ELSEVIER

Mathematics and Computers in Simulation 39 (1995) 225–231

MATHEMATICS AND COMPUTERS IN SIMULATION

APSIM: an agricultural production system simulation model for operational research

R.L. McCown *, G.L. Hammer, J.N.G. Hargreaves, D. Holzworth, N.I. Huth

QDPI / CSIRO Agricultural Production Systems Research Unit, P.O. Box 102, Toowoomba, Australia, 4350

1. Introduction

In 1991 the CSIRO Division of Tropical Crops and Pastures and the Queensland Department of Primary Industries established the Agricultural Production Systems Research Unit (APSRU) with a mission "to benefit subtropical Australia and the nation through client-oriented agricultural systems R&D leading to improvements in production efficiency, risk management, and sustainability". APSRU brought together two groups which had each developed, as part of their systems research capability, a model for simulating cropping systems. PERFECT [1] was developed primarily to simulate the effects of erosion on the productivity of vertisols in the Australian subtropics, as influenced by soil management [2], but has been used for operational research in crop production [3]. AUSIM [4] was developed to serve research in the semi-arid tropics of Australia and Africa, particularly with respect to rotations and intercropping of coarse grains and legumes. Since the formation of APSRU, we have developed a new crop production systems model drawing on both AUSIM and PERFECT and the experience gained in their development.

APSIM (Agricultural Production System sIMulator) has been developed to assist the search for better farming strategies and the development of aids to better production decision making under risk. The agricultural systems of particular concern are those in which rainfall is uncertain and often deficient, and where fertility depletion and/or soil erosion threatens the economic future of crop production. A useful simulation package must deal credibly with both the season-to-season variability of production and the long term trends in production in response to changes in the soil resource. Effectiveness and efficiency in doing this requires: (1) crop models with sufficient sensitivity to extremes of environmental inputs to predict yields well enough to provide the inputs for economic analyses, including the analysis of risk, (2) models to

* Corresponding author.

SSDI 0378-4754(95)00063-P

simulate trends in soil productivity and erosion as influenced by management, including crop sequences and crop residue management, (3) software that allows easy re-configuration of a system model, (4) a friendly computing environment for development, testing, and use of the models, and (5) software that is reliable and easy to maintain as it evolves. This paper describes the concepts, software, and management strategies that contribute to APSIM's achievement of these attributes and capabilities.

2. Structure of the model and program

The key concept is the central position in the model of the soil rather than the crop, in spite of the fact that the primary simulated variable is crop yield. Changes in the status of soil state variables are simulated continuously in response to weather and management. Crops come and go, each finding the soil in a particular state and leaving it in an altered state. All crops share the same path of soil + atmosphere that contains various processes in or above the soil e.g. soil water and nitrogen cycles and surface residue decomposition. This concept allows ready simulation of the effects of one crop on another *via* its effects on the soil in both sequences and mixtures of crops.

This model concept is implemented in the APSIM program using the structure shown in Fig. 1. Various high order processes are represented as modules which relate to each other only through the *Engine*. Plant growth modules are interchangeable, and more than one can be connected simultaneously. This "plug in - pull out" capability enables the achievement of flexible simulation of cropping systems while using the best crop models for yield prediction. Before a simulation run, the growth routines for the required crops are selected (by ticking boxes in a Microsoft Windows screen menu). Each crop "plants itself" according to user-specified rules or schedules implemented by the *Operations Manager*.

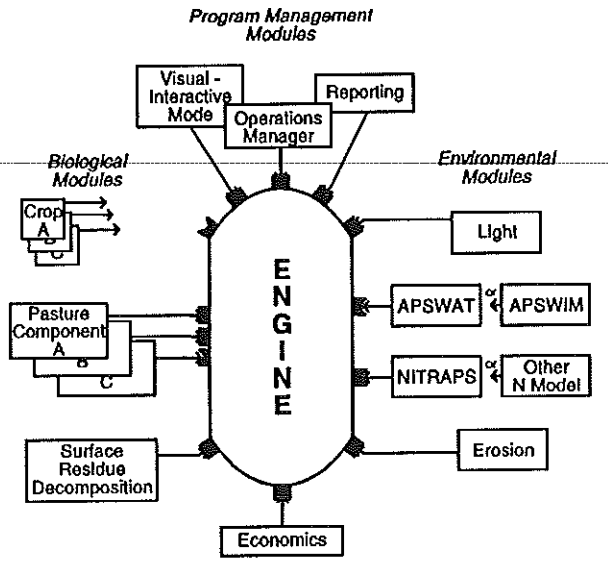

Fig. 1. The structure of the APSIM program.

The Operations Manager contains representations of relevant management actions realistically taken in response to conditions. Actions, e.g. choice of crop, planting, application of fertiliser, tillage, can be either scheduled or controlled using conditional rules. The language for expressing rules is based on "IF (condition) THEN (action)". This form allows great flexibility and enables ready construction of complex rules. The *System Log* records all operations and enables the establishment of complex conditional strategies using management history to be specified and evaluated.

Interchangeability of alternative models of a given environmental process, e.g. water balance, enables selection of the model best suited to needs in a particular application. This feature also makes it easy to compare the performance of different modelling approaches, all other modules remaining unchanged.

The Engine has been designed to perform mainly one function, i.e. the passing of messages to modules from either itself or other modules. Program functions, such as reporting, which might well have been included in the Engine, instead, appear as *Program Management* modules (Fig. 1). This forcing of all functions other than communication into modules means that enhancements to APSIM can be added with minimal change to existing code.

3. The user computing environment

A user interface has been developed that provides a suite of tools for developing, testing, and maintaining module code, running the model, and presenting and analysing output. This computing environment is provided as a Microsoft Windows program, with multiple windows, mouse and keyboard input, pull down menus, and option selection buttons.

The operating environment for key modelling operations is shown in Fig. 2. The tools which are available are shown around the perimeter. For code development the user can install an editor of choice. Clicking on the button of a module opens the source code file in the editor. A novel aid to code development is provided by the *Test Bed Generator* (TBG). Code for analysis is selected using the editor. The TBG analyses the code for variables used and writes a driving routine that reads inputs and produces an output file. The APSIM graphics utility enables output to be readily graphed and the behaviour of mathematical functions observed.

The routine created by the TBG is also used to trap logical and coding errors in the source code. The aim of testing is to make a routine crash or produce incorrect results by using very wide ranging inputs. The test bed reads test input files and generates output that is evaluated for mathematical errors, extrapolation of equations, interactions between equations, and processing order.

The validation and calibration operations require iterative sequences of code development, model configuration, model runs and analysis of model output. Selection of the editor and commands to compile, link, and run the model are all done using a pull-down menu. Flexible output control routines are provided to construct two-way tables of observed experimental data and simulation data output and to display graphically or print observed-predicted comparisons. A statistical package is available *via* Windows, and output can be included in graphs.

The environment for using the model in research is designed for ease in manipulating the configuration of the systems model, presenting appropriate input data, making simulation runs,

presenting output, and making comparisons and analyses of both physical and economic outputs. Modules are selected for inclusion in a run by clicking on the appropriate boxes. Parameter files for crop cultivars and for soil taxons are selected from the database, APSDAT, using the *Database* pull-down menu. Retrieved files can be modified, e.g. specification of initial values for soil state variables, using the Editor. Historical weather input files are similarly selected from the database, APSMET.

A run is specified by construction of a *Run Control File* using the *Editor*. This file designates the modules and files that have been selected on the screen. The name of the output file is designated and characteristics of output specified. Information for controlling the run is supplied as Start and End dates and by identifying the Management file. The latter file is created using the Editor and contains rules and or schedules for farm management actions during the period of the run and/or conditional or non-conditional directions for program execution.

The executable file is created by selecting *Make APSIM* from the *Compile* pull-down menu. This links selected modules and builds a runtime file. A run is initiated by selecting *Run APSIM* from the *Run* menu. Output tables can be examined using the editor or graphed. From the Run menu, graphs in the form of time series, x–y, actual *vs* predicted, or cumulative probability can be selected. Utilities to aid comparison of output are readily accessible, e.g. stochastic dominance and mean-variance analyses.

The system has been designed so that the source code can be ported to another computing platform, e.g. to gain greater computation speed for large runs.

4. Investment in control of software quality

A major investment has been made in software quality and in systems for maintaining quality as APSIM evolves. We have implemented the software configuration and quality control management system described by Arthur [5], i.e. modular design, coding style standards, moderation of code by peers, careful testing of code by other programmers, change requests, version control, and staged, scheduled development and releases of new versions. Of the criteria contributing to software quality, the following deserve special comment. First, keep program decomposed into a large set of small, highly independent, closed subroutines which can be called from any other subroutine and which can be separately compiled. Second, maximise subroutine independence by minimising the data relationships between modules and maximising relationships within subroutines. Third, define subroutines such that each performs one function, and fourth, keep modules small to aid independence, readability, and ease of testing. Additionally, state variables are only changed in a single high level subroutine, with the lower level subroutines returning the rates of change. This reduces problems of order of processing..

All code is written in Standard Fortran 77. Structuring, extensive use of program modules, use of meaningful variable and subroutine names and meaningful clarifying comments in the code contribute to the readability and maintainability of the program. Standard documentation is used for all subroutine header templates and includes descriptions and units in the

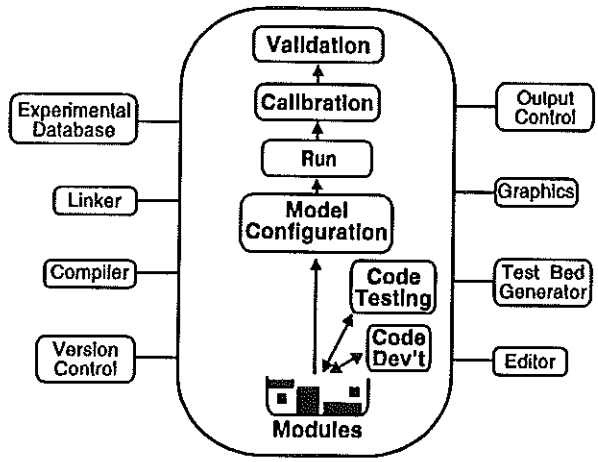

Fig. 2. The user environment for program development and maintenance. Alternative operations are shown within the boundary and tools and resources outside.

declaration of all variables/arrays. Both modules and subroutines within modules are easily added, removed, and interchanged.

To capitalise on past achievements in model development, select current models/submodels are reverse-engineered and then redesigned and re-coded into functional subroutines using the design principles described above. Important parts of APSIM code began by redesign of the soil water, soil nitrogen and crop growth submodels from CERES Maize [6], with substantial modifications based on other programs. APSIM reuses the redesigned growth routine as a template for other crops. This strategy means that models of other crops are based on an existing well-designed and tested module, thus reducing potential for errors, design time, coding time, debugging time and testing time. Familiarity of design, functional units and names facilitates recoding and maintenance. But this does not restrict the addition, replacement, or omission of code in developing a growth routine for a new crop.

5. Management of software evolution and change

To ensure order in the process of software evolution, we use a system described by Arthur [5] in which changes are initiated and recommended by scientists, but changes managed by professional programmers. Source code of all production releases are maintained in libraries using a *Version Control* system (Fig. 2). Enhancements to process modules are initiated by scientists. A prototype must pass review by a *module working group* of peers and then proceeds to the programming team as a *change request*.

6. The biological, environmental, and economic modules

Although our aims include the capability to use virtually any crop or soil model in APSIM after appropriate interfacing, we also set out to provide superior modules of biological and

environmental processes utilising the re-designed and re-engineered crop growth, soil water and soil nitrogen routines as a starting point. The APSIM crop growth template combines re-engineered routines from CERES Maize modified for the semi-arid tropics [7,8] with simplifying concepts and routines from QSORG [9] and QSUN [10]. In the case of a layered storage type soil water model, APSWAT results from re-engineering of CERES water balance and introducing alternative infiltration and runoff subroutines from PERFECT [1]. Implementation of Richard's equation in SWIM [11] is the basis of an alternative water movement module in APSIM which includes solute transport algorithms. In the case of nitrogen, NITRAPS results from the re-engineering of the CERES nitrogen routine and the introduction of a labile organic nitrogen pool [12].

In APSIM, a surface residue decomposition module contains functions for (a) decomposition of material in contact with the soil, as influenced by tillage and water supply, temperature, mineral nitrogen status of the surface layer of soil and (b) movement of standing material to the soil surface.

The erosion module is that of PERFECT [1,2].

The Economics module keeps track of cash flows during multi-period production runs, enabling rules based on the cash flow position in the Operations Manager. Whenever an activity is called by another module, the cost of this activity is retrieved from an *Economics Parameter File*. The Economics Parameter File is set up from a database of standard costs (by crop, activity, and region). Crop prices are handled similarly.

7. Conclusions

APSIM represents a step forward in simulation of the complexity of management of a paddock over time in terms of production efficiency, stability, and sustainability. This is achieved first, by improved representation of certain aspects of cropping systems which enables important phenomena to be better simulated. Second, good routines of diverse models can be easily recombined to provide a superior configuration. Third, APSIM provides an infrastructure that can support convergent effort by teams in the testing and improving of model functions using field data.

Acknowledgments

The multiplicity of contributions to APSIM by other members of APSRU defies individual acknowledgment. APSIM would not exist without the dedicated team effort of APSRU staff. We acknowledge the commitment of the team.

References

[1] M. Littleboy, D.M. Silburn, D.M. Freebairn, D.R. Woodruff and G.L. Hammer, PERFECT–A computer simulation model of Productivity Erosion Runoff Functions to Evaluate Conservation Techniques, Queensland Department of Primary Industries Bulletin, QB89005, 1989.

[2] M. Littleboy, D.M. Silburn, D.M. Freebairn, D.R. Woodruff, G.L. Hammer and J.K. Leslie, Impact of soil erosion on production in cropping systems. I. Development and validation of a simulation model, Aust. J. Soil Res 30 (1992) 757–774.
[3] J.P. Dimes and D.M. Freebairn, Analysis for optimal water use in grain cropping systems of N.E. Australia, Proceedings of the 7th Australian Society of Agronomy Conference, February 1996, Toowoomba, Queensland, to appear.
[4] R.L. McCown and J. Williams AUSIM: A cropping systems model for operational research, Proc. SSA IMACS 1989 Biennial Conference on Modelling and Simulation, ANU (1989).
[5] L.J. Arthur, Software Evolution–The Software Maintenance Challenge, (Wiley, New York, 1988).
[6] C.A. Jones and J.R. Kiniry, CERES-Maize: a simulation model of maize growth and development. Texas A&M University Press, College Station, Texas, 1986.
[7] P.S. Carberry, R.C. Muchow and R.L. McCown, Testing the CERES-Maize simulation model in a semi-arid tropical environment, Field Crops Research 20 (1989) 297–315.
[8] P.S. Carberry and D.G. Abrecht, Tailoring crop models to the semiarid tropics, in: R.C. Muchow and J.A. Bellamy, eds., Climatic Risk in Crop Production: Models and Management for the Semi-Arid Tropics and Subtropics, (Wallingford, CAB International, 1991) 157–182.
[9] G.L. Hammer and R.C. Muchow, Quantifying climatic risk to sorghum in Australia's semiarid tropics and subtropics: Model development and simulation, in: R.C. Muchow and J.A. Bellamy, eds., Climatic Risk in Crop Production: Models and Management for the Semi-Arid Tropics and Subtropics (Wallingford, CAB International, 1991) 205–232.
[10] S.C. Chapman, G.L. Hammer and H. Meinke, A sunflower simulation model: I. Model development, Agronomy J. 85 (1993) 725–735.
[11] P.J. Ross, Efficient numerical methods for infiltration using Richards' equation, Water Resources Research, 26 (1990) 279–290.
[12] J.P. Dimes and R.L. McCown, Potentially mineralizable N — a new role in predicting soil N supply, Proceedings of the 6th Australian Society of Agronomy Conference, February 1992, Armidale, New South Wales (1992) 374–377.