MICRO-COMPUTER ANALYSIS OF AQUATIC EFFECTS DUE TO ACID PRECIPITATION: PART 1. DEVELOPMENT OF USER SOFTWARE.

D. A. Swayne[1], A. S. Fraser[2], D.C.L. Lam[2], L. White[1].

[1]Department of Computing and Information Science, University of Guelph, Guelph, Ontario N1G 2W1
and
[2]National Water Research Institute (Canada), 867 Lakeshore Road, Box 5050, Burlington, Ontario L7R 4A6.

86-6

Micro-Computer Analysis of Aquatic Effects due to Acid Precipitation: Part I. Development of User Software

D. A. Swayne[1]
A. S. Fraser[2]
D. C.-L. Lam[2]
L. White[1]

[1]Department of Computing and Information Science,
University of Guelph,
Guelph, Ontario, N1G2W1
and
[2]National Water Research Institute (Canada),
867 Lakeshore Road, Box 5050,
Burlington, Ontario, L7R4A6

ABSTRACT

This paper describes the development of software for a user workstation for the analysis of acid rain data.

The station is implemented for an IBM Personal Computer Model XT or AT. The workstation is used to build a local database abstracted by the user from various environmental data collections concerned with lake acidification. The query language is based on a pictorial representation of the geographical area under analysis, arranged in an hierarchical fashion, to allow the user to examine the database with a geographical perspective. The data are organized in a flexible fashion for the user, and made available for browsing and simple statistics in a spreadsheet fashion.

The aim of the station is to remove the programming requirement from the user, and to allow data management using only the menus, schema names, and user defined functions.

The utility of automatic reasoning systems to support the workstation is also under investigation.

Support from Environment Canada through Department of Supply and Services contracts for design and implementation are gratefully acknowledged.
A. S. F. is in charge of the overall project as scientific authority and a major contributor in the software development. D. C.-L. L. is a contributor to the project in both the software conceptual design and the scientific application. D. A. S. has managed the prototyping of various components of the system and supervised a part of the detail design. L. W. is the principal designer and programmer of the user interface. Peter Wong and John Storey from the University of Guelph have also made significant contributions. Peter wrote a major portion of the DBMS program, and John has undertaken an extensive revision of the main interface program, and the implementation of many of the system functions.

RÉSUMÉ

Ce document décrit la mise au point d'un logiciel d'analyse de données sur les pluies acides, destiné à être utilisé à un poste de travail individuel.

Le poste de travail est conçu pour être compatible avec un ordinateur personnel IBM modèle XT ou AT. Les utilisateurs s'en serviront pour extraire des données sur des régions particulières à partir des différents fichiers sur l'acidification des lacs. Le langage d'interrogation est basé sur une représentation graphique des régions géographiques classées par ordre d'importance, ce qui permet à l'utilisateur de consulter la base de données selon les régions. L'organisation des données offre une grande souplesse : on peut passer en revue les fichiers en entier ou obtenir des statistiques simplifiées disposées en tableaux.

Le but de ce projet est d'éliminer la programmation et de permettre une gestion facile des données en faisant appel à des menus, des noms de programmes et des fonctions dédiées à l'utilisateur.

Nous sommes également en train d'évaluer l'utilité de combiner des systèmes de raisonnement automatique au poste de travail.

1. Introduction.

Early in 1985, the authors of this paper began series of discussions to develop an understanding of the desirable features of a workstation for an investigator of lake acidification. This workstation was to provide a comfortable environment for access, retrieval, and analysis from large databases which contain measurement of water quality, precipitation, and soil data. It would also assist in the construction of simulations of the effect of parameter changes in the existing environmental chemistry, and in a later stage it would provide reasoning tools for the selection of important factors affecting lake quality in a particular watershed. The first application of this system would be to the Quebec region, a subject of another paper to this conference.

The system in its final form would have three levels of decision support. The first of these would be simple data analysis. The analysis of water chemistry data, for example, would involve a statistical library, calculation of certain well-known parameters involving ions active in the acidification process, and manipulation of the quantity of inputs (e.g. sulfates) in the system. At the next level, time-simulation would be supported, so that effects not currently observable could be predicted. At the third level, we planned the implementation of an expert system to investigate both the relative importance of various factors in lake acidification and the usefulness of remedial measures in particular instances of acidification. This paper describes the progress at the time of writing. Many of the features of the RAISON Micro system, as it is called, are currently in a preliminary prototype stage. Only the first level has taken on the polish of a nearly-finished product. The remaining levels will be integrated into the workstation when they pass from prototype to satisfactory working stage.

2. System Environment.

The team, supported by contract funding from Environment Canada, proceeded in July-August 1985. The computing environment consisted of :

Hardware	
	-one PC/AT
	-one PC/XT
Components	-color graphics
	-mouse
	-hard disk
	-full memory(640k)
Software	
	-Lattice C
	-Microsoft C
	-Halo Graphics
	-terminal emulation with file transfer

-statistical package
-BASICA (where necessary)
-LOGO (initially),
and Lisp (eventually)
-elements of a discrete
simulation package

Source Data

-Digitized Maps
-Large databases
such as NAQUADAT at
NWRI.

The medium of the C programming language was chosen for its availability on other developmental resources at Guelph, its reliability, readability, and the availability of student programmers with graphics and database experience using C as the implementation language. Halo graphics has provided a reasonable and robust environment for the development of visually important segments of the project, and both C and BASIC support for the programming tasks.

File transfer and statistical analysis functions were not to be developed by the project, as the results would not be comparable with commercially available products.

BASICA would be used where necessary, but would be replaced eventually by C source code.

Initial prototyping for expert system functions would be accomplished by the list processing functions contained in LOGO. This choice was made of necessity initially, since funding was not available for a Lisp interpreter.

3. System Structure.

The system programs, running under DOS, are arranged in an hierarchical fashion. The two main system programs are called **DBMS** and **INTERFACE**.

DBMS.

At the top level, **DBMS** gives the user a choice of:

-host retrieval,
-create or edit schema,
-data record manipulation,
-remote login to mainframe,
-data loading into local database,
-access to local data **INTERFACE**.

The remote login is eventually to be equipped with built-in scripts to extract data from commonly used databases without requiring user intervention. Data retrieval uses a file transfer utility to ensure a reasonable reliability level in data transmission.

A data **schema** or logical relation between data elements may be either constructed by the user, or generated from header records provided by the mainframe database. Missing data values are flagged appropriately. Data is organized by sampling station and calendar date usually. Data fields are named for relevant physical parameters such as flow volume, Na^+, Ka^+, etc. Standards for encoding these data are kept in table form, so that pickup from the incoming data file can generate a schema by parsing the header record and scanning each data record for relative position of the data item. Some data editing remains to be inserted, such as limit-of-reasonableness, and item-count. Input from the file transfer utility is further filtered as it is appended to the local database, to trap spurious characters or records which might corrupt the local database.

At the next level, or possibly running in an independent mode, is the **INTERFACE** system.

INTERFACE.

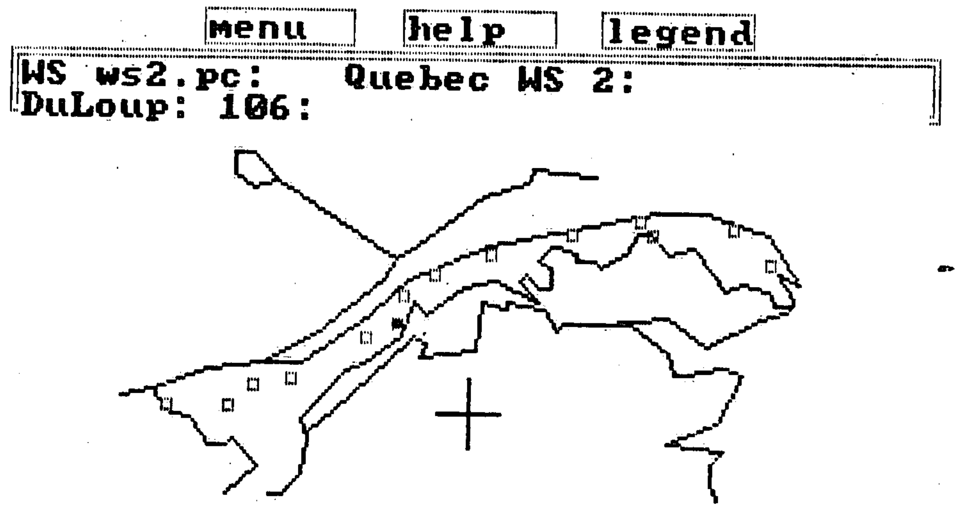
The interface programs interact with the user through a pull-down menu. Many commands have been implemented using keys ("m" for menu being one example). All of the menus and functions are implemented through table files which allow the programs to increase in functionality without re-compiling the main driver routines at each step. A user may select to browse through watershed data, or optionally to extract new data from the local database. The data query language is based on a map of the main area in question (e.g. Canada, Quebec). The user is supplied with a cross-hair in the center of the map, and the means to move it, either with arrow keys or a mouse. Each **region** contained within the main area (e.g Region 4, involving the Ottawa River and its Quebec province tributaries) is selectable by pointing at it with the cross-hair and keying the **select**. (This is the center of the arrow keypad or the mouse button). The map now is replaced by a larger scale map of the region. Should the region be subdivided into several **watershed** subdivisions, the selection process is repeated. All of this is driven by tables and/or the file system. At each level the map is represented by a file consisting of:

-coordinates of points to be joined to form map outlines.
-line terminators
-map terminator
-coordinates of map-finder/station (small boxes on map),

-either	or
-printname of next map	-printname of station,
-filename of next map down in hierarchy,	-database code name of sampling station selected,

-and (where applicable) the watershed area in km ^{2}.

Figure. Pictorial representation of data query.



In order to facilitate prototyping of the system, a simple map editor was developed and included in the menu. Lines may be drawn or traced and stations may be added.

Various functions (color selection and so forth) are also implemented in the menu).

The user of the local database selects a date range, and points at the watershed station for which the data is to be displayed. The information is retrieved using the date range specified and the name of the watershed as stored in the map file. This data is transferred into a table from which a **spreadsheet** is displayed on the screen. The map and the spreadsheet coexist and may replace each other on the screen.

Spreadsheet.

The spreadsheet program was initially written using mostly the graphics package, but in its latest version it is displayed on the monochrome monitor of the PC simultaneously with the map on the color graphics device. A detailed description of the spreadsheet layout is found in [Fra86], including a sample of the display. Our discussion will center around design considerations for the spreadsheet. Data display is column oriented, and the size of the data matrix is assumed to be considerably larger than the screen display. cursor movement is used to space horizontally and vertically through the data.

Rows of data are arranged in increasing order of metric date, yy/mm/dd.

Column labels are derived from the schema description, and the names reflect encoded information about the data, stored in a schema file. references to a particular column e.g. *Na* are through its symbolic name.

Built-in functions include conversions from milligrams/liter (mg/l) to milli-mole-equivalents (mmeq), and from -log $_{10}$(concentration) to (concentration). Inverse functions are of course included. A state vector containing the units of measurement of the columnar data is used to ensure integrity of the columns. Other functions which are being added either as primitives or in the user library compare the units required in the calculation to those present in the state vector so that the integrity of the function is maintained while at the same time side-effects (such as the alteration of the data in a column not undergoing computation) are minimized.

New columns may be created by calculation, and these columns may also overwrite existing columns. A single entry in the spreadsheet may be overwritten by means of a primitive Boolean function which censors the result of the calculation over a whole column, restricting the deposition of the result to the row in which the change is to be made. Thus the complexity of the instruction set is kept to a minimum - operations involve only column indices, except for a Boolean vector for censoring the result. Partially implemented is a history mechanism and recovery of a previous spreadsheet.

Functions are stored in character strings for which aliases or labels may be attached. The function library is thus capable of generating compound functions, although the implementation of this facility at the time of writing is as yet incomplete.

Other Features.

A reasoning/learning facility is planned (not yet implemented) for the system. A primitive prototype has been tested off-line using the LOGO programming system's list processing facility. While this is an essential feature of the finished product, the urgent details of the user interface and the local database have occupied most of the time spent.

Ancillary Systems.

INTERFACE uses a terminal emulator with a file-transfer capability. PC-Talk is the one currently in use, chiefly because it is in the public domain, as is the statistics package - EPISTAT. A Lotus-123 interface exists, and we expect to be able to convert our data and (if needed) add a line to the INTERFACE menu.

4. Development Plan.

We have used rapid prototyping throughout. A rudimentary DBMS and INTERFACE were running within two months of the start of the project. This is a product coded almost entirely by senior undergraduate students from the University of Guelph, who often leave on short notice when permanent employment is offered, and there are some inherent risks in prototyping if the documentation trails development by too long a time. This has not become a problem, partly because of the integrity of the individuals involved, and partly because of the knowledge gained from the prototyping phases by the principals who remain working on the project. No special tools, other than C and Halo Graphics have been used in the project, and the public domain software already mentioned.

5. Conclusions.

This report outlines the ongoing development of a user workstation for a data-rich scientific environment. The product - RAISON Micro - seeks to give the user a rigorous yet friendly environment for the manipulation of acid rain data. Where practical, off-the-shelf software is being used, but a large portion of the code is written in such a way as to make transportation onto a new generation of machine feasible, and to modify the existing environment with a minimum of effort.

The application of the product is the substance of another paper [Fra86].

REFERENCES

[Fra86]

A. S. Fraser, D. C.-L. Lam, D. A. Swayne, and L. White, Micro-Computer Analysis of Aquatic Effects due to Acid Precipitation: Part II. Applications to Eastern Canada Data - this Conference.