG|FILE, <how many days back>

7.10.9 EMS_Database_Search

Arguments: <area> , MSG|FILE, <stem name>, <pattern>

7.10.10 EMS_Database_Names

Arguments: <msg names stem>, <file names stem>

7.11 Userlist Handling

7.11.1 EMS_User_List

Arguments: <user stem name>

7.11.2 EMS_User_Add

Arguments: <user name>

7.11.3 EMS_User_Delete

Arguments: <user name>

7.11.4 EMS_User_Address_List

Arguments: <user name>, <address stem name>

7.11.5 EMS_User_Address_Add

Arguments: <user name>, <address>

7.11.6 EMS_User_Address_Delete

Arguments: <user name>, <address>

7.11.7 EMS_User_Comment

Arguments: <user name>, [<new value>]
Result : <value>

7.11.8 EMS_User_Nick

Arguments: <user name>, [<new nick name>]
Result : <nick name>

7.11.9 EMS_User_NameFromAddress

Arguments: <address>
Result : <user name>

7.12 Flow File Handling

7.12.1 EMS_Flow_Get

Arguments: <domain>, <filenames stem>, <addresses stem>, <kinds stem>

7.12.2 EMS_Flow_Check

Arguments: <domain>, <filename>, <address>, [<kind>]
Result : <result> [OK|NO]

7.12.3 EMS_Flow_Add

Arguments: <domain>, <filename>, <address>, <kind>, <mode>

7.12.4 EMS_Flow_Remove

Arguments: <domain>, <filename>, <address>, <kind>

7.13 Message-File Handling

7.13.1 EMS_Item_Alloc

Arguments: <msg id>, <area name>, MSG|FILE, [<number>]

7.13.2 EMS_Item_Free

Arguments: <msg id>

7.13.3 EMS_Item_Write

Arguments: <msg id>

7.13.4 EMS_Item_Number

Arguments: <msg id>, [<new number>]
Result : <number>

7.13.5 EMS_Item_Area

Arguments: <msg id>, [<new area name>]
Result : <area name>

7.13.6 EMS_Item_Lines_Count

Arguments: <msg id>, TEXT|HEADER
Result : <num of lines>

7.13.7 EMS_Item_Lines_Get

Arguments: <msg id>, TEXT|HEADER, <text stem name>

7.13.8 EMS_Item_Lines_Set

Arguments: <msg id>, TEXT|HEADER, <text stem name>

7.13.9 EMS_Item_Lines_Delete

Arguments: <msg id>, TEXT|HEADER

7.13.10 EMS_Item_Line

Arguments: <msg id>, TEXT|HEADER, <line number>, [<new line text>]
Result : <line text>

7.13.11 EMS_Item_Line_Delete

Arguments: <msg id>, TEXT|HEADER, <line number>

7.13.12 EMS_Item_Line_Add_Head

Arguments: <msg id>, TEXT|HEADER, <new line text>

7.13.13 EMS_Item_Line_Add_Tail

Arguments: <msg id>, TEXT|HEADER, <new line text>

7.13.14 EMS_Item_Line_Unique

Arguments: <msg id>, TEXT|HEADER, <text to match>, <new line text>

7.13.15 EMS_Item_Lines_Format

Arguments: <msg id>, [<quote text>], [<quote width>], [<quote options>]

7.13.16 EMS_Item_File_Read

Arguments: <msg id>, TEXT|HEADER, <file name>, [APPEND]

7.13.17 EMS_Item_File_Write

Arguments: <msg id>, TEXT|HEADER, <file name>, [APPEND]

7.13.18 EMS_Item_SeenBy_Get

Arguments: <msg id>, <address stem name>

7.13.19 EMS_Item_SeenBy_Set

Arguments: <msg id>, <address stem name>

7.13.20 EMS_Item_Path_Get

Arguments: <msg id>, <address stem name>

7.13.21 EMS_Item_Path_Set

Arguments: <msg id>, <address stem name>

7.13.22 EMS_Item_Header_From

Arguments: <msg id>, [<new value>]
Result : <value>

7.13.23 EMS_Item_Header_To

Arguments: <msg id>, [<new value>]
Result : <value>

7.13.24 EMS_Item_Header_Subject

Arguments: <msg id>, [<new value>]
Result : <value>

7.13.25 EMS_Item_Header_AttachedFile

Arguments: <msg id>, [<new value>]
Result : <value>

7.13.26 EMS_Item_Header_Date_Created

Arguments: <msg id>, [<new value>]
Result : <value>

7.13.27 EMS_Item_Header_Date_Received

Arguments: <msg id>, [<new value>]
Result : <value>

7.13.28 EMS_Item_Header_Address_From

Arguments: <msg id>, [<new value>]
Result : <value>

7.13.29 EMS_Item_Header_Address_To

Arguments: <msg id>, [<new value>]
Result : <value>

7.13.30 EMS_Item_Header_Link_Down

Arguments: <msg id>
Result : <value>

7.13.31 EMS_Item_Header_Link_Up

Arguments: <msg id>
Result : <value>

7.13.32 EMS_Item_Header_Flag

Arguments: <msg id>, <flag name>, [<new status>]
Result : <status>

7.13.33 EMS_Item_Header_Edit

Arguments: <msg id>, <requester name>, <editor command + arguments>
Result : <result>

7.14 Areafix

7.14.1 EMS_Areafix_Activate

Arguments: <node address>, [<area name>]

7.14.2 EMS_Areafix_Passivate

Arguments: <node address>, [<area name>]

7.14.3 EMS_Areafix_Node_Add

Arguments: <node address>, <area name>

7.14.4 EMS_Areafix_Node_Delete

Arguments: <node address>, <area name>

7.14.5 EMS_Areafix_Node_Links_Get

Arguments: <node address>, [<active areas stem name>], [<passive areas stem name>]

7.14.6 EMS_Areafix_Node_Links_Set

Arguments: <node address>, [<active areas stem name>], [<passive areas stem name>]

7.14.7 EMS_Areafix_Node_IsLinked

Arguments: <node address>, <area name>
Result : <result>

7.14.8 EMS_Areafix_Node_MatchGroup

Arguments: <node address>, <area name>
Result : <bool>

7.14.9 EMS_Areafix_Area_Add

Arguments: <area name>, <node address>

7.14.10 EMS_Areafix_Area_Delete

Arguments: <area name>, <node address>

7.14.11 EMS_Areafix_Area_Links_Get

Arguments: <area name>, [<active nodes stem name>], [<passive nodes stem name>]

7.14.12 EMS_Areafix_Area_Links_Set

Arguments: <area name>, [<active nodes stem name>], [<passive nodes stem name>]

7.15 Custom Configurations

7.15.1 EMS_CustomCfg_Get

Arguments: <cfg name>, <item>, <item data label>, <data stem>
Result : <result>

7.15.2 EMS_CustomCfg_Set

Arguments: <cfg name>, <item>, <item data label>, <data stem>
Result : <result>

7.15.3 EMS_CustomCfg_Add

Arguments: <cfg name>, <item>

7.15.4 EMS_CustomCfg_Delete

Arguments: <cfg name>, <item>

7.15.5 EMS_CustomCfg_Free

Arguments: <cfg name>

7.15.6 EMS_CustomCfg_Flush

Arguments: <cfg name>

7.15.7 EMS_CustomCfg_Search

Arguments: <cfg name>, <search pattern>, <result stem>, [REVERSE]

7.16 Requesters

7.16.1 EMS_String_Select

Arguments: <title>, <text var>
Result : <result>

7.16.2 EMS_File_Select

Arguments: <title>, <file var>
Result : <result>

7.16.3 EMS_Area_Select

Arguments: <area name var>
Result : <result>

7.16.4 EMS_User_Select

Arguments: <user name var>, [<address var>]
Result : <result>

7.16.5 EMS_Do_Choice_Single

Arguments: <title>, <labels stem>, <active label var>
Result : <result> [OK|CANCEL]

7.16.6 EMS_Do_Choice_Multi

Arguments: <title>, <labels stem>, <labels' status stem>
Result : <result> [OK|CANCEL]

7.16.7 EMS_Do_Request

Arguments: <text>, <buttons>
Result : <selected button>

7.17 Stem Manipulation

7.17.1 EMS_Search_In_Stem

Arguments: <stem name>, <search pattern>, [<last match>]
Result : <next match>

7.17.2 EMS_Add_To_Stem

Arguments: <stem name>, <new item>

7.17.3 EMS_Translate_Stem

Arguments: <format>, <modifiers stem>, <data stem>
Result : <result>

7.17.4 EMS_Sort_Stem

Arguments: <stem name>, [ADDRESS]

7.17.5 EMS_Calc_Stem_Width

Arguments: <stem name>, [ADAPT|ADAPT_RIGHT]
Result : <width>

8 Scripts

This chapter is still under translation. Please refer to the italian one.

8.1 ShutDown.ems

8.2 Help.ems

Arguments: [<cmd pattern>]

8.3 General.ems

Arguments: <command> [<argument> [',' <argument> ...]]

8.4 ListDBnames.ems

8.5 Automaint.ems

Arguments: [<areas pattern>]

8.6 MaintInternal.ems

Arguments: [<areas pattern>]

8.7 TestRouting.ems

Arguments: <from> <to> <flavor>

8.8 ExtractFile.ems

Arguments: <area name> <msg num>

8.9 MaintFiles.ems

Arguments: <area pattern> <option>

8.10 Export.ems

Arguments: [<areas pattern>]

8.11 Import.ems

Arguments: <mode> <domain>

8.12 Count.ems

8.13 AreaStat.ems

Arguments: <session number> <area name>

8.14 Multisend.ems

Arguments: <kind> <file> [<file> ...]

8.15 AreaFix.ems

8.16 LocalAreaFix.ems

8.17 TickAction.ems

Arguments: [<option> ...]

8.18 DMreply.ems

Arguments: <session number> <area name> <msg num> <cmd> [<cmd_args> ...]

Appendix A How To Register

EMS is a shareware program and in order to use it easily you should register, sending a fee. There are different fees, depending on your needs:

```
Only Point: 45.000 Lire =  50 DM = 30 US$ = 20 UK
Even  Node: 90.000 Lire = 100 DM = 65 US$ = 45 UK
```

If you register as a point you can always upgrade to the complete version, simply paying the difference. But if you were registered to GCChost v3.x, you can do the upgrade to EMS until 31-12-93 at these prices:

```
From Node  to Node : 50.000 Lire =  55 DM = 35 US$ = 25 UK
From Point to Node : 60.000 Lire =  65 DM = 40 US$ = 30 UK
From Point to Point: 20.000 Lire =  20 DM = 15 US$ = 10 UK
```

If you live in Italy, you can register directly from the author, filling up the Registration Form that you will find in the archive file of EMS and sending it through e-mail to **Davide Massarenti**, at 2:332/505.3@fidonet.org or 39:102/501.3@amiganet.ftn.

The author's snail-mail address is:

```
                    Davide Massarenti
                   Via Mascherella, 11
                  41100 Modena (MO) Italy
```

You can use an eurocheque or the POSTAL GIRO ACCOUNT n. 12106415 (always Davide Massarenti). For normal cheque, please add 10.000 Lire as a contribution for the bank-transaction.

If you like outside Italy, you can concact through e-mail the following supporter, who can give you more information:

```
Italy:

 Mario Mure'      2:332/505@fidonet.org     |+39-59-226454
                  2:332/2@fidonet.org       |
                  39:102/501@amiganet.ftn   |
```

```
Denmark:

 Adam Sjoegren           2:230/149@fidonet.org     |+45-31354775
                         39:141/109@amiganet.ftn   |

 Anders Wegge Jakobsen   2:230/821@fidonet.org     |+45-86-755253
 (Just support)
```

```
Germany:

 Robert Hofmann   2:2400/24@fidonet.org     |
                  39:171/101@amiganet.ftn   | Zyxel 19.2
                  16:100/9024@zyxelnet.ftn  |
                  47:52/23@fisnet.ftn       |+49-911-429747

                  2:2400/924@fidonet.org    |
                  39:171/102@amiganet.ftn   | ISDNB,C
                  16:100/9025@zyxelnet.ftn  |
                  47:52/123@fisnet.ftn      |+49-911-9941681

                  Robert_Hofmann@f24.n2400.z2.wnbbs.nbg.sub.org
```

```
Holland:

 Stephan Wijman   2:2802/123@fidonet.org    |+31-40-480556
```

```
Sweden

 Magnus Thelander 2:203/603.18@fidonet.org
                  2:2400/24.118@fidonet.org

                  Magnus_Thelander@p18.sandy.bbs.bad.se
```

```
United Kingdom:

 Phil Clifford    2:253/404@fidonet.org     |
                  39:138/1@amiganet.ftn     |+44-246-236510
                  39:13/1@amiganet.ftn      | NOT 24hrs

                  pan@arkadia.wizdom.royle.org

 Steve Dockar     2:250/320@fidonet.org     |+44-532-391100
                  39:138/14@amiganet.ftn    | 24hrs

                  dox@dabbs.adsp.sub.org too
```

```
United States:

 Bill Peck        1:321/242@fidonet.org     |+1-413-448-8367
```

Appendix B Error codes

In EMS the error codes are hierarchically organized as two levels: the first is the one that is written in the `rc` variable by ARexx command, and can be:

```
EMS_ERR_NOMEM       rc = 61
EMS_ERR_IO          rc = 62
EMS_ERR_OBJ         rc = 63
EMS_ERR_FILESYSTEM  rc = 64
EMS_ERR_DATABASE    rc = 65
EMS_ERR_ADDRESS     rc = 66
EMS_ERR_TOSSER      rc = 67
```

the other level can be found calling the `EMS_LastError()` command, whose result is a string describing the error. This string has the following format:

```
<primary error>:<secondary error>:...
```

Here there are the value for the `<secondary error>`:

EMS_ERR_IO

EMS_FILE_ERR_IOERR	1
EMS_FILE_ERR_LONGLINE	2
EMS_FILE_ERR_ENDCHAR	3
EMS_FILE_ERR_SRTPKT	4
EMS_FILE_ERR_CANTOPEN_IN	5
EMS_FILE_ERR_CANTOPEN_OUT	6

EMS_ERR_OBJ

EMS_OBJ_ERR_NOT_SAME_FATHER	1
EMS_OBJ_ERR_NO_FILE	2
EMS_OBJ_ERR_BUFFER_TOO_SMALL	3

EMS_ERR_FILESYSTEM

EMS_FS_ERR_CANTCD	1
EMS_FS_ERR_NOTADIR	2
EMS_FS_ERR_CREATEDIR	3

EMS_ERR_DATABASE

EMS_DB_ERR_HEADER	1
EMS_DB_ERR_BODY	2

EMS_DB_ERR_WRONGSEENBY	3
EMS_DB_ERR_WRONGPATH	4
EMS_DB_ERR_WRONGMSGNUMBER	5
EMS_DB_ERR_DONTEXIST	6
EMS_DB_ERR_CANTCREATE	7

EMS_ERR_ADDRESS

EMS_AD_ERR_NO_ADDRESS	1
EMS_AD_ERR_WRONG_FIDO	2
EMS_AD_ERR_WRONG_UUCP	3

EMS_ERR_TOSSER

EMS_TSR_ERR_WRONG_PACKET_TYPE	1
EMS_TSR_ERR_WRONG_MSG_HEADER	2
EMS_TSR_ERR_WRONG_PASSWORD	3
EMS_TSR_ERR_NO_ORIGIN	4
EMS_TSR_ERR_UNLISTED_NODE	5
EMS_TSR_ERR_HACKER_PKT	6
EMS_TSR_ERR_NO_ARCHIVER	7
EMS_TSR_ERR_CANTARC	8
EMS_TSR_ERR_CANTUNARC	9
EMS_TSR_ERR_NO_DOMAIN	10
EMS_TSR_ERR_NO_AREA	11
EMS_TSR_ERR_CANTLOADCONFIG	12
EMS_TSR_ERR_DUPLICATE	13
EMS_TSR_ERR_WRONG_CRC	14
EMS_TSR_ERR_NO_DATABASE_MSG	15
EMS_TSR_ERR_NO_DATABASE_FILE	16
EMS_TSR_ERR_NO_ARCHIVER_NODE	17

Index

Summary of Switches

Summary of Variables

Summary of Configuration Keywords

Summary of ARexx Commands

Table of Contents