Reprinted from

Computers and electronics in agriculture

Computers and Electronics in Agriculture 14 (1996) 301–315

Developing transportable agricultural decision support systems: Part 2. An example

Gianni Jacucci [a], Mark Foy [a,*], Carl Uhrik [b]

[a] *Laboratorio di Ingegneria Informatica (LII), Dipartimento di Informatica e Studi Aziendali (DISA), Università degli Studi di Trento, Via Fortunato Zeni 8, 38068 Rovereto (TN), Italy*
[b] *University of Phoenix, DTP Faculty, 7800 E. Dorado Pl., Englewood, CO 80111, USA*

ELSEVIER

Computers and electronics in agriculture

GENERAL INFORMATION

Audience: All scientists, engineers, environmentalists and economists concerned with applications of computers and electronics to agriculture and related industries.

Aims and Scope: *Computers and Electronics in Agriculture* provides international coverage of advances in the application of computer hardware, software and electronic instrumentation and control systems to agriculture, forestry and related industries. The latter include horticulture (in both its food and amenity aspects), forest products, aquaculture, animal/livestock science, veterinary medicine and food processing.

The journal publishes original papers, reviews, applications notes and book reviews on topics including computerized decision-support aids (e.g., expert systems and simulation models) pertaining to any aspect of the aforementioned industries; electronic monitoring or control of any aspect of livestock/crop production (e.g. soil and water, environment, growth, health, waste products) and post-harvest operations (such as drying, storage, production assessment, trimming and dissection of plant and animal material). Relevant areas of technology include artificial intelligence, sensors, machine vision, robotics and simulation modelling.

Publication information: *Computers and Electronics in Agriculture* (ISSN 0168-1699). For 1996 volumes 14 and 15 are scheduled for publication. Subscription prices are available upon request from the Publisher. Subscriptions are accepted on a prepaid basis only and are entered on a calendar year basis. Issues are sent by surface mail except to the following countries where Air delivery via SAL mail is ensured: Argentina, Australia, Brazil, Canada, Hong Kong, India, Israel, Japan, Malaysia, Mexico, New Zealand, Pakistan, PR China, Singapore, South Africa, South Korea, Taiwan, Thailand, USA. For all other countries airmail rates are available upon request. Claims for missing issues should be made within six months of our publication (mailing) date. Please address all your requests regarding orders and subscription queries to: Elsevier Science B.V., Journal Department, P.O. Box 211, 1000 AE Amsterdam, The Netherlands, tel. (+31-20)4853642, fax (+31-20)4853598.

In the USA and Canada: For further information on this and other Elsevier journals please contact: Elsevier Science Inc., Journal Information Center, 655 Avenue of the Americas, New York, NY 10010, USA. Tel. (212) 6333750; fax (212) 633 3764; telex 420-643 AEP UI.

US mailing notice: *Computers and Electronics in Agriculture* (ISSN 0168-1699) is published monthly except in February, May, July and October (total 8 issues) by Elsevier Science (Molenwerf 1, P.O. Box 211, 1000 AE Amsterdam). Annual subscription price in the USA is US$500.00 (valid in North, Central and South America), including air speed delivery. Second class postage paid at Jamaica NY 11431.

USA POSTMASTER: Send address changes to Computers and Electronics in Agriculture (ISSN 0168-1699), Publications Expediting Inc., 200 Meacham Avenue, Elmont, NY 11003.

AIRFREIGHT AND MAILING in the USA by Publications Expediting Inc, 200 Meacham Avenue, Elmont, NY 11003.

Back volumes: Please contact the Publisher

Computers and electronics in agriculture

ELSEVIER Computers and Electronics in Agriculture 14 (1996) 301–315

Developing transportable agricultural decision support systems: Part 2. An example

Gianni Jacucci [a], Mark Foy [a,*], Carl Uhrik [b]

[a] Laboratorio di Ingegneria Informatica (LII), Dipartimento di Informatica e Studi Aziendali (DISA), Università degli Studi di Trento, Via Fortunato Zeni 8, 38068 Rovereto (TN), Italy
[b] University of Phoenix, DTP Faculty, 7800 E. Dorado Pl., Englewood, CO 80111, USA

Accepted 7 September 1995

Abstract

Information in the area of developing transportable decision support systems (DSSs) (agricultural or otherwise) has been scarce. Therefore, developers of DSSs have had some difficulties in constructing DSSs which could be widely used. In the first part of this two-part paper series, a conceptual framework was introduced which proposes methods for making DSSs transportable. This framework serves as a checklist and includes recommendations about general implementation, user interaction, data management, and model aspects. To illustrate the implementation of these recommendations, the development of a DSS within a project called SYBIL is discussed in detail. Therein, portable and public domain tools have been used to build the DSS with a graphical user interface (GUI) which satisfies the general implementation aspects of the framework. Further, the development of a flexible data management system has been essential in the project so that different types of data can be easily handled without changing the models. In order to allow models embedded within the DSS to be successfully transported between regions, a novel artificial intelligence (AI) adaptation methodology was implemented. The main component of this adaptation methodology is a genetic algorithm (GA), an AI search technique. By linking a GA to an agricultural model, the model becomes more robust because the model is able to *adapt* to the region in which it is being used. Overall, by using the framework criteria to select/create these tools/components, the DSS has been made easy to transport and disseminate.

Keywords: Decision support system; Modeling; Adaptation; Portability; Genetic algorithm

1. Introduction

In agriculture, decision support systems (DSSs) often cannot be transported between regions due to problems associated with diverse hardware platforms, data

* Corresponding author. Present address: 1818 19th Street No. 309, Boulder, CO 80302-5563, USA.

0168-1699/96/$15.00
SSDI 0168-1699(95)00039-9

compatibility, and accuracy. These can be classified as problems of *DSS/model technology transfer*. One solution to these problems is to build transportable DSSs which can be moved between regions, thereby allowing wide DSS dissemination and utilization. Towards this goal, a conceptual framework for developing transportable DSSs has been proposed (see part one of this two-part paper series), and a DSS has been produced by following these guidelines.

A DSS (which addresses grape and apple management) developed under a European Commission (EC) project called SYBIL has utilized the framework's guidelines. The models in this DSS are designed to help growers determine when it is necessary to spray against certain pests. In creating this DSS, many design decisions have been made in the context of the framework's recommendations, and two major components have been designed and implemented to satisfy the framework's data management and model aspects:

(1) a *flexible meteorological data management system* (called the Meteo C Library) — this library provides robust data management allowing many different types of data sources to be managed;

(2) an *agricultural model tuned by a genetic algorithm (GA)* (called an AGMOD-GA) — such a component is created by applying a model adaptation methodology to permit a model to work in a new location.

Overall, the framework has been very useful in creating a DSS that can be easily disseminated throughout the European Union (EU).

First, in Section 2, the conceptual framework (presented in detail in the first part of this two-part paper series) will be outlined. In Section 3, the application of the framework's criteria will be described with respect to a DSS within the SYBIL project. Lastly, conclusions are given.

2. Conceptual framework overview

In a nutshell, the conceptual framework is broken into four aspects:

(1) *general implementation aspects* — emphasizing low-cost portability;

(2) *user interaction aspects* — focusing on ease-of-use and flexibility;

(3) *data management aspects* — highlighting the importance for the DSS to handle many types of data;

(4) *model aspects* — stressing that models should be adaptable and that mechanisms should be in place to help the user perform this adaptation.

3. Applying the framework's recommendations within a DSS

Following the conceptual framework, a DSS has been created within EC Project SYBIL. This DSS has utilized tools and techniques which conform with the recommendations of this framework; this is illustrated in Fig. 1. The following is a discussion of how these framework elements have been applied and implemented within this particular DSS.

First, to put this discussion in context, an introduction to project SYBIL will be given. Subsequently, SYBIL will be examined with respect to the general

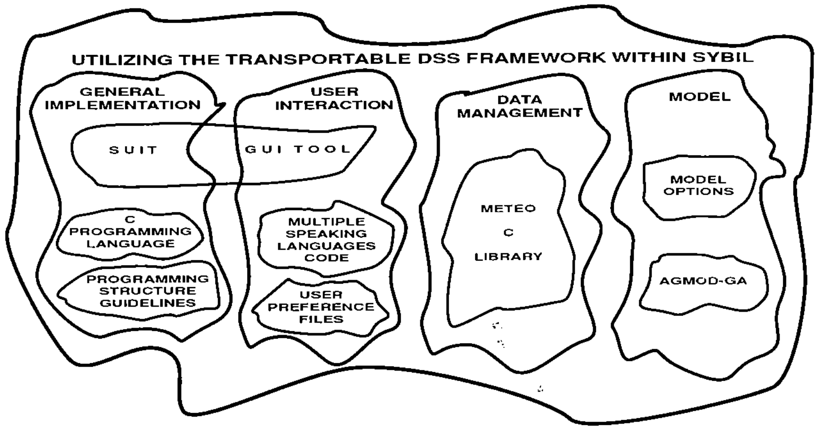

Fig. 1. Utilizing the transportable DSS framework within SYBIL.

implementation, user interaction, data management, and model aspects of the framework for transportability.

3.1. Description of Project SYBIL

EC Project SYBIL (consisting of five partners from four countries) involves the implementation of computerized DSSs to assist farmers in intelligently governing their crops such that environmental impact is reduced and economic returns are increased. Existing agro-meteorological computer models from multiple sources are integrated into the portable, user-friendly DSSs designed to assess the risk of a crop to pest and fungus damage. By evaluating this risk, the farmer has the option to apply pesticides and fungicides only when needed and avoid using these, often environmentally damaging, chemicals blindly on a regular basis or when the risk of pest and fungus damage is small. This evaluation has the potential to save the farmer both time and money because expensive chemicals will not be applied without benefit to the crop.

The DSS described here (which has been developed using the framework previously discussed) is targeted to grape and apple growers. Fig. 2 displays the first screen of this DSS, and Fig. 3 displays the structure of this DSS. Following the framework previously outlined, the techniques used and components created specifically for this DSS will be described in the following sections.

3.2. General implementation

The application of the general implementation (public domain tools, portable tools, structured programming) and two of the user interaction aspects of this framework [graphical user interface (GUI) and inclusion of multiple speaking languages] form the base of this DSS.

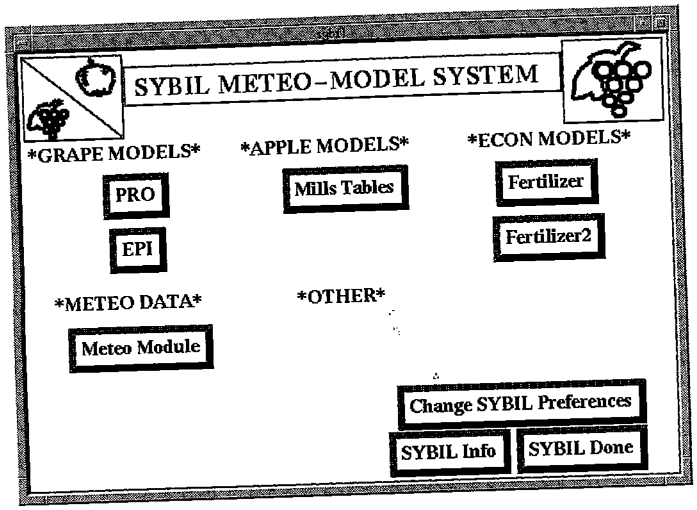

Fig. 2. The DSS's main screen.

To satisfy the *portability* and *public domain tools* requirements (so all participants could be involved without additional expense on their part) plus the *GUI* requirement, a tool/library called *SUIT* (Simple User Interface Toolkit) was selected (Conway and the University of Virginia, 1992). The focus of this library is to allow the quick development of a GUI as a "front-end" to a computer application, but two general requirements were also satisfied by the choice of this tool/library as follows:

(a) the library is available for four different platforms (three of which are commonly found in agricultural computing environments), thereby making the final system *portable* between four different platforms (IBM-PCs with DOS, IBM-PCs with Microsoft Windows Version 3.1, Macintosh machines, and UNIX workstations with X-Windows);

(b) the library is written in the C programming language, therefore the system is made further *portable*;

(c) the library/tool is in the *public domain* and can be obtained freely by most groups.

DSS portability had implications for *program structure* because if different partners are to modify the system, certain conventions should be agreed upon for program development. The program structure was controlled in a straight-forward way through the development of a short document establishing guidelines for the structure of the system (e.g., file names, procedure names, indenting rules, etc.).

3.3. User interaction

As mentioned above, as well as meeting the general recommendations for a portable public domain tool, SUIT satisfies the *GUI* requirement specified by the user interaction aspects of the framework. However, there are two additional user

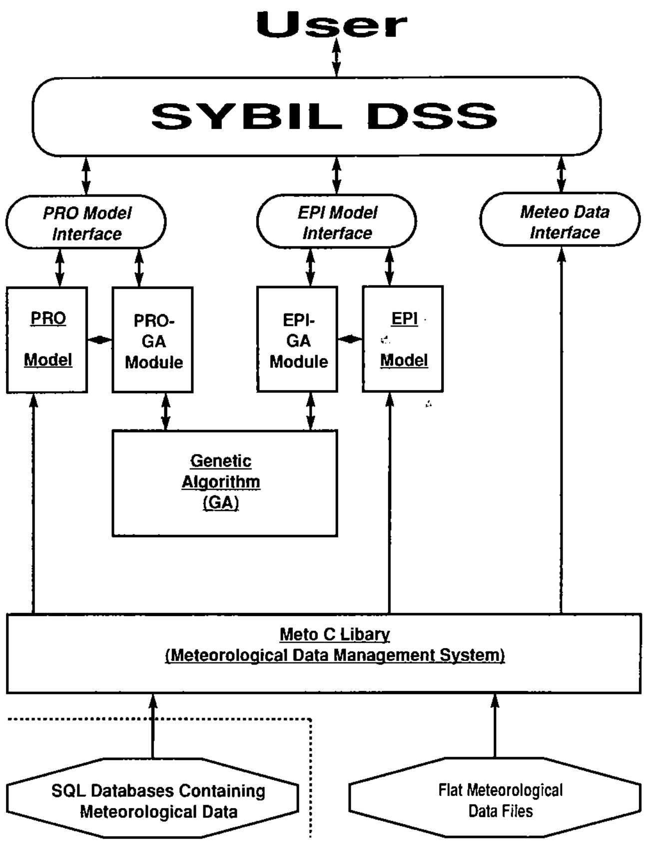

Fig. 3. Structure of this DSS.

interaction recommendations; one regarding multiple speaking languages and one regarding custom-tailoring problem-solving scenarios to user preferences, both of which are considered below.

3.3.1. A flexible strategy for multiple speaking languages

The inclusion of multiple spoken languages in this DSS has been particularly important because this DSS is developed within an EC project with partners coming from four EU countries. Therefore, this DSS needs the ability to present messages in all four *speaking languages* of the participating EU countries (i.e., Italian, German, French, and Danish), plus the English language. This was implemented by creating separate code files (i.e., language string files) for each screen, and placing all text that is to be presented to the user in these separate files as a sequence of assignments to string variables for which the different languages correspond to different sub-strings. In this way, all the text strings are separated from the rest of the code, and text can be translated and placed in close proximity to the versions of the same text in other languages. When text is to be displayed on the screen, the string corresponding to the user's first language preference (e.g., Italian) is queried, and if it exists, it is used on the screen. If the string is not available (i.e., empty), the system tries to use the string corresponding to the user's second language (e.g., French), but if it is not available, then the default language (which for this DSS was set to English) is used, as the default language string is always available.

3.3.2. User specific decision scenarios

To allow users to specify their decision scenarios, user preference files have been implemented within this DSS. These preference files include: one for global DSS preferences, one for meteorological data preferences, one main preference file for each model, and other preference files for sub-components like instantiations of adaptation components (such as the AGMOD-GA instantiations, which will be discussed later). These allow the user to specify what types of inputs to use for certain calculations, model calculation preferences, and how outputs should be displayed. Individually customized versions of these preference files can be created, and each user can have her/his own set or multiple sets for different purposes.

To address the need to transform the output of the models into advice which the DSS operator can use to make a decision, a feature for obtaining expanded explanations has been included if the user desires these explanations. These explanations give information about what the models have discovered, and how this could be related to the real world. For instance, in one of the models, possible interpretations are given to the risk factors produced by the model (risk factors indicating certain stages of a fungus infection). These were not furnished in the original model, but by providing this information, the user is better equipped to usefully interpret the model output and use it to make decisions.

3.4. Data management

To satisfy the framework's requirement to provide robust data management within the DSS, a flexible and modular data management component was implemented. Because this DSS deals almost exclusively with meteorological data, this data management component focuses on meteorological data and is called the *Meteo C Library*. This library allows seamless access to meteorological data in any type of ASCII file or from SQL databases, thereby providing "generic" access (i.e., multiple access methods) to meteorological data by all models within a DSS.

Fig. 4 illustrates the structure of this Meteo C Library, and how models interact with the library.

Interaction between a model and the Meteo C Library begins when the user inserts a user preference file into a model (i.e., stipulates which user preference file the model should use). This file specifies the data source (an ASCII file or an SQL database), the year for model calculation, which meteorological station data should be used (*stat#*), and other user preferences regarding the model calculation. After this file is inserted, and loading of data or model calculation is initiated, the Meteo C Library is called to initialize the meteorological data connection. Towards

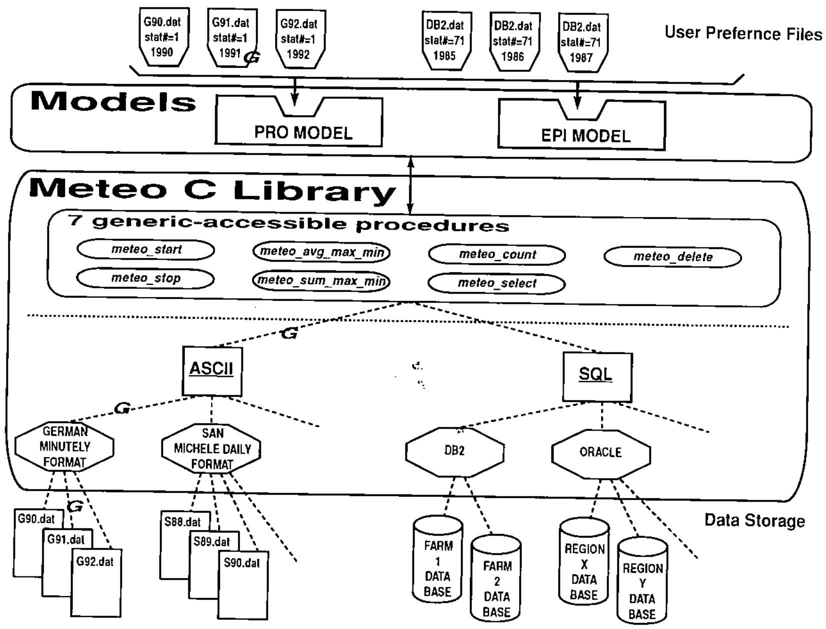

Fig. 4. Structure of the Meteo C Library.

the goal of modular, extensible access to data, seven "generic" library procedures have been made accessible to models. No matter which model or what type of data is being used, the model first calls the *meteo–start* procedure, passing to this procedure the data source file name, year, and station number. Upon the first call of this start procedure, no active data connections exist (i.e., in Fig. 4 all dashed links below the seven accessible "generic" procedures are not formally connected). The start procedure then calls other procedures in the library that handle access to more specific types of data. These more deeply embedded procedures determine if the data source is a flat file (ASCII) or a database (SQL), and furthermore, the format type (e.g., German Minutely Format or San Michele Daily Format for ASCII sources, or Oracle- or DB2-type databases for SQL). This library does this by directly examining the "header" of the data source where keywords specifying type and format are placed. Each type of data has its own keyword which must be "registered" into the Meteo C Library. Using the keywords read from the data source, connections to the appropriate internal library procedures are made (and remembered) so that subsequent calls to the library (e.g., a call to *meteo–avg–max–min*) will use the source of data specified in the last *meteo–start* call. When the model is finished with the data source, it calls *meteo–stop*, and all connections are broken.

Fig. 4 also gives an example of the data connection made when a user inserts a user preference file (shown in the figure as the file at the top, second from the left) which: (a) points to the *G90.dat* file, (b) specifies *1991* as the year for calculation, and (c) stipulates that data from station number 1 (*stat#=1*) be used. When model calculation is initiated with these parameters, links labeled *G* in the figure are "connected" and all subsequent requests for data use these links to supply data to the model.

In addition to providing "generic data access" to the library and dynamic sensing of the physical structure of data being used, the Meteo C Library creates a standardized internal representation for the values of all possible variables, which is needed for a seamless access to data stored in granularities or units differing according to source. This representation is maintained as a list of variables (e.g., date, meteo station number, temperature, humidity, etc.) and characteristics of their values (units, print format, etc.). This list can be expanded by "registering" new variables into the library as necessary if new data sources are added which contain data not yet present in the variables list. Thus, each variable's internal units are standardized to some agreed units (e.g., temperature in degrees Celsius), and whenever data is read into the library, the data will be converted (if necessary) from the units stored in the data source, to the units used by the library. This ensures that data sent from the library to the models is of one particular type, and conversions do not have to be made in the model based on different data types. Moreover, because meteorological data is often found in many different frequencies (e.g., hourly, daily, etc.), the Meteo C Library handles frequency conversion from higher frequency to lower frequency (e.g., minutely to hourly) as needed, based on what data is requested from a model. This conversion is simply handled through averaging (as in the case of temperature) or summing (as in the case of rain) the more frequent values into the frequency or time period requested by the model. Overall, these features allow all data (no matter where they are from) to be stored in the same way inside the Meteo C Library, and only when reading the data from the data source into the library will it be necessary to perform conversions.

In general, with these types of data management flexibility, when the DSS is to be transported to different regions where data is maintained in different ways, neither *models* within the DSS nor *data* will need to be changed so that the data can be used within the DSS (e.g., it will not be necessary to change model code nor to re-format data within an ASCII file nor to move data from an SQL database to an ASCII file or vice versa). From the user's point of view, data is automatically converted within this library between the units and granularity used to the units and granularity required. Therefore, computer models (agricultural or other) can access meteorological data in a standardized way (i.e., using the same calls and methods), no matter what type of data is being used (minutely, hourly, daily, weekly, monthly, or yearly — SQL database or ASCII file).

Under this scenario, the only action required when transporting the DSS is to *register* the meteorological data source within the Meteo C Library. Since the library is an independent service provider, it can be expanded to include additional data sources, and this will not have any effect on the model code. That is, new data sources (ASCII file formats or SQL database types) can be added without disrupting the model code which requests meteorological data.

Overall, this library is a step towards establishing a common/standard meteorological data platform for the management of meteorological data within DSSs.

3.5. Model aspects

The framework prescribes two recommendations with respect to the model aspects. One recommendation is to include as many user relevant options and parameters as feasible so that the model is flexible to begin with. The other recommendation is to provide a mechanism for adapting model parameters to give accurate results when used with data from regions other than the one in which the model was developed and calibrated. The use of each of these recommendations in SYBIL is described below.

3.5.1. Flexible models

With regard to flexible models, model options were included in the main grape model within this DSS which is called the Plasmopara Risk Oppenheim (P.R.O.) model. The P.R.O. model is a life-cycle simulation model which addresses *Plasmopara viticola*, a fungus which attacks grape vines. The model helps a grape grower decide when to perform the first spraying against the fungus by predicting when the fungus has spread to a stage where it will begin to damage the crop. The model has been used in Germany to help grape growers reduce the amount of fungicide needed to control this fungus (Hill, 1989, 1990a, b, c, 1993; Hill et al., 1993).

The P.R.O. model was modified to allow only part of the calculations to be done, to have leaf wetness estimated if it was not available, and to have darkness hours determined in many different ways (from solar radiation, from constants set by the user, or from latitude).

For example, with respect to leaf wetness in the P.R.O. model, the following leaf wetness options (which are presented to the user via the screen shown in Fig. 5) are included:

(a) do not consider leaf wetness (i.e., leave out the calculations which require leaf wetness; this is possible in the P.R.O. model);

(b) consider leaf wetness using leaf wetness measurement present in the meteorological data;

(c) consider leaf wetness using rain measurements present in the meteorological data (i.e., estimate leaf wetness — e.g., if rain is greater than a user specified value, such as 0.1 mm, then assume the leaf wetness to be 100%; otherwise assume leaf wetness to be 0%);

(d) consider leaf wetness by prompting the user to enter whether there was no dew/light dew/heavy dew.

3.5.2. Model adaptation component

For satisfying the need for a component which will adapt models to particular locations, in the DSS discussed here, an artificial intelligence (AI) methodology has been developed and put into use. This methodology conforms to the requirement of

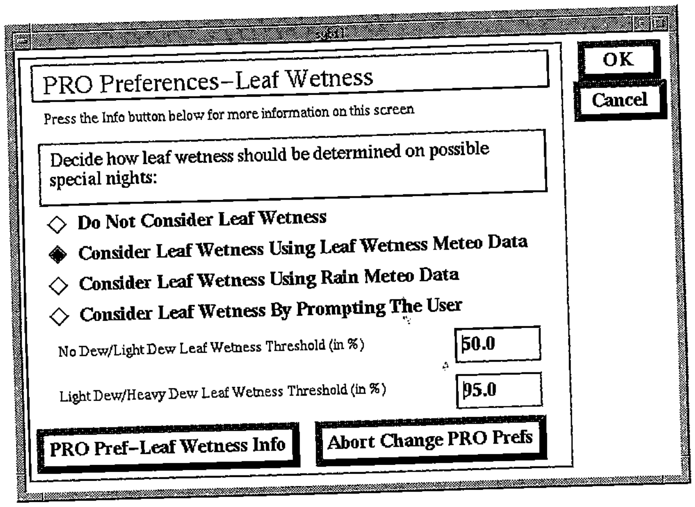

Fig. 5. Leaf wetness model option screen.

minimal user intervention (partially due to the AI technique utilized); therefore it has been found to be easy to use and does not require special expertise on the part of the user.

3.5.2.1. Overview of the adaptation methodology. By using historical data for the target location (geographical region), a model's parameters can be *tuned* so that the model is better adapted to the region in which the model is to be used than would be the case in taking the default parameters usually available for the region in which the model was developed. This tuning is facilitated by the creation of a component which can search for *good* parameter settings, which means finding settings for which the model produces results relatively consistent with the historical outcomes. Moreover, one hopes this tuning of parameters to historical data is an effective estimator for future events as well, thus producing relatively accurate predictive outputs for outcomes that have not yet been seen.

Additionally, it should be emphasized that this methodology is particularly appealing because it is not strictly empirical or analytical, but both. That is, this methodology does not perform a search to *fit the historical data* from a particular location into an empirical algorithm; rather it performs the search in a larger context, *fitting the model parameter settings* to a particular location. Therefore, the resulting instantiation of the adapted/localized agricultural model (with the new parameter settings inside) is as good (or as bad) as the original model. Consequently, if the model is biologically significant (e.g., if it simulates biological events), then this feature is not lost by this adaptation methodology since the model is used in the same form (i.e., *the structure of the model is left intact*), only the model parameter settings are changed.

3.5.2.2. Elements of the adaptation methodology. Generally stated, this methodology prescribes the utilization of:

(i) historical situation data;

(ii) historical outcome data;

(iii) the model; and

(iv) an intelligent search method (in this case, a GA, which is an AI search technique).

Historical situation data is the basic data required by the model in question. In the domain of agricultural models, this often includes meteorological data since this is frequently an important input to the model. In this methodology, the more historical situation data that is available, the better.

The presence of *historical outcome data* plays a large part in how accurately a model will be adapted using this methodology. This is due to the basic fact that models accept situations and computes outcomes. The historical outcome data will be used to *fit* the model parameter settings to the new region in question. Therefore, when constructing a component using the methodology described here, it must be possible to match model outputs to some combination of outcomes and/or events in the real world (and there must be one-to-one correspondence). For risk assessment models, historical outcome data regards the occurrence of fungus or pest problems in past years (epidemiological data); or for crop growth models, historical outcome data regards crop yield in past years.

In this methodology, the agricultural *model* is fundamental because it will be used to evaluate how well particular model parameter settings work in the given region (i.e., with the given historical data). In particular, the intelligent search method will repeatedly call upon the model as it proposes new model parameter settings that need to be evaluated. This methodology has the capability to address many types of agricultural models: risk assessment models, damage prediction models, and crop growth simulation models, to name a few.

The *intelligent search method* is an important part of this methodology because knowledge of the domain is often hard to codify, i.e., 'rules of thumb' are vague and difficult to construct, and this holds true for the domain of agricultural models. The selection of an intelligent search method can help to alleviate this difficulty. This is due to the fact that intelligent search methods do not rely on 'rules of thumb'.

The selection of the particular intelligent search method to be employed was made among several possible methods: hill-climbing, simulated annealing, and GAs. As with previous work in this area (Sequeira et al., 1994), GAs were selected as the most desirable method because:

(a) they can perform unbiased search;

(b) they make no assumptions about the search space (i.e., the search space does not have to be smooth or regular);

(c) they carry out a more effective search of an irregular, multi-dimensional space because they search from a population of points rather than a single point;

(d) their search is not random, but systematic (they utilize operators which are patterned after natural genetics); and

(e) they have been shown effective at finding optimal or near-optimal solutions to a wide variety of dynamic real-world problems (Grefenstette, 1985, 1987; Goldberg, 1989; Schaffer, 1989).

For a complete description of GAs and how they function, refer to Holland (1975) and Goldberg (1989).

3.5.2.3. Utilizing the adaptation methodology — the Agricultural Model-GA (AGMOD-GA). To allow a GA to search the space of an agricultural model's parameters, the GA uses the model as the *evaluation function*. Furthermore, the model uses the historical situation data (discussed earlier), and the GA additionally uses the historical outcome data (also discussed earlier) in combination with the output of the model. Whenever the GA wants to evaluate one instance of model parameter settings, the agricultural model is called, and the final outcome is returned through an objective function to the GA so that a "fitness" can be computed. This resulting general component is called an Agricultural Model-GA or an AGMOD-GA.

Fig. 6 illustrates an agricultural model and a GA linked to form an AGMOD-GA.

The function of the AGMOD-GA is to find near-optimal model parameter settings for the given desired behavior (i.e., matching the given historical outcome data). There are three main steps involved in the execution of a typical AGMOD-GA. First, the agricultural model and the GA are initialized. With the simple GA implemented in this case, an entirely random initial set (i.e., population) of parameter settings is generated. This has the effect of starting the search in a number of different random points in the space. A collection of random starting points does not have a negative effect on GA performance because a GA searches from many different points at the same time, not just from one point. The second step in a

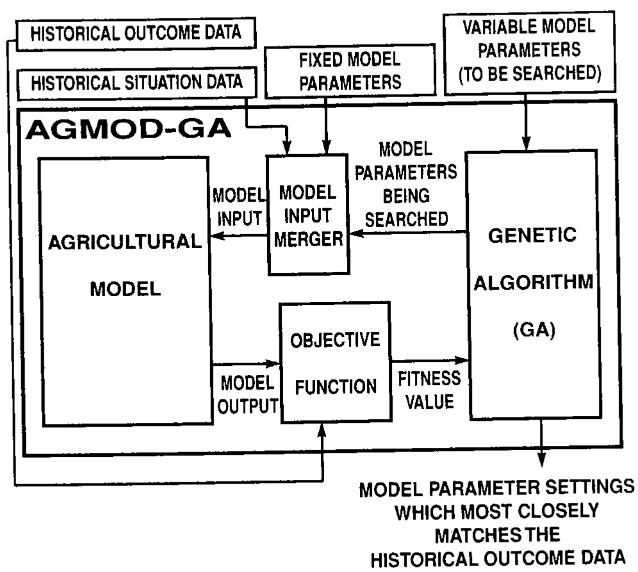

Fig. 6. Structure of an AGMOD-GA.

typical AGMOD-GA is the fitness computation (i.e., evaluation of each population member's worth). This involves taking each GA population member and executing one or more model executions using the model parameter settings represented by this member. These model executions utilize the user-provided historical situation data, with one execution initiated for each one of these sets of data. The outcomes from these model executions are compared against the user-provided historical outcome data (which is the target). The further the model outputs are from the historical outcome data, the lower the fitness, and inversely, the closer the model outputs are, the higher the fitness. This fitness evaluation step is executed many times because new population members are continually being generated by the GA. Fitness evaluation is usually continued until the GA has converged on a suitable optimal or quasi-optimal solution. The third and last step is the evolution of the GA population. This involves applying operations to the population members. The three operators used in a typical GA are reproduction, crossover, and mutation. They act by treating the GA bit strings (which represent model parameters) in a way analogous to the evolution of chromosomes in genetics (Goldberg, 1989).

The preliminary testing of an AGMOD-GA component constructed in this DSS framework (a module called the PRO-GA which uses the P.R.O. model mentioned previously) is promising and is explained in more detail in Jacucci et al. (1994, 1995).

4. Conclusions

To describe the usage of a previously described conceptual framework for developing transportable DSSs (presented in detail in part one of this two-part paper series), the creation of a DSS (within EC Project SYBIL) is described. In particular, the application of the framework's recommendations are illustrated through a detailed description of the DSS components and structure. All aspects of the framework are discussed in this context. Due to the desire and need for wide distribution of the final system, portability and adaptability were of particular interest.

The first step in applying the framework's general criteria was to consider the computer tools in the context of the models which were to be included and the targeted users. Second, human–computer interaction was considered so that the resulting system would be easily used. Third, data management techniques were designed such that different types of data could easily be used within the DSS. This involved the implementation of one of the two major components of the DSS: a robust data management system called the Meteo C Library. After these base ideas were in place within the DSS, the models could be included. This involved considerations about model options and how models can be adapted to run in different regions. From these considerations was included the second of the two major components: a model adaptation methodology/component called the AGMOD-GA.

Upon close examination of the final DSS, the system seems to be very transportable due to the above described actions. In particular, it appears as if many different groups could use the DSS because the system (and therefore the included models) can handle many different types of data (formatted in different ways) without having to change the models. Because of this feature in particular, as well as the fact that the system can be run on two common computer platforms, the collaborating agricultural agencies have expressed interest in the DSS, but at this time no extensive testing has yet been performed with these agencies.

Acknowledgments

Funding has been provided for the SYBIL project (project number PL 900615) by the EC CAMAR Program (Competitiveness in Agriculture). The authors wish to thank Dr. Val Reilly, SYBIL project supervisor from the EC, for his support and encouragement.

Furthermore, the authors would like to thank all EC Project SYBIL partners: Technische Universität München, Weihenstephan (Technical University of Munich) (TUM) (D); Bayerischen Staatsministerium für Ernährung, Landwirtschaft und Forsten (BALIS) (D); Association de Coordination Technique Agricole (ACTA) (F); and Danish Institute of Plant and Soil Science (DIPSS) (DK). In particular, thanks go Dr. Georg Wendl and Dr. Georg Froehlich at TUM; Institut und Bayerische Landesanstalt für Landtechnik (Institute of Agricultural Engineering) in Freising, Germany for their ideas, work, and collaboration on the meteorological data management component (Meteo C Library).

References

Conway, M.J. and the University of Virginia (1992) The SUIT (Simple User Interface Toolkit) Version 2.3 Reference Manual. University of Virginia, Department of Computer Science (e-mail: suit@uvacs.cs.virginia.edu), Blacksburg, VA.

Goldberg, D.E. (1989) Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, MA.

Grefenstette, J.J., Editor (1985) Proceedings of an International Conference on Genetic Algorithms and Their Applications. U.S. Navy Center for Applied Research in Artificial Intelligence. L. Erlbaum Associates, Hillsdale, NJ.

Grefenstette, J.J., Editor (1987) Proceedings of the Second International Conference on Genetic Algorithms. L. Erlbaum Associates, Hillsdale, NJ.

Hill, G.K. (1989) Mit P.R.O. System Spritzungen Sparen. Weinwirtsch., 125: 28–31.

Hill, G.K. (1990a) Das Peronospora-Risikoprognosemodell Oppenheim (P.R.O.) im Praxistest. Dtsch. Weinbau, 44: 514–517.

Hill, G.K. (1990b) Studies on *Plasmopara viticola* epidemics in the vineyard. In: Proc. Atti del Convengo su Modelli Euristici ed Operativi in Agricoltora, Caserta, Italy, 27–29 Sept. 1990. Societa Italiana de Fitoiatria ed., pp. 266–273.

Hill, G.K. (1990c) Plasmopara Risk Oppenheim — A deterministic computer model for the viticultural extension. In: Proc. Atti del Convengo su Modelli Euristici ed Operativi in Agricoltora, Caserta, Italy, 27–29 Sept. 1990, Societa Italiana de Fitoiatria ed., pp. 182–194.

Hill, G.K. (1993) Severe primary infections of *Plasmopara viticola* at bloom — Causal climatic conditions and threshold of damage on bunches. In: Proc. OILB Meeting on Integrated Control in Viticulture, Bordeaux, France, 2–5 March, 1993.

Hill, G.K., Breth, K. and Spies, S. (1993) The application of the P.R.O. — Simulator for minimizing of plasmopara sprays in the frame of an integrated control project in Rheinhessen, Germany. Vitic. Enol. Sci., 48: 176–183.

Holland, J.H. (1975) Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, Mich.

Jacucci, G., Foy, M. and Uhrik, C. (1994) SYBIL DSS: localization of agricultural risk assessment models. In: Proceedings of Decision Support 2001 (DSS2001) (The 17th Annual Geographic Information Seminar & The Resource Technology '94 Symposium), American Society of Photogrammetry and Remote Sensing.

Jacucci, G., Foy, M. and Uhrik, C. (1995) An Artificial Intelligence (AI) Methodology for the Adaptation of Agricultural Models. Engineering Applications of Artificial Intelligence. Pergamon Press Ltd., Oxford.

Schaffer, J.D., Editor (1989) Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann Publishers, Inc., Los Altos, CA.

Sequeira, R.A., Olson, R.L., Willers, J.L. and McKinion, J.M. (1994) Automating the parameterization of mathematical models using genetic algorithms. Comput. Electron. Agric., 11: 265–290.

Computers and electronics in agriculture

Submission of articles. Manuscripts should be submitted in triplicate to R.L. Olson (Americas) or to S.W.R. Cox (rest of the world). Review articles should be submitted in triplicate to M.A. Foster. Book reviews to C.J. Studman (addresses on inside front cover). Authors whose native language is not English are strongly advised to have their manuscripts checked by an English-speaking colleague prior to submission.
Authors in Japan please note: Upon request, Elsevier Science Japan will provide authors with a list of people who can check and improve the English of their paper (*before submission*). Please contact our Tokyo office: Elsevier Science Japan, 20-12 Yushima 3-chome, Bunkyo-ku, Tokyo 113; tel. (03)-3833-3821; fax (03)-3836-3064.

All questions arising after acceptance of the manuscript, especially those relating to proofs, should be directed to: Computers and Electronics in Agriculture, Elsevier Science B.V., P.O. Box 1527, 1000 BM Amsterdam, The Netherlands, tel. (+31-20)4853276, fax (+31-20)4853258.

Electronic manuscripts. Electronic manuscripts have the advantage that there is no need for the rekeying of text, thereby avoiding the possibility of introducing errors and resulting in reliable and fast delivery of proofs.
For the initial submission of manuscripts for consideration, hardcopies are sufficient. For the processing of *accepted papers*, electronic versions are preferred. After *final acceptance*, your disk plus two, final and exactly matching printed versions should be submitted together. Double density (DD) or high density (HD) diskettes (3½ or 5¼ inch) are acceptable. It is important that the file saved is in the native format of the wordprocessor program used. Label the disk with the name of the computer and wordprocessing package used, your name, and the name of the file on the disk. Further information may be obtained from the Publisher.

Proofs. One set of proofs will be sent to the author to be checked for printer's errors. In case of two or more authors please indicate to whom the proofs should be sent.

Reprints and page charges. There is no page charge. Fifty reprints of each paper and 100 reprints of each review article published will be supplied free of charge. Additional reprints can be ordered on a reprint order form which is included with the proofs.

COMPUTERS AND ELECTRONICS IN AGRICULTURE HAS NO PAGE CHARGES

All contributions will be carefully refereed for international relevance and quality. Submission of an article is understood to imply that the article is original and unpublished and is not being considered for publication elsewhere.

For a full and complete Guide for Authors, please refer to
Computers and Electronics in Agriculture, Vol. 14, No. 1, pp. 91–97

0168-1699/96/$15.00

∞ The paper used in this publication meets the requirements of ANSI/NISO 239.48-1992 (Permanence of Paper)

PRINTED IN THE NETHERLANDS