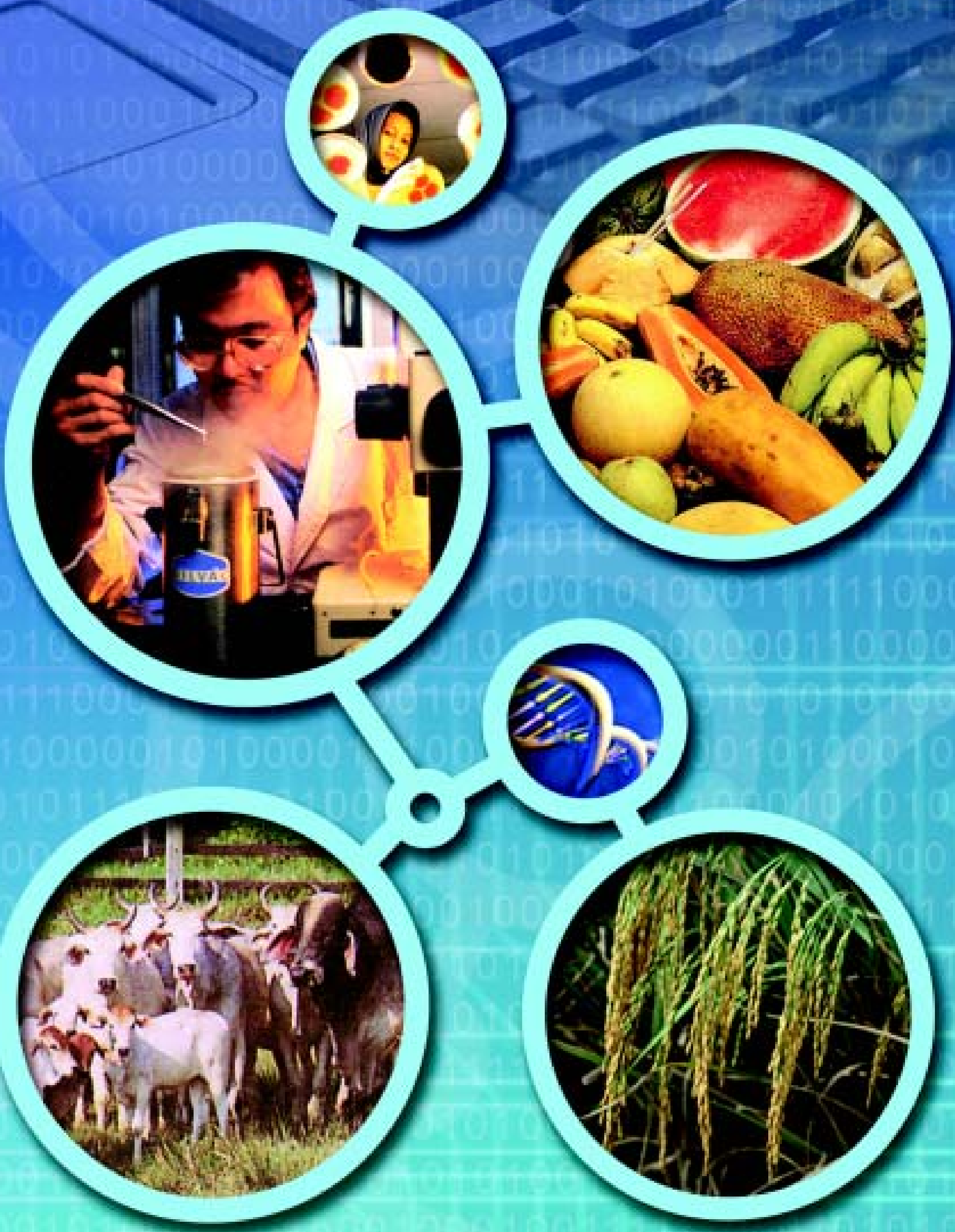
National Conference On AgrICT2005

Theme: Revolutionising Agriculture Through ICT

PROCEEDINGS



Main Organiser:

Malaysia Agricultural Research & Development (MARDI)

Supported by:

Ministry of Agriculutre and Agro-Based Industry Malaysia (MOA)

Co-organiser:

The Malaysian Institute of Microelectronic System (MIMOS)

Multimedia Development Corporation (MDC)

The Institution of Engineers, Malaysia (IEM)

CAB International

P R O C E E D I N G S

National Conference on AgrICT 2005

Theme: "Revolutionising Agriculture through ICT"

Date: 27-28 September 2005
Venue: Berjaya Times Square Hotel and Convention Center, Kuala Lumpur
Organizer: Malaysian Agricultural Research and Development Institute (MARDI)

In cooperation with

MIMOS Berhad
Multimedia Development Corporation (MDC)
The Institute of Engineers Malaysia (IEM)
CAB International

Endorsement by
Ministry of Agriculture and Agro-Based Industry, Malaysia (MOA)

Edited by
Dr. Ibni Hajar Rukunudin
Dr. Umi Kalsom Abu Bakar
Dr. Mohd Norowi Hamid
Dr. Chan Chee Wan
Muhamad Isa Othman
Muhammad Zamharir Ahmad

ISBN 967-936-491-7

First Published 2005

Published by:
Malaysian Agricultural Research and Development Institute (MARDI)
Ministry of Agriculture and Agro-Based Industry, Malaysia

DO NOT REINVENT THE WHEEL: EXTENDING THE LIFE SPAN OF AGRICULTURE MODELS

Teh[1], C. B. S., Henson[2], I., Harun[2], H., Goh[3], K. J., Husni[1], M. H. A.
Department Land Management[1], University Putra Malaysia, Serdang, Selangor, Malaysia.
Email: chris@agri.upm.edu.my; Tel: 03-89466976

[2] Malaysian Palm Oil Board, Bangi, Selangor, Malaysia

[3]Applied Agric. Res. Sdn. Bhd., Sg. Buloh, Selangor, Malaysia

Keywords: design pattern; model; oil palm; reusability; extendibility

ABSTRACT

The objective of this paper was to discuss several guidelines in model design so that agriculture models are developed to support model reusability and extendibility and further development by other workers; consequently, prolonging the life span of models. SAWIT is an oil palm growth and yield model that has been designed specifically for these purposes. It achieved these purposes by following three concepts: 1) decouple the interactions among the model components, 2) separate the model engine and model user interface, and 3) design the model engine to be independent of any hardware and software. These concepts created a stable and organised framework for model perseverance.

INTRODUCTION

The model development process is often guided by two goals: achieving model accuracy and model perseverance. Of the two, model accuracy is understandably of greater importance. A model is created primarily to solve a problem, and a model is only useful if it solves the given problem with the desired precision. In contrast, however, model perseverance is rarely achieved. Agriculture models often have short life spans. Models are created, used in limited conditions then discarded for newer ones. Moreover, there is a great deal of overlap among agriculture models, where different models often share the same or similar approaches or equations. Consequently, these problems have led Loomis (1985) and Seligman (1990) to remark the development of agriculture models today are akin to "reinventing the wheel". It would be better instead if a few useful existing models are identified for a given task, then developed further by various researchers. The scientific community working on a few existing models instead of creating new ones from scratch has several advantages. Fewer models provide focus, avoid duplication of work, and help to speed progression of knowledge through the intense scrutiny and sharing of knowledge by workers from various background and experience.

Nonetheless, achieving model perseverance requires much thought in the model design. Consequently, it is the objective of this paper to discuss an agriculture model that has been designed specifically to have a stable and organised framework that allows for model evolution; that is, adaptable to change and subsequently a longer life span. The model to be used for this paper's discussion is an oil palm model called SAWIT.

THE MODEL DESIGN OF SAWIT

SAWIT is a model that simulates the oil palm growth and yield, taking into account water and flowering cycle stresses (Teh *et al.*, 2005). This model was designed to overcome three common flaws found in most agriculture models: 1) tight coupling among the model components, 2) no separation between the model user interface and model engine, and 3) dependency on hardware and software.

Interaction Among Model Components

It is good practice to solve a large problem by dividing it into several smaller and more specific tasks. Consequently, an agriculture model is often an assemblage of smaller submodels or components, where each component performs one or more highly specific tasks. Together, these components interchange information, perform their respective calculations, and ultimately solve the given problem.

This interchanging of information means that there is a great deal of interaction or coupling among the components. And it is this interaction among the model components that hinders a model's adaptability to change. This is because closely linked components create a very intricate network of information flow, where one component not only requires information from other components, but it also supplies information to other components (Figure 1). Consequently, this tortuous intricacy creates a rigid model structure, which makes it difficult to modify, evolve and reuse the model for new knowledge or approach. A change in one model component, for example, often breaks the whole model structure because a change produces a "domino effect" where other components are undesirably affected and have to be modified as well. This "domino effect" results from components that are locked together in an inflexible relationship.

The SAWIT model reduces this tight coupling or interaction among components by following the Courier design pattern (Gamma and Helm, 1996) which is a derivative of the Mediator pattern (Gamma *et al.*, 1995). Figure 2 shows that the SAWIT model comprises of seven components but in contrast to Figure 1, the intricate coupling among components has been very much reduced. The many-to-many relationships among the components have been simplified to a one-to-one relationship in SAWIT.

Unlike the other six components, the microclimate (MC) component actually does not perform any calculations, but its role is vital. It acts as a mediator, forming a hub or communication centre which promotes loose coupling by keeping the other components from referring to each other explicitly. Except for the MC component, the other six components do not communicate with each other directly, and they do not even "know" the existence of each other. For the six components, the MC component is their only source of external information.

The solar radiation component (RAD), for example, calculates the daily solar interception by the oil palm. To do this, the RAD component requires information regarding the current leaf area index (LAI). The RAD component "submits" a request for this information to the MC component. Being the mediator, the MC component calls the appropriate LAI routine found in the growth component (GROWTH). Obtaining this information from GROWTH, MC completes the request cycle by finally relaying the information to the RAD component.

This design is crucial because it allows for one component to be modified for new knowledge or approach without breaking the whole model structure. This design even allows for one component to be replaced completely by another component. In the current SAWIT model, for example, the hourly solar irradiance is simulated by the meteorology component (METEO). In the future, a simpler calculation may be sufficient and the current METEO component could be replaced entirely by a new component METEO2. Entire replacements of one or more components can be done without breaking the whole model structure because, as stated earlier, no two components (except for MC) communicate directly with each other.

One problem that could arise when a component is replaced completely is that certain request cycle could become incomplete. The RAD component, for example, requires information about the solar position to calculate the canopy extinction coefficient. This information is handled by MC by calling a routine found in METEO. But if the current METEO component is to be replaced by METEO2 and if this new component does not calculate the solar position, then this particular request by RAD cannot be fulfilled. Nonetheless, this problem is solved easily by creating a new component (SOLARPOS) which calculates the solar position. The SOLARPOS is then "attached" to the MC component (Fig. 3) so that the mediator now calls the routine found in SOLARPOS instead of METEO as was done before. Moreover, because the RAD component does not “know” the existence of other components, the attachment of the new component SOLARPOS and the replacement of METEO by METEO2 do not affect the RAD component in any way. This concept is akin to the “plug-and-play” concept to describe the connection between a computer and its peripherals.

Separation Between the Model User Interface and Model Engine

Another problem often found in agriculture models is the lack of separation between the *user interface* and model *engine*. The user interface in the model is the section of code that deals with user interaction such as producing a graphical-based menu system, complete with buttons and dialog boxes, to receive user selection or input for model parameters, as well as to display or print the model output. The model engine, in contrast, is the section of code that deals with calculations that form the core and purpose of the model. Unfortunately, in some agriculture models, there is an indistinguishable mix between both sections of code dealing with the user interface and model engine. This again creates code that is both difficult to understand and modify. But more importantly, a change in the model user interface code undesirably affects the model engine code.

Recognising this problem, the SAWIT model separates both sections (Figure 4). And because of this separation, it is easy to identify parts of code that has to be modified, and modification does not break the whole model structure. To achieve model perseverance, it is the model core, or its engine, that would have to be scrutinised and developed further by workers. The model user interface section is merely a front for users to interact with the model engine, and keeping this section apart form the model engine greatly simplifies the model development of SAWIT.

Dependency on Hardware and Software

One obstacle to model sharing is when a model is developed unnecessarily to work only on a specific hardware and software platform. A model that runs only on Intel-based PC systems and on Microsoft Windows, for example, hinders collaborative modelling research. Furthermore, some models can only be compiled on a compiler from a specific company only.

These problems happen because modellers often fail to distinguish the standard and non-standard features of the computer programming language they are using. Popular programming languages used by the agriculture modelling community such as C++, Pascal and FORTRAN are guided by the international standards set by ISO (International Organization for Standardization). Keeping models to these standards is essential because programs written following the standards are guaranteed to compile and run on any system or platform.

These standards are followed by all large software companies, but complications arise when these companies often add extensions to the language that are unique only to the company’s compiler. For example, the command “_open” is a C++ command to open a file. This command, however, is specific only to Microsoft C++ compilers. Thus, a model that uses any company-specific commands will fail to compile on a compiler from a different company. Likewise, a model that calls or uses any hardware- or software-specific instructions will fail to compile and run on a different platform.

One major advantage of making a model hardware and software independent is the ease of designing the model’s user interface. As discussed in the previous section, a model consists of two sections: the model engine and the model user interface. By keeping both sections separate as advised earlier, the model engine can be written using standard language so that it remains hardware and software independent and thus the model engine can be used regardless of the platform. The model user interface, however, can be written using non standard language because developing a user interface often involves using specific hardware and software instructions (such as accessing the mouse or printer).

In the SAWIT model, the model engine was written in standard C++; consequently, it is independent of any hardware and software. Its model user interface, however, was written using the MFC (Microsoft Foundation C++ Classes) to design its graphical user interface. The MFC library can only be compiled on Microsoft C++ compiler, and the user interface can only be used on the Windows operating system.

Though both of its model sections in SAWIT are separated, they are still able to communicate with each other (Figure 4). Via the user interface, the user will enter the model parameters and choose a menu option to begin the model simulation. The user interface section then calls the appropriate routine in the model engine to begin the model run. When completed, the simulation results are displayed in the user interface either by reading the output file or by reading the storage held in the computer's memory.

Notably, because both model sections are kept apart, the model engine can be developed further without the distraction of the type of hardware and software. Recall that it is the model engine section that is of greater interest because it is this section that has to be modified when new knowledge is acquired. The model user interface is merely a front for the model engine.

CONCLUSION

SAWIT is an example of an agriculture model that has been designed specifically for prolonged life span by supporting further model development by various workers. This is achieved by following three important concepts: 1) decouple the interactions among the model components, 2) separate the model engine and model user interface, and 3) make the model engine independent of any hardware and software.
The source code for SAWIT is available upon request from the author.

ACKNOWLEDGMENT

This study was funded by the Ministry of Science, Technology and Innovation, Malaysia, IRPA RMK8 EAR Grant No. 01-02-04-0686-EA001.

REFERENCES

Gamma, E. and Helm, R. 1996. The Courier pattern. *Dr. Dobb's Sourcebook.* 3: 55-59.
Gamma, E., Helm, R., Johnson, R. and Vlissides, J. 1995. Elements of reusable object-oriented software. Addison-Wesley, Reading, MA.

Loomis, R. S. 1985. Systems approach for crop and pasture research. In: Proceedings of the 3rd Australian Society of Agronomy Conference, Hobart, Tasmania, pp. 1-8.

Seligman, N. G. 1990. The crop model record: promise or poor show? In: Rabbinge, R., Goudriaan, J., van Keulen, H., Penning de Vries, F.W.T. and van Laar, H.H. (Eds.) Theoretical production ecology: reflections and prospects. Pudoc, Wageningen, pp. 249-258.

Teh, C. B. S., Henson, I. E., Harun, H., Goh, K. J. and Husni, M. H. A. 2005. Modelling oil palm growth and yield. In: Teh, C.B.S., Ahmed, O.H., Fauziah, C.I., Izham, A., Wan Noordin, W.D. and Zakaria, Z.Z. (Eds). SOILS 2005: Advances in soil science for sustainable food production. Proceedings of the Malaysian Society of Soil Science 2005. pp. 204-206.

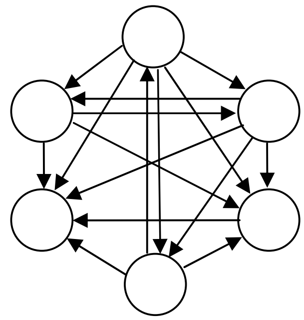
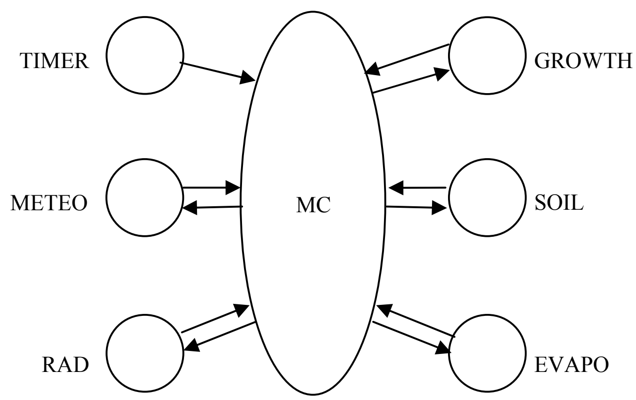


FIGURE 1. Model Components Typically Interact With Each Other. {Note: arrows denote flow of information}.



FUGURE 2. The SAWIT Model Consists of Six Somponents: timer (TIMER), meteorology (METEO), solar radiation (RAD), oil palm growth (GROWTH), soil water balance (SOIL) and evapotranspiration (EVAPO). The seventh component, microclimate (MC), is the mediator for the six components. Note: the TIMER component does not request for any information.

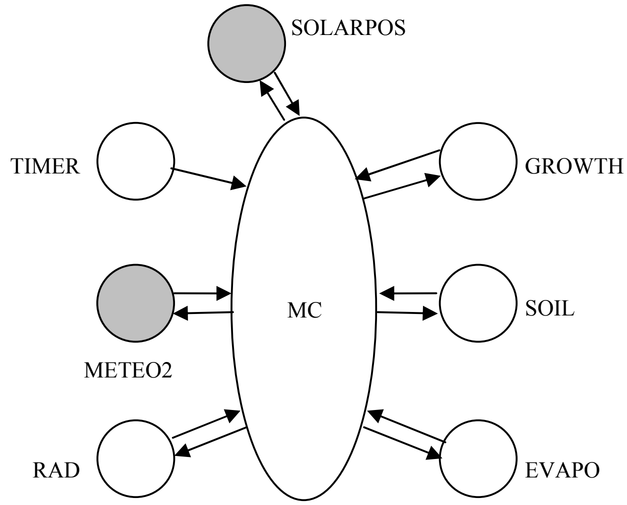


FIGURE 3. A scenario for SAWIT when the METEO component is replaced by METEO2. A new component, SOLARPOS, is “attached” to support the request for information on the solar position.

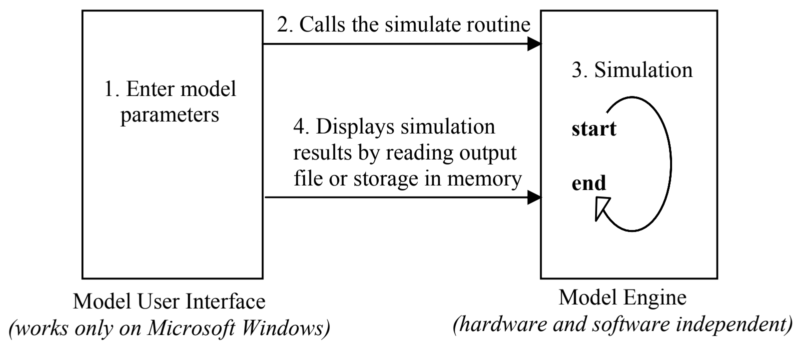


FIGURE 4. The Separation Between the Model Engine and Model User Interface in SAWIT