No.1

"Fortalecimiento del Programa Nacional de Monitoreo Atmosférico: Extensión de la cobertura del Sistema Nacional de Calidad del Aire (SINAICA)"

Final Report

Matías Software Group S.A. de. C.V.

November 2004

Table of Contents

Final Report

***Preamble*.**

By the end of 2002, Matias Software Group developed a working prototype to demonstrate the possibility of automating centralized air quality monitoring data acquisition, leveraging on existing TCP/IP technologies to integrate Mexico City, Guadalajara and Toluca's data into a single federal database through an ISP uplink. The National Air Quality Information System (SINAICA) was born.

Nevertheless, changes in the network's data acquisition systems, increased expectations from the users, and a general desire for upgrade and usability, motivated further modifications and extensions to the initial prototype, thus, by the second semester 2003 a second version of SINAICA was released, basically a new and improved rewrite of the system with several improvements especially in the data link and display modules.

At the same time, new monitoring networks were readying throughout the country, so Puebla's network was added, totaling four integrated networks to SINAICA, and preliminary diagnostics and analysis were made to accommodate for more networks, with Monterrey and Salamanca's foreseen for the near future.

General Considerations

Taking SINAICA's second version as a starting point, a diagnostic revealed some adjustments were necessary to further extensions, as well as several areas of improvement in the query and display subsystems.

These elements were taken into account and integrated into the new contract's terms of reference, freely translated below:

> "... Incorporation of two networks: Monterrey and Salamanca. Since the former has gone through a modernization process in both measuring and analysis equipment as well as in data acquisition systems, a new dedicated data extraction module was required. Such module will have to interface to

the SQL RDBMS that such system uses. As for the latter, even though recent samples from its database suggest a simple storage format, it will nevertheless require the specific extraction module. This network will begin concentrating data to its control center, located at Guanajuato's government Ecology Institute, in Guanajuato, Guanajuato, and relay from there to SINAICA.
Also identified as susceptible of integration to SINAICA are the networks in Irapuato, Celaya, Ciudad Juarez, Tijuana, Mexicali, Aguascalientes, Torreón, Zacatecas and San Luis Potosi"
"At the same time, CENICA's experimental network, with its two reference stations in Mexico City should be integrated."
"Identify necessary preconditions and propose solutions for integration of manual monitoring networks"
"A number of possible improvements to the second version of the query application have been stated, as well as new query modules tailored for the special needs of scientists, executive officers and the general public."

Activities

The terms of reference identify nine activities which in turn result in twelve deliverable products.

For this report they have been correlated in the table below, with action number in the left, description in the center field and deliverable product number in the right. Where there has been no associated product with an action due to its generic nature or because such action is a design prerequisite for another, the abbreviation 'GEN' is used.

# Act	Description	# Prod
1	Make an analysis of both manual and automatic air quality monitoring networks in order to establish the specific protocols used for data extraction and transmission to SINAICA.	1
2	Integrate to SINAICA monitoring networks from Monterrey, Salamanca, Irapuato, Celaya, Cd. Juárez, Tijuana, Mexicali, CENICA plus those which at the end of the contract may be added.	5
3	Design and program a data extraction and transmission system, unifying the network control centers requirements.	2,7,8

# Act	Description	# Prod
4	Program an administration module tho allow for the extraction and transmission system monitoring, both for in site (CENICA) and remote operation.	2
5	The provider should insure the best Internet access for the first year, including the link between INE and CENICA. It will be the network's responsibility to facilitate this infrastructure installation.	GEN
6	The provider should incorporate encrypted data transmission ability to the system via use of X509 certificates, and should also provide the adequate tools for certificate generation and management.	11
7	Train INE/CENICA personnel in the installation, operation and adequation of the delivered applications for control centers, stations and database systems.	GEN
8	Program applications for manual network data handling, as well as for CENICA's experimental center data (XRF, Balloons, VOC, etc.) for their integration into SINAICA. Detailed description included in **Appendix 1.**	6
9	Modify storage and presentation subsystems in order to include the following characteristics, preferably in separate modules: Real time query Specialized queries Local network validated data Historical data, federal validated data	9

Deliverable Products

The contract terms of reference identify and describe twelve deliverable products, these descriptions are freely translated ahead, followed by a brief recount of the attained results as well as a reference to the pertaining appendix where needed.

Since many of these products have materialized as software developments, they have been installed according to their nature in the new SINAICA database server, in the monitoring networks equipment, in CENICA's equipment or in the data extraction and collection module. Additionally, a companion CD compilation of all software is provided with this report.

1. Document data type, formats and storage technology used at all the automatic monitoring networks.

To comply with this task, a number of visits to chosen networks (those whose formats were still not fully documented) were planned and made. These visits are summarized in the following table. **Appendix 1** describes the results of the diagnostic as well as data and file types and formats in full detail.

Network	*Activity Date*	*Activity Description*
CJU	Apr 13	Telephone Conference
	Apr 29 to May 1	Diagnostic
	Jul 22	Telephone Conference
	Jul 23 to Jul 25	Integration to new SINAICA
GDL	Aug 12	Integration to new SINAICA
MTY	Mar 31	Diagnostic
	Jun 17	System Test
	Jul 26	Integration to new SINAICA
MXC	Jun 01	Diagnostic
		Integration to new SINAICA
PUE	Aug 30	Integration to new SINAICA
ROS	Jun 01	Diagnostic
		Integration to new SINAICA
BAJIO/GTO	Apr 14	Visit to control center GTO
BAJIO/LEO	Apr. 15	Diagnostic
BAJIO/IRA	Apr 15	Diagnostic
BAJIO/SAL	Apr 16	Diagnostic
		Integration to new SINAICA
TIJ	Jun 23	Diagnostic
	Jun 27	System Test
		Integration to new SINAICA

Extensions of SINAICA

Network	*Activity Date*	*Activity Description*
TKT	Jun 01	Diagnostic Integration to new SINAICA
TLC	Aug 15	Diagnostic Integration to new SINAICA

2. Setup application for the extraction module improvements.

Since the beginning of the project and according to general considerations and especially activity number 3, an autonomous extraction, collection and transmission system was chosen.

Full details on the system and its mechanics are stated in **Appendix 2** .

3. Full source code of all modifications and new developments, list of all programmed applications, libraries used for development. In all cases the provider should grant INE/CENICA a non exclusive license allowing for use as well as unlimited distribution, even commercial distribution of all applications.

The companion CD includes all source code.

A full list and relation of all programmed applications and libraries used is described in **Appendix 3** .

Given the nature of the project, the chosen platform and the cooperative effort that SINAICA was meant to be since its beginnings, the Free Software Foundation's General Public License (FSF-GPL) was chosen.

Appendix 4 includes the full text of such license.

4. A report documenting installation and operation procedures for all programs of the third version of the system.

See **Appendix 3.**

5. Installation of 14 extraction and collection systems.

As it is documented in **Appendix 1** , not all cities stated were susceptible of integration, either because they didn't have a control center or in some cases because there was no network at all. That meant devising alternate strategies to integrate those cities, which finally allowed the full integration of: Guadalajara, Ciudad Juárez, Mexicali, Ciudad de México, Monterrey, Puebla, Rosarito, Salamanca, Tecate, Tijuana, Toluca and CENICA's experimental network.

The dates in which the networks were integrated to SINAICA are stated in the table from point 1.

In particular, several problems with Guanajuato's communication infrastructure impeded the timely integration of Celaya and Irapuato networks, which required modifications to the original project goals. Nevertheless the system is ready for their integration and the corresponding extraction equipment was given to CENICA.

6. Data management modules for: a) manual air quality monitoring networks, b) Monitoring and analysis equipment at CENICA: continuous VOC monitoring, automatic stations, Carbon analyzer and pilot balloons.

The two automatic stations the CENICA operates were integrated directly to SINAICA as well as their historic database.

Data from station "Revolución" is not yet available to the general public due to revisions being made by CENICA personnel.

The companion CD includes three of the four programmed modules: Pilot Balloon application, XRF module and Carbon analyzer module. The VOC module was developed as a set of Excell macros which are already installed in the corresponding equipment.

7. SQL data extraction module, both Linux and Windows version.

This module, which is used by Monterrey network, was integrated to the data extraction and collection system detailed in Appendix 2. It has been operating since July 26.

8. Datalogger data extraction module.

This module, originally designed for CENICA's dataloggers, was also integrated to the data extraction and collection system and is used in all Baja California's networks (Tijuana, Rosarito, Tecate and Mexicali).

9. Real time query and report modules, specialized query modules incorporating GIS technology to display maps.

Every query and report module has been installed in CENICA's new server and are available online at http://sinaica2.ine.gob.mx/ besides being included in the companion CD.

10. Automatic data validation extensions to the extractor and storage modules.

These extensiolns were included partly in the data extraction and collection system and partly in the storage subsystem at CENICA's new server, as well as being fully included in the companion CD.

11. X509 Certificate generation and administration system.

A full Certification Authority (CA) system was implemented, leveraging on the platform base technologies. Modules for generation, handling and revocation of certificates were included. Also, Secure Socket Layer technology was added to the data extraction and collection module.

At the present time, CENICA is preparing the administrative procedures to begin generating certificates for all SINAICA participants.

12. 15 copies of the final report, both in Spanish and English.

This document and its appendices are the final report.

Commentaries and Recommendations

As an overall evaluation of the project results, we at Matías Software Group believe to have fulfilled the stated goals.

However, we believe that a project of this nature has many and varied development paths, and in particular we suggest the following:

- Query modules internationalization in several languages and units.
- 8 and 24 hour mobile averages.
- Additional statistic processes.
- Data validation helper modules.
- General public concern queries, such as zonal maps, alerts, outdoor exercise recommendations, etc.

We also believe that correct functioning of the whole system requires continuous support and feedback between CENICA and the integrated networks, as well as financial support, both at the State and Federal levels.

One of the areas in more need of attention, is personnel training in methodology and computer skills.

The whole project will not see its full extent and application until federal regulations and standards are adopted. This was the number one concern we heard from all networks.

To finish, we want to thank all the enthusiast support we had from CENICA personnel during development of the project, and also thank JICA for giving us the opportunity to collaborate on this noble effort: to better the air we breathe, to improve the world we live in.

Appendix 1

Data file type and format characteristics and storage technologies used at the RAMAs. (Automatic Atmospheric Monitoring Networks, Redes Automáticas de Monitoreo Atmosférico)

Introduction.

This document details the results of the diagnosis, analysis and implementation process for the networks incorporated to the SINAICA.

We proceeded with an analysis of the implementation of the acquisition , process and data storage systems of the different RAMAs, as well as the communications infrastructure available at each one of them, with the purpose of finding the possible strategies for acquiring the received information from the stations that integrate the net and the availability of media for the communication with the CENICA equipment.

Throughout this stage of the analysis we detected the following problematic:

Nonexistence of a network as a whole.

In several projected cities and as noted in table 1 of the reference terms, we didn't really count with a "monitoring network", we counted with a set of automatic monitoring stations which were not connected, in some cases with incipient connectivity for some of them (case of the Bajío cities), in others with communication towards polling systems abroad (Ciudad Juárez case) and in others with no connectivity capabilities at all (case of the Baja California networks).

In all of these cases, we didn't find a "Control Center", and even when in some cases we counted with the compromise from behalf of the local authorities to establish it, this wasn't possible within the contract's time frame, so we had to present alternative schemes for their inclusion to the SINAICA.

Communication problems with the "Control Center"

In those cases where a "Control Center" was found, the diagnosis of the communication schemes between the stations that formed the net and the control center threw the following:

- Proprietary protocols in the different data acquisition devices.
- Use of RS-232 serial links through "modems", except for Ciudad Juárez, where radio links are used.
- In some cases the transmission is made through leased lines and in other using commuted lines from the public telephone network.

In general the main problems that the RAMAs face in its daily operation are related with this communication, during the present contract, more than 90% of the problems that the networks revealed fall in this category. Communication loss between stations and polling systems was common even on the networks operated by US people.

Meanwhile on the control center side we found that:

- Some networks don't count with a structured storage system, get the information from their stations through the generation of "reports" in text with various formats.
- When we counted with structured data storage, this was made with differing systems, each one with its own definitions regarding the file structure and/or used data formats. The only case which counts with a high-level data storage system is Monterrey city, where a relational database manager based on SQL is used.
- Generally the hourly averages are calculated locally.
- In general we couldn't determine the existence of neither established homogeneous procedures nor standard data validation means; the "raw" received data set is subject to each network's manual validation and quality assurance process, which can take as long as weeks or even months.
- All of the control centers of the monitoring networks count with some level of Internet connectivity, although for the reasons exposed below we couldn't consider them to readily have interconnection.

Deficiencies on the use of Flags.

Even though in all checked cases, the monitoring equipment installed at the stations counted with the ability of associating "flags" to the data for qualification, only the stations of Ciudad Juárez are taking advantage of the systematic use of this feature. On this respect we found that:

- The sensors for different state and operative conditions of the monitoring equipment were not connected to the "datalogger".
- The "dataloggers" haven't been programmed with the detection limit parameters, ranges or flagging criteria.
- The used systems commonly map the problem detecting flags to some invalid value as a marker (i.e. '-99.99'), loosing at that point the origin of the problem.
- There are no homologous criteria regarding the semantics and/or coding of the flags in use.

The Collect/Transmit system.

Due to the detected problematic, it is required that the collecting system implements generic and diverse importing means that allow to retrieve the hourly averages of the measurements from the native storage media of each network and in some cases directly from the "dataloggers" so they can be transmitted in a homologous way to the concentrating SINAICA system, which requires the development of a higher-level abstraction model.

This section details the methodology and means used during the development of the collecting system.

The design of these incorporate the following considerations:

- Minimize the impact on the operation of the existing systems.
- Minimize the impact on the workload on the network operating personnel.

- Maximize the automation of the importing process of the data.
- Achieve the most possible of generality.
- Guarantee the security of the transition of the information.

To be able to formalize the term "generic *means*" it was required to elaborate an abstraction model that generalizes the notion of 'datum' which is summarized as the following:

A 'datum', for example the measurement's hourly average, invariably has a related set of 'meta data' which characterize it:

- Measured variable. (Ozone, NOX, Temperature, etc.)
- Related measuring unit. (ppm, °C, m/sec, etc.)
- Origin station.
- Time span. (Date and hour)
- Etc.

Depending on the existence or not of structured systems and on the particular design used for a monitoring network, the metadata that characterize a given datum will be one and only one of the following classes:

1) Constants.
2) Intrinsics.
3) Table of file name function.
4) Field location function.
5) Function of the value of other field in the record.

Note that:

- All the functions have a defined inverse.
- The "text reports" can be regarded as "tables" developing the adequate "parsers".

- The "datalogger" case can be regarded as a particular case of a sequential access table.

The datum location, within the files that form the system tables is determined by those non-constant nor intrinsic metadata.

From this model it is inferred that it is possible to consider the development of an "Universal Extractor" that made an abstraction of the design differences between the various systems in use.

Such an Universal Extractor, through simple parametrization, can obtain the data using the meta data as indexes.

The parametrization for each one of the different storage systems is reduced then to classifying each one of the meta data in any of the mentioned classes and the actual mapping functions.

The different formats and/or storage systems of the data found at the visited networks and the existence or lack of a "Control Center" are summarized in the next table:

Network/City	**Key**	**Control center**	**Format or System**
México	CMX	YES	DBF
Guadalajara	GDL	YES	DBF
Monterrey	MTY	YES	SQL
Toluca	TLC	YES	DBF
Puebla	PUE	YES	EDAS
CENICA	CEN	Almost	EDAS
Salamanca	SAL	YES	TEXTO
Ciudad Juárez	CJU	NO	TEXT
Tijuana	TIJ	NO	Datalogger

Network/City	Key	Control center	Format or System
Mexicali	MXC	NO	Datalogger
Rosarito	ROS	NO	Datalogger
Tecate	TKT	NO	Datalogger

In the United States an important proportion of the operating environment monitoring networks use ESC company's E-DAS system which stores the information in proprietary, non-standard formats.

In Mexico, on the other side, most of the visited monitoring networks use systems developed under dBase or similar tools, so they store the received information in databases that use DBF (Data Base Format) variants.

Because this is the dominant class among such networks we will focus first on this format characteristics.

The DBF format.

The DBF format was originally developed by Ashton Tate business for the dBase II product in 1983 and its huge popularity resulted in various “compatible” systems, among which we can mention Clipper and Fox-Pro.

Today they are considered to be “obsolete” database systems, but because the development of the different storage systems in use date from the late 90s, it is understood that it is being used today.

The database systems of the different monitoring networks that use it were designed and developed by different entities and there is no homogeneous structure regarding fields, data types or file naming, which would impend the development of an unified system using any of the above mentioned tools in its native form.

However the internal structure of the DBF files is now well documented, and even though there exist variations of it, product of revisions that the dBASE product had along its life cycle and the extensions that the different adopting software houses introduced in it, the general structure of the internal format was kept without radical changes.

As a result we regard it possible to retrieve the stored information directly from the body of the files.

Next we detail the internal structure of the DBF files with the purpose of clarifying the methodology used by the "Universal Extractor".

In the following descriptions we refer to the dBASE product, but you will have to consider any other product compatible with the annotated version as being on the same category, except for the cases where there exist differences that are explicitly indicated.

A file in DBF format is structured in three parts: the heading, the field description and the actual data records, just as shown in figure 1.

Heading
Field description
Data records

Figure 1.

The first byte on the heading identifies the version of the software used to create the file, which could be any of the following:

Version	**Software**
02H	dBase II
03H	dBase III-IV
83H	dBase III+ with "memo" fields
8BH	dBase IV+ with "memo" fields
F5H	FoxPro with "memo" fields

Table 1

The length of the heading itself and that of the field description records are set by the version number, in the case of dBase II (version 02H), the heading occupies bytes 0 through 7, followed by the field description records for which a maximum of 32 bytes is reserved.

Then, in version 2 (02H) of a DBF file the following structure will be used:

Position	Length	Description
0 (00H)	1	Version (always 02h)
1 (01H)	2	Number of records (0-FFFFH)
3 (03H)	3	Date of last write access (DDMMYY) in binary
6 (06H)	2	Data record length plus one, up to 1000.
8-519 (08H-207H)	16*n	16 bytes for each field description record
16*n+9	1	Sets the heading end (0DH)

Table 2

In this dBase version, as the space for a maximum of 32 possible fields is reserved, the data records will invariably start from byte 521 (209H), and the unused space for field descriptions that follows the end of heading mark will appear full of zero bytes (00H).

The 16 bytes that describe each field under dBase II have the following structure:

Position	Length	Description
0 (00H)	11	Field name (ASCII string)

Position	Length	Description
11 (0BH)	1	Field type See table 6.
12 (0CH)	1	Field length in bytes Binary 0 until FFH
13 (0DH)	2	Reserved
15 (0FH)	1	Decimal positions at the field

Table 3

In dBase III, compatibles and later versions the heading always has a 32 byte length, followed by the field description records, in which, as opposed to the older version and having increased the field maximum to 128, the space is not expressly reserved for the field maximum, and the data records will follow the field description records.

The following table shows the resulting structure.

Position	Longitude	Description
0 (00H)	1	Version (see table 1)
1 (01H)	3	Last write access date (YYMMDD) In binary
4 (04H)	4	Number of records in the file 32 bits, unsigned, *'little endian'*
8 (08H)	2	Heading length 16 bits, unsigned, *'little endian'*
10 (0AH)	2	Data record length plus one
12 (0CH)	20	Reserved
32 (20H)	32*N	32 bytes for every field description record
32 * (N+1)	1	Marks the heading end (ODH)

Table 4

On dBase III and later the field description records increased to 32 bytes, with the following structure:

Position	Length	Description
0 (00H)	11	Field name (ASCII string)
11 (0BH)	1	Field type See table 6.
12 (0CH)	4	Reserved
16 (10H)	1	Field length in bytes
17 (11H)	1	Field decimals
18 (12H)	14	Reserved

Table 5

In the field descriptions, the field type is coded by the means of one only byte, an ASCII character from the following table, where in dBASE II only the first tree values are used.

Character	Field type	Possible values
C	String	ASCII characters
N	Numeric	- 0...9
L	Boolean	YyNnTtFf
D	Date	Numeric, in YYYYMMDD format
M	Memo	Block number
F	Numeric	- 0...9

Table 6

The data records of all the dBase versions have exactly the same structure: a variable length string.

Length that depends on the number of fields and set in the heading, where the first byte is reserved as a flag, and the next ones for the ASCII representation of the field's values using no marker or delimiter, the 'string' fields are padded with white spaces (character 20H), just like any field that doesn't have any data.

The first byte is used by dBASE as an indicator that the record has been erased through the delete operation, which for speed is just marked with the '*' character, postponing until the use of the pack operation that the marked records are reused and overwritten.

This last has to be taken into account when reading a record because the number of records that appears in the heading includes those marked as deleted.

A given *n* field's *O offsets* results from the sum of the *L* lengths of the preceding fields.

$$On=\sum_{a=1}^{n-1} La$$

As you may see, the variations in the internal format of the different versions is fundamentally reduced to the size of some structures and the order of some fields, but the operations involved in calculating the index for a given field inside the data records are the same.

The "Universal Extractor" uses in a direct way the DBF files of the different data bases without having to know the high-level structure of the original data base.

Variations between the studied networks.

None of the three networks that use this format use the same structure , even though it only takes simple arithmetics for the first two below described to access the records:

In the case of the Mexico city network a monthly table is used for every variable subject to monitoring, where one record (row) is stored for every hour, starting with the one belonging to the 0:00 hours of the first day of the month, and where the fields (columns) refer to the stations. Even though there is a reserved column for the flag storage, this column is always empty in the files that the network provides.

We consider that this format is the most rigid of the ones used, because all the tables have to be restructured to add new stations to the system.

In the case of Toluca city, one table is used per month per variable per station, it is in the file name where the corresponding variable and station name is coded, just having a record for every hour and a couple of fields with the value and the flag.

We consider that even if it is the simplest format it is also the most inefficient due to the great quantity of involved files.

In the Guadalajara network case, only one monthly file is generated, where one record is stored for every day for every variable-station combination, and where the columns correspond to every hour of the day.

Although the access becomes more complex because a parallel index file is required to avoid sequential searches, once the file is indexed, a compact, efficient and easily extensible design results. This is why we opted to use it as an intermediate format for those cases where the networks didn't count with structured files, cases of Salamanca and Ciudad Juárez, and we ended up using the same module as we will further describe.

The E-DAS format from ESC.

The internal storage format used by Environmental Systems Corporation's E-DAS proprietary format doesn't count with any documentation.

The system counts with incorporated exportation means for the EPA AIRS format, however these are not used in Mexico so they were not considered as an option.

Notwithstanding this, and because the relevant legislation allows proprietary format analysis with interoperability purposes, we used decoding techniques that allowed the incorporation of this format to the "Universal Extractor" model.

The details of the format are presented next:

A data file is created for every month, the file name tells the month and year of the readings. The name comes with three letters followed by four digits and the ".dat" extension. The two first digits refer to the two least significant digits of the year and the following two represent the month. So, the file corresponding to May of 2003 would be called "hly0305.dat."

The file contains exclusively records, without heading. The size of the records is 648 bytes and are of fixed position. There is one record for each day, station and measurement. Inside every record there are first a group of fields that describe the record and following that the values for the day.

On the table we describe the positions and fields for the first group:

Position	Length	Description
0 (00H)	2	Station number
2 (02H)	2	Variable number
4 (04H)	8	Variable name
12 (0CH)	8	Variable units
20 (14H)	8	Station name
28 (1BH)	92	Constant values

After position 120 the hourly data are located, 24 per record. The length of each group is of 22 characters.

On the table are shown the positions, fields and descriptions for each group, with positions relative to the group:

Position	Length	Description
0 (00H)	4	Floating point variable with the measurement value of the variable
4 (04H)	6	Reading flags
10 (0AH)	1	Blank space
11 (OBH)	1	Serial with the hour
12 (0CH)	9	Reserved field

For example:

```
00000000: 3031 3031 5753 2020 2020 2020 4d2f 5320 0101WS M/S
00000010: 2020 2020 5055 4542 4c41 2d31 0100 0100 PUEBLA-1....
00000020: 9af9 79c4 9a3f 1c46 2a2a 2a2a 2a2a 2a2a ..y..?.F********
00000030: 2va2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a ****************
00000040: 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a ****************
00000050: 2a2a 2a2a 2a2a 2020 2020 2020 2020 202a ****** *
00000060: 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a ****************
00000070: 2a2a 2a2a 2a2a 2a2a ed64 0640 0000 0000 ********.d.@....
00000080: 0000 2000 0000 0000 0000 0020 202a 2dbd .. ........ *-.
00000090: 8940 0000 0000 0000 2001 0000 0000 0000 .@...... .......
000000a0: 0020 202a 0590 9a40 0000 0000 0000 2002 . *...@...... .
000000b0: 0000 0000 0000 0020 202a 6dd0 d440 0000 ....... *m..@..
000000c0: 0000 0000 2003 0000 0000 0000 0020 202a .... ........ *
000000d0: 4900 c640 0000 0000 0000 2004 0000 0000 I..@...... .....
000000e0: 0000 0020 202a f84b c840 0000 0000 0000 ... *.K.@......
000000f0: 2005 0000 0000 0000 0020 202a 4b8c a040 ........ *K..@
00000100: 0000 0000 0000 2006 0000 0000 0000 0020 ...... .......
00000110: 202a 282f ab40 0000 0000 0000 2007 0000 *(/.@...... ...
00000120: 0000 0000 0020 202a e4a9 5640 0000 0000 ..... *..V@....
00000130: 0000 2008 0000 0000 0000 0020 202a e92c .. ........ *.,
00000140: 0c40 0000 0000 0000 2009 0000 0000 0000 .@...... .......
00000150: 0020 202a 271c 6440 0000 0000 0000 200a . *'.d@...... .
00000160: 0000 0000 0000 0020 202a 1352 aa40 0000 ....... *.R.@..
00000170: 0000 0000 200b 0000 0000 0000 0020 202a .... ........ *
00000180: 651f 8d40 0000 0000 0000 200c 0000 0000 e..@...... .....
00000190: 0000 0020 202a 1410 b040 0000 0000 0000 ... *...@......
000001a0: 200d 0000 0000 0000 0020 202a b4cf 8740 ........ *...@
000001b0: 0000 0000 0000 200e 0000 0000 0000 0020 ...... .......
000001c0: 202a c15d 8a40 0000 0000 0000 200f 0000 *.].@...... ...
000001d0: 0000 0000 0020 202a 0d58 a240 0000 0000 ..... *.X.@....
000001e0: 0000 2010 0000 0000 0000 0020 202a 5d3b .. ........ *];
000001f0: 9440 0000 0000 0000 2011 0000 0000 0000 .@...... .......
00000200: 0020 202a e39c 0e40 0000 0000 0000 2012 . *...@...... .
00000210: 0000 0000 0000 0020 202a f5e3 0d40 0000 ....... *...@..
00000220: 0000 0000 2013 0000 0000 0000 0020 202a .... ........ *
00000230: e509 d93f 0000 0000 0000 2014 0000 0000 ...?...... .....
00000240: 0000 0020 202a f500 8040 0000 0000 0000 ... *...@......
00000250: 2015 0000 0000 0000 0020 202a aaf0 7340 ........ *..s@
00000260: 0000 0000 0000 2016 0000 0000 0000 0020 ...... .......
00000270: 202a efdc 5c40 0000 0000 0000 2017 0000 *..\@...... ...
00000280: 0000 0000 0020 202a 3031 3032 5744 2020 ..... *0102WD
00000290: 2020 2020 4752 4144 4f53 2020 5055 4542 GRADOS PUEB
000002a0: 4c41 2d31 0100 0100 9af9 79c4 9a3f 1c46 LA-1......y..?.F
000002b0: 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a ****************
```

We can see a whole record and the first group of the next one.

The heading can be seen on figure 1:

```
00000000: 3031 3031 5753 2020 2020 2020 4d2f 5320 0101WS M/S
00000010: 2020 2020 5055 4542 4c41 2d31 0100 0100 PUEBLA-1....
00000020: 9af9 79c4 9a3f 1c46 2a2a 2a2a 2a2a 2a2a ..y..?.F********
00000030: 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a ****************
00000040: 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a ****************
00000050: 2a2a 2a2a 2a2a 2020 2020 2020 2020 202a ****** *
00000060: 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a 2a2a ****************
00000070: 2a2a 2a2a 2a2a 2a2a ********
```

Figure 1 Record heading.

```
00000070: ed64 0640 0000 0000 .d.@....
00000080: 0000 2000 0000 0000 0000 0020 202a .. ........ *
```

Figure 2 First value of the record.

On figure 2 we see the value, which is the first one on the record. As it is observed, the first four bytes have the value of the measurement. Collapsing the sub-record and starting the count on zero:

```
ed 64 06 40 00 00 00 00 00 00 20 00 00 00 00 00 00 00 00 20 20 2a
 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
```

The value is an Intel native *float* and it is *little-endian* . On byte number eleven the hour serial is stored. It starts from zero, but may have some offset.

Flags are located from byte number four to number nine. On this example, the flags are on zeros.

```
00003350:                          acef b340 4010 ....... *...@@.
00003360: 0000 0000 2003 0000 0000 0000 0020 202a .... ........ *
```

Figure 3 Record with values on the flag.

On this other example, corresponding to the record shown on figure 3, the corresponding flag value is: 401000000000 in hexadecimal. Let us see, then, that the hour serial is three at the eleventh position.

```
ac ef b3 40 40 10 00 00 00 00 20 03 00 00 00 00 00 00 00 20 20 2a
 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
```

Index Files

For every data file there is an index file. It has the same name but with ".inx" extension. For example, the index file belonging to **hly0307.dat** is **hly0307.inx** .

This file is also composed of fixed-length 128 byte records. Each record is composed of a small 20 byte heading in which the station and variable data appears. On the two first bytes appears the variable and in the next two the station. The rest of the heading contains the '001H************' constant.

On each record, after the heading, fifty four short integers occur. The first thirty one fields match with the calendar days, and the following twenty three appear in zeros. The months with less than thirty one days have zeros in the fields that are not used.

In the index file there is a record per station and variable. In each record there is a reading for each day in one of the first thirty one fields. In case there was no reading done, a zero appears. Otherwise the number with the data file record with the actual readings appears. This is, the procedure for reading a record in the data file given the station and variable to read, goes through finding in the index the record with the mentioned tuple and seeking inside the record the desired day. The read value is the record number that will be read from the data file.

Flag field description.

The possible flags that happen in each of the fields of the record are divided in data flags and state flags, represented in table 1 and table 2, respectively.

According with the examined listings, the six used flags are '<', 'D', 'C', 'M', '-', 'V' and 'I'.

From the six flag bytes, the first three represent data flags and the last three state flags. The most used flags and the mask with which they are obtained can bee seen on table 3, exclusively taking the first tree bytes.

In the stat flags case, the flag that says that the reading should not be used is obtained with the 0x000100 mask over the three least significant bytes. In no case did we find any other flag lifted in this group.

Flag	Meaning
<	Less than ##% Data
P	Power Fail
D	Disabled
T	Out-of-Control
F	Boiler O®-Line
B	Bad Status
C	Calibration
M	Maintenance
O	Analog Over range
U	Analog Under range
A	Arithmetic Error
+	Maximum
-	Minimum
R	Rate of Change
H	High-High Alarm
L	Low-Low Alarm
h	High Alarm
l	Low Alarm
J	High Rate of Change
j	Low Rate of Change
V	DIS #1 Obs
W	DIS #2 Obs
X	DIS #3 Obs
Y	DIS #4 Obs

Flag	Meaning
Z	DIS #5 Obs.
f	Undocumented
	Frame 1: Data flags

Flag	Meaning
I	Ignore?
?	Undocumented
¤	Undocumented
>	Undocumented
=	Undocumented
m	Undocumented
^	Undocumented
v	Undocumented
E	Undocumented
d	Undocumented
9	Undocumented
z	Undocumented
a	Undocumented
Q	Undocumented

Flag	Mask
<	0x001000
D	0x004000
C	0x040000
M	0x080000
-	0x000100

Flag	Mask
V	0x000001
	Frame 3: Data flags

The SQL RDBMS format

The use of a Relational Database Managing Software that additionally uses a high-level language such as SQL for storage greatly simplifies the data extraction.

In the case of Monterrey city there is an MS-SQL Server installed that even if it is a commercial product with high licensing fees, it is in reality no more than a specialized version of Sybase made for the Microsoft Windows NT operating system. Sybase counts with ample documentation and support for diverse operating systems.

Using the standard Perl5 libraries for the Sybase SQL Server, which have no licensing fees, the data extraction was reduced to a simple SQL sentence:

```
SELECT * FROM hravedata
WHERE parameter = ? AND date >= ? AND date <= ?
ORDER BY date;
```

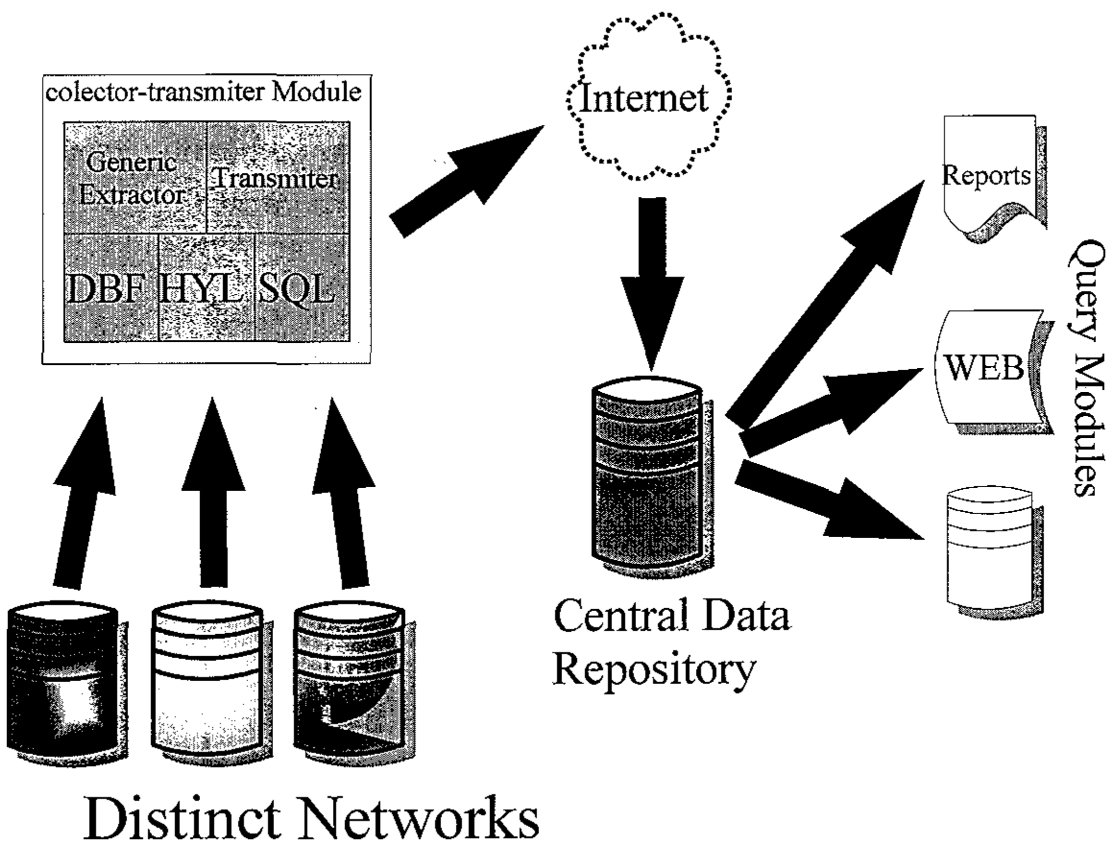
It is worth mentioning that the Monterrey system implementation uses a separate database for each station, using the names “SUROESTE”, “NOROESTE”, “CENTRO”, “NOESTE” and “SURESTE” so each one of them has to be interrogated separately.

Appendix 2
Implementation

Introduction

This appendix describe the implementation details of the new version of SINAICA

The flow of data between the distinct components of SINAICA are shown in a schematic manner in the following diagram:



Its important to note that this general schema of operation has proved to be successful, basically because its modular design, that makes it a flexible and extensible one.

That way this new version of SINAICA doesn't pretend to modify the general design, was find at the same time that was extended the area covered by the system to new networks, enhance and strengthen its capabilities.

This contract permits us to re-implement some of its components, extend the functionality of others and reduce the technological dependence of the system as a whole.

Autonomous Data Collector and Transmitter System

In previous SINAICA versions, the data extraction of each networks own system and its transmission to the central database was delegated to a program running under the MS Windows operating system, which would run in one of the control center's computers.

This scheme presented several inconveniences detected on previous versions, which can be summarized in a significant raise both as operation costs and as workload for the network operators.

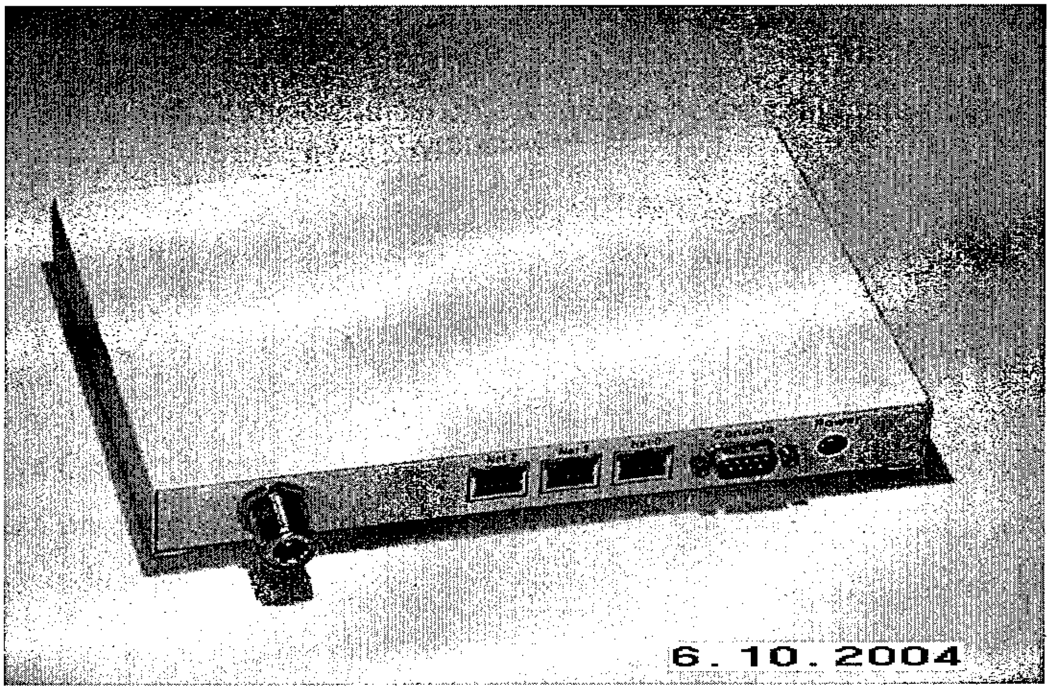
We started from the base that the new system had to comply with eight essential characteristics: strength, efficiency, extensibility, portability and ease of support, while it would still reduce the associated expenses.

To comply with these proposed requirements we established as a first requisite that all the used software should be open and with no licensing fees, because this guarantees the portability and support ease, as well as it affects the economic aspect; Appendix 3 mentions other characteristics of open software "Free Software".

Matias Software Group had developed for other communication projects the "MSG Communication Server" platform, using a special variant of the Linux operating system for embedded systems. The utilized hardware is an embedded computer based on the AMD Elan520 processor, an Intel compatible architecture, of low cost, low energy consumption and zero maintenance, which doesn't use mobile parts and is easily replaceable.

We found that the extraction modules could be fitted and ported to this platform, eliminating the dependency towards a dedicated personal computer, and from these efforts came the autonomous Data Collector and Transmitter System (SCTD).

The following picture show the aspect of the autonomous DCTS:



Remote Administration

To approach this issue we integrated a module which allows to query the SCTD's logs in real time through a web page accessible via the monitoring network's intranet. Such module allows the timely diagnosis of problems and helps the central technical support staff. Furthermore, as part of the specialized consulting module, a system was integrated, which follows a central network transfer and relevant event log.

Appendix 3
Technical Details

Introduction.

This document details the technical characteristics of the programs and subsystems that form the third version of SINAICA

Platforms:

The deferents modules of the system runs in one or more of three distinct platforms:

- Fedora Core 2 (Kernel Linux v2.6.x) for i386 [FC2]
- MSG Communications Server (Kernel Linux v2.4.x) for ELAN520 [MCS]
- MS-Windows 98 svr2 for i388 [WIN32]

In the first annotated runs the data base management system and the HTTP server that forms the central storage and query subsystem.

In the second runs the transmitter and extractor system installed in the monitoring networks.

At least, in MS Windows runs some of the manual data management modules, like the pilot balloons capture system or the organic and elemental carbon management module.

Requirements

All of the system's functionality rests on the use of four fundamental tools: The Linux operating system, the PostgreSQL relational database manager, the GranJefe HTTP server (derived from Apache) and the Perl5 programming language, all among the best fruits of the "free software" community and of course free of licensing costs and of free distribution.

Although the system's nucleus is composed of free software, we have taken care that none of the system's components, as marginal as they may be, require the use of proprietary software subject to licensing costs. Even the components developed for the Microsoft Windows platform have been programmed using techniques that allow their operation in other platforms and they do not require components exclusive of this operating system.

All of the above looked after since the initial analysis and design stages so that all of the components can be freely distributed and the national and international communities can take advantage of these developments without falling into licensing costs.

The only module that drifts apart from these linings is the Excel macro package developed for the statistics process of the COV data, as being so explicitly required by CENICA because there is a huge amount of data that they have stored in this system.

Compilation Parameters.

The development of the system components was done using the Perl5 programming language.

Perl5 is an interpreted programming language, and because of that, the resulting executable modules don require to be compiled.

In all cases the standard interpreter of the used platform was used, that neither require to be compiled, only to satisfy the completeness of this report the original compilation parameters of the interpreter in all platforms used is annotated:

Perl in Linux: (Perl 5.8.3 in Fedora Core 2)

```
Summary of my perl5 (revision 5.0 version 8 subversion 3) configuration:
  Platform:
    osname=linux, osvers=2.4.21-4.elsmp, archname=i386-linux-thread-multi
     uname='linux tweety.devel.redhat.com 2.4.21-4.elsmp #1 smp fri oct 3 17:52:56
edt 2003 i686 i686 i386 gnulinux '
           config_args='-des -Doptimize=-O2 -g -pipe -march=i386 -mcpu=i686
-Dversion=5.8.3 -Dmyhostname=localhost -Dperladmin=root@localhost -Dcc=gcc
-Dcf_by=Red Hat, Inc. -Dinstallprefix=/usr -Dprefix=/usr -Darchname=i386-linux
-Dvendorprefix=/usr -Dsiteprefix=/usr -Duseshrplib -Dusethreads -Duseithreads
-Duselargefiles -Dd_dosuid -Dd_semctl_semun -Di_db -Ui_ndbm -Di_gdbm -Di_shadow
```

```
-Di_syslog    -Dman3ext=3pm    -Duseperlio    -Dinstallusrbinperl    -Ubincompat5005
-Uversiononly -Dpager=/usr/bin/less -isr -Dinc_version_list=5.8.2 5.8.1 5.8.0'
    hint=recommended, useposix=true, d_sigaction=define
                   usethreads=define     use5005threads=undef     useithreads=define
usemultiplicity=define
    useperlio=define d_sfio=undef uselargefiles=define usesocks=undef
    use64bitint=undef use64bitall=undef uselongdouble=undef
    usemymalloc=n, bincompat5005=undef
  Compiler:
     cc='gcc', ccflags ='-D_REENTRANT -D_GNU_SOURCE -DTHREADS_HAVE_PIDS -DDEBUGGING
-fno-strict-aliasing            -I/usr/local/include             -D_LARGEFILE_SOURCE
-D_FILE_OFFSET_BITS=64 -I/usr/include/gdbm',
    optimize='-O2 -g -pipe -march=i386 -mcpu=i686',
       cppflags='-D_REENTRANT -D_GNU_SOURCE -DTHREADS_HAVE_PIDS -DDEBUGGING -fno-
strict-aliasing -I/usr/local/include -I/usr/include/gdbm'
          ccversion='', gccversion='3.3.3  20040412  (Red  Hat  Linux  3.3.3-7)',
gccosandvers=''
    intsize=4, longsize=4, ptrsize=4, doublesize=8, byteorder=1234
    d_longlong=define, longlongsize=8, d_longdbl=define, longdblsize=12
    ivtype='long', ivsize=4, nvtype='double', nvsize=8, Off_t='off_t', lseeksize=8
    alignbytes=4, prototype=define
  Linker and Libraries:
    ld='gcc', ldflags =' -L/usr/local/lib'
    libpth=/usr/local/lib /lib /usr/lib
    libs=-lnsl -lgdbm -ldb -ldl -lm -lcrypt -lutil -lpthread -lc
    perllibs=-lnsl -ldl -lm -lcrypt -lutil -lpthread -lc
    libc=/lib/libc-2.3.3.so, so=so, useshrplib=true, libperl=libperl.so
    gnulibc_version='2.3.3'
  Dynamic Linking:
      dlsrc=dl_dlopen.xs,  dlext=so,  d_dlsymun=undef,  ccdlflags='-rdynamic  -Wl,-
rpath,/usr/lib/perl5/5.8.3/i386-linux-thread-multi/CORE'
    cccdlflags='-fPIC', lddlflags='-shared -L/usr/local/lib'
   Compiled at Apr 15 2004 13:09:17
```

Perl in MSG Communications Server

```
Summary of my perl5 (revision 5 version 8 subversion 4) configuration:
  Platform:
    osname=linux, osvers=2.4.26-lfs-i386-fc1, archname=i386-linux-thread-multi
     uname='linux xochitl 2.4.26-lfs-i386-fc1 #3 thu jul 15 21:10:51 cdt 2004 i686
genuineintel unknown gnulinux '
    config_args=''
    hint=previous, useposix=true, d_sigaction=define
                   usethreads=define     use5005threads=undef     useithreads=define
usemultiplicity=define
    useperlio=define d_sfio=undef uselargefiles=undef usesocks=undef
    use64bitint=undef use64bitall=undef uselongdouble=undef
    usemymalloc=n, bincompat5005=undef
  Compiler:
     cc='cc', ccflags ='-D_REENTRANT -D_GNU_SOURCE -DTHREADS_HAVE_PIDS -fno-strict-
aliasing -I/usr/include/db-1.86',
    optimize='-O2',
     cppflags='-D_REENTRANT -D_GNU_SOURCE -DTHREADS_HAVE_PIDS -fno-strict-aliasing
-I/usr/include/db-1.86 -D_REENTRANT -D_GNU_SOURCE -DTHREADS_HAVE_PIDS -fno-strict-
aliasing -I/usr/include/db-1.86'
    ccversion='', gccversion='3.3.3', gccosandvers=''
    intsize=4, longsize=4, ptrsize=4, doublesize=8, byteorder=1234
```

```
    d_longlong=define, longlongsize=8, d_longdbl=define, longdblsize=12
    ivtype='long', ivsize=4, nvtype='double', nvsize=8, Off_t='off_t', lseeksize=4
    alignbytes=4, prototype=define
  Linker and Libraries:
    ld='cc', ldflags =''
    libpth=/lib /usr/lib
    libs=-lnsl -ldb -ldl -lm -lcrypt -lutil -lpthread -lc
    perllibs=-lnsl -ldl -lm -lcrypt -lutil -lpthread -lc
    libc=/lib/libc-2.3.3.so, so=so, useshrplib=true, libperl=libperl.so
    gnulibc_version='2.3.3'
  Dynamic Linking:
         dlsrc=dl_dlopen.xs,  dlext=so,  d_dlsymun=undef,  ccdlflags='-Wl,-E  -Wl,-
rpath,/usr/lib/perl5/5.8.4/i386-linux-thread-multi/CORE'
    cccdlflags='-fpic', lddlflags='-shared'
  Compiled at Jul 16 2004 16:50:08
```

Perl in Windows: (ActiveState Corp's Perl 5.8.0 build 806)

```
Summary of my perl5 (revision 5 version 8 subversion 4) configuration:
  Platform:
    osname=MSWin32, osvers=4.0, archname=MSWin32-x86-multi-thread
    uname=''
    config_args='undef'
    hint=recommended, useposix=true, d_sigaction=undef
    usethreads=undef use5005threads=undef useithreads=define usemultiplicity=define
    useperlio=define d_sfio=undef uselargefiles=define usesocks=undef
    use64bitint=undef use64bitall=undef uselongdouble=undef
    usemymalloc=n, bincompat5005=undef
  Compiler:
     cc='cl', ccflags ='-nologo -Gf -W3 -MD -Zi -DNDEBUG -O1 -DWIN32 -D_CONSOLE
-DNO_STRICT    -DHAVE_DES_FCRYPT          -DNO_HASH_SEED     -DPERL_IMPLICIT_CONTEXT
-DPERL_IMPLICIT_SYS -DUSE_PERLIO -DPERL_MSVCRT_READFIX',
    optimize='-MD -Zi -DNDEBUG -O1',
    cppflags='-DWIN32'
    ccversion='', gccversion='', gccosandvers=''
    intsize=4, longsize=4, ptrsize=4, doublesize=8, byteorder=1234
    d_longlong=undef, longlongsize=8, d_longdbl=define, longdblsize=10
    ivtype='long', ivsize=4, nvtype='double', nvsize=8, Off_t='__int64', lseeksize=8
    alignbytes=8, prototype=define
  Linker and Libraries:
      ld='link', ldflags ='-nologo -nodefaultlib -debug -opt:ref,icf   -libpath:
"C:\Perl\lib\CORE"  -machine:x86'
    libpth=C:\PROGRA~1\MICROS~3\VC98\lib
    libs= oldnames.lib kernel32.lib user32.lib gdi32.lib winspool.lib  comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib  netapi32.lib uuid.lib wsock32.lib
mpr.lib winmm.lib  version.lib odbc32.lib odbccp32.lib msvcrt.lib
       perllibs=   oldnames.lib  kernel32.lib  user32.lib  gdi32.lib  winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib  netapi32.lib uuid.lib
wsock32.lib mpr.lib winmm.lib  version.lib odbc32.lib odbccp32.lib msvcrt.lib
    libc=msvcrt.lib, so=dll, useshrplib=yes, libperl=perl58.lib
    gnulibc_version='undef'
  Dynamic Linking:
    dlsrc=dl_win32.xs, dlext=dll, d_dlsymun=undef, ccdlflags=' '
       cccdlflags='  ', lddlflags='-dll  -nologo -nodefaultlib -debug -opt:ref,icf
-libpath:"C:\Perl\lib\CORE"  -machine:x86'

  Compiled at Jun  1 2004 11:52:21
```

Developed Programs and Libraries used.

The following table is the complete list of the developed programs and the libraries used by any one of them.

Program	**Function**	**Platform**
td3gui.pl	TD3.theodolite capture application	WIN
	Win32API::CommPort Win32::SerialPort Win32::API Win32::GUI DBI (1.43) Globo::Query Globo::Graph Globo::Calc Math::Trig (1.02) Globo::Parse Globo::TD3 Time::Local (1.07) Tk (804.027) Win32::TieRegistry (0.25)	
carbongui.pl	OC and EC capture application	WIN
	Win32::API Win32::GUI DBI (1.43) DBD::ODBC Tk (804.027) Win32::TieRegistry (0.25)	
rmaxtr	Data extraction and collection system	MCS

	RMA::Storage MIME::Base64 (2.12) Storable (2.06) RMA::Transfer LWP::UserAgent (2.001) Symbol (1.04) RMA::Xtract Time::Local (1.04) RMA::Xtract::DataLogger Device::SerialPort (1.000002) RMA::Xtract::DBF XBase (0.241) RMA::Xtract::HLY RMA::Xtract::Mzip Sys::Syslog (0.03)	
dat2dbf	Salamanca's data format parser.	MCS
epa2dbf	Ciudad Juárez data format parser	MCS
	DB_File (1.808) Fcntl (1.05) File::Basename (2.72) Time::Local (1.07) XBase::Fast	
RMA::PcdCA.pm	GranJefe data reception module	FC2
	Data::Dumper (2.121) MIME::Base64 (2.21) RMA::Saver DBI (1.42) DBD::Pg (1.32) RMA::Normalizer Time::Local (1.07) Storable (2.09) Unicode::String (2.07) XML::Simple (2.12)	
MSG::RMA.pm	GranJefe query	FC2

	DBI (1.42) DBD::Pg (1.32) Encode (1.99) MSG::RMA::Graf SVG::TT::Graph::Bar (0.06-msg) SVG::TT::Graph::Line (0.06) SVG::TT::Graph::TimeSeries (0.06-msg) MSG::RMA::XRF Time::Local (1.07)		

Appendix 4

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES

SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and an idea of what it does.
Copyright (C) yyyy  name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w'.  This is free software, and you are welcome
to redistribute it under certain conditions; type `show c'
for details.
```

The hypothetical commands `show w'` and `show c'` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c' ; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision'
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

Appendix 5
Pilot Balloon Subsystem

Introduction

This system was divided in two modules, one for data capture to be executed connected to the theodolite and to be run in Microsoft Windows and other for data query using the WEB in CENICA's server.

Compilation and ordering of the existing databases.

CENICA delivered to Matias Software Group a CD with data grouped in three directories: C1 (2000 campaign), C2 (2001 campaign), C3 (2002 campaign), KyToon01 (Tests with this type of balloon) and MitcamDF.

The delivered data came in several formats all derived from the one used for TD3 communications. The only observations that had unrecoverable damage were C1/01021213.AZC and C2/00071213.UNI, because the theodolite suffered some kind of malfunction which made it report incoherent numbers. We recommend to put attention on future theodolite failures because it may have a electronics programming error, although due to the low incidence level, it seems like a problem that will appear on very few occasions in practice.

Once the data was normalized in a uniform format, they were uploaded to the SINAICA data base, at the globo.obs table, with the following structure:

```
 Column  |            Type             | Modifiers
---------+-----------------------------+--------------
 fecha   | timestamp without time zone |
 lugar   | text                        |
 comm    | text                        |
 interv  | integer                     | not null
 nmuest  | integer                     | not null
 func    | text                        |
 azimuth | real[]                      | not null
 elev    | real[]                      | not null
 visible | boolean[]                   |
 discz   | real[]                      |

Check restrictions:
```

```
    "$1" CHECK (array_upper(elev, 1) >= nmuest AND array_upper(azimuth, 1) >=
nmuest)
    "obs_azimuth" CHECK ((0::double precision <= ALL (azimuth)) AND
(360::double precision >= ALL (azimuth)))
    "obs_elev" CHECK ((0::double precision <= ALL (elev)) AND (360::double
precision >= ALL (elev)))
    "obs_interv" CHECK (interv > 0)
    "obs_nmuest" CHECK (nmuest >= 2)
```

Of special interest is to mention that the campaign to which a given observation belongs is given at the comm field.

Theodolite-database interface.

We agreed to create an interface that worked under the Win-32 platform, to reduce the requirements on the laptops that would run the download. The program was made in Perl and comes inside a Windows setup program with the companion CD, along with documentation, source code and specifications.

To achieve the communication with the theodolite, we did some research from the documentation of the original download device and a serial prototyping tool, resulting an ex-profeso intermediate cable to connect the theodolite with the computer, using the original theodolite-download device as the only intermediary.

Wind vectors generation.

After a small trigonometric analysis, we reached the following conclusions:

Let A be the azimuth angle, E the elevation angle recorded by the theodolite, T the associated time by the given record and z(t) the estimated height function against time,

if H is the magnitude of the projection of the (A, E, z(T)) vector to the z=0 plane, then $H = z(T) / \tan E$

and finally so, $X = H (\sin A)$, $Y = -H (\cos A)$.

Although the initial balloon position datum is missing, this is irrelevant in principle, specially because what is being looked for is the derivative, or deltas, of the balloon positions, where the initial constant is lost. By simple differential calculus, we deduce that the values of the vectors resulting from the balloon position deltas are maintained, even when the initial constant approaches zero, and so it ends outside the problem scope.

This solution ended up registered as a data base function, called **globo.xyz** , whose declaration is defined in the **SINAICA_def.sql** file in the companion compact disc. Also, for additional convenience, we included a ***globo.winds*** view, which delivers the wind vectors already calculated, available through SQL.

Analysis and reporting application.

A Web page which presents the available observations with their data was developed, along with an SVG graph generator of the observation paths and wind fields.