ELSEVIER

Ecological Modelling 94 (1997) 7–16

ECOLOGICAL MODELLING

Modularity and genericness in plant and ecosystem models

James F. Reynolds[a,*], Basil Acock[b]

[a]*Department of Botany, Phytotron Building, Box 90340, Duke University, Durham, NC 27708-0340, USA*
[b]*USDA, ARS, Remote Sensing and Modeling Laboratory, Building 007, Room 008, BARC-W, 10300, Baltimore Avenue, Beltsville, MD 20705-2350, USA*

Abstract

In this paper, we present definitions of modularity and genericness that are based on sets of criteria and rules for model design and which encompass the goal of developing an efficient and flexible structure for plant and ecosystem models. Model structure should be based on modules that (1) relate directly to real world components or processes; (2) have input and output variables that are measurable values; and (3) communicate solely via these input and output variables. Such a model structure has the advantage that it can be incrementally improved by simply replacing one module with another that has the same input and output variables. The underlying mechanism in the replacement module can be different, which facilitates the incorporation of the latest experimental research results and allows modelers to readily test alternative hypotheses about mechanisms. Thus, modularity and genericness open models to contributions from many authors, facilitate the comparison of alternative hypotheses, and extend the life and utility of simulation models.

Keywords: Modules; Generic design criteria; Modularity rules; Model re-use

"New ideas become accepted when the old generation dies and a new generation arises that is familiar with them." (Max Planck)

1. Introduction

During the past 20 years, dozens of agricultural and ecological simulation models—representing a wide range of empiricism and mechanism—have been developed (Joyce and Kickert, 1987; Reynolds and Acock, 1985; Reynolds et al., 1996; Reynolds and Leadley, 1992; Whisler et al., 1986). With few exceptions, these models have the following characteristics: (1) they were developed by an individual or a small group of scientists; (2) each has a unique structure (variables considered,

* Corresponding author. Tel.: +1 919 6607404; fax: +1 919 6607425.

0304-3800/97/$17.00
PII S0304-3800(96)01924-2

logic, etc.); (3) their documentation consists of a brief description published in the scientific literature; (4) they are not readily transportable to other platforms or laboratories for reuse by others; and (5) they are maintained only by their developers (if at all). As a result, these models have limited application, rarely outlive their developers, represent considerable duplication of effort, and are difficult to critique.

Both the agricultural and ecological sciences have placed a high priority on the use of simulation models to predict the potential effects of climate change and elevated atmospheric CO_2 concentration on plants and ecosystems (see Houghton et al., 1990; IGBP, 1990; IPCC, 1991; Melillo et al., 1990), and to develop decision aids for farmers, land managers and policy-makers. How can we choose from among the many models that are being produced? Increasingly, 'competitions' are being staged between models developed by different laboratories and research groups to compare predictions of a few key variables, such as crop yield or productivity (e.g. LIDET, 1995; Ryan et al., in press; Vangrinsven et al., 1995; VEMAP, 1995). In the long run, the value of comparing models will rest on our ability to elucidate the underlying causes of different predictions: e.g. variations in model structure, definitions of functional relationships, the logic employed, and the consequences of specific assumptions. This presents a major challenge. How do we evaluate and interpret the performance of models beyond the level of predictions per se? Do we compare the mathematical representations of key processes? Since most large simulation models contain a mixture of empirical and mechanistic formulations, should we attempt to quantify this mixture? Should we evaluate the sensitivity of different models to the use of alternative solution algorithms and to different assumptions? How do we judge the representations of key interactions? Should we identify the best single model or select the best parts from many models and build a general model based on the collective wisdom of the community?

We argue that plant and ecosystem models should be developed with an efficient and flexible structure, which can only be accomplished by dividing the system into biologically meaningful units or modules. We believe that doing so will address many of the problems listed above. We present definitions of modularity and genericness that are based on sets of criteria and rules for good model design. The essence of these is that the structure should be designed with modules that: (1) relate directly to real world components or processes; (2) have input and output variables that are measurable values; and (3) communicate solely via these input and output variables. Such a model can be incrementally improved by replacing one module with another module that has the same input and output variables but different underlying mechanisms.

In the following text we have tried to distinguish between model *design* and *implementation*. When we discuss models and modeling, we are referring to their design and the design process. Many plant modelers consider modeling as writing code, possibly because there has been a tendency to omit the design stage in the past. As a result, plant modelers have not developed a language for discussing model design, and many are not familiar with the language used by professional computer programmers. To explain some of the criteria and rules of modular design used by computer programmers, we have resorted to using parallel examples from software implementation.

2. Modularity defined

The dictionary definition of modular is: "constructed with standardized units or dimensions for flexibility and variety in use" (Webster, 1983). This raises the obvious question: "What standards?" Many plant modelers now recognize modularity as a 'good thing to have' and it is commonplace to see claims of modularity in the literature, although these modelers may only mean that their computer code (implementation) is divided into subroutines termed 'modules'. However, professional computer programmers have spent considerable effort on defining the desirable features of good software design. To them, the term 'modularity' has acquired a special meaning, connoting properties such as the separation and independence of the modules.

2.1. Modular design criteria

Meyer (1990) identified five criteria for good modular design: (1) decomposability, (2) composability, (3) understandability, (4) continuity, and (5) protection (Fig. 1). These design criteria enhance modularity and address many of the problems found in the models being used today (see Reynolds et al., 1989). They are independent of the design process used to produce the model, and the computer language used to implement it.

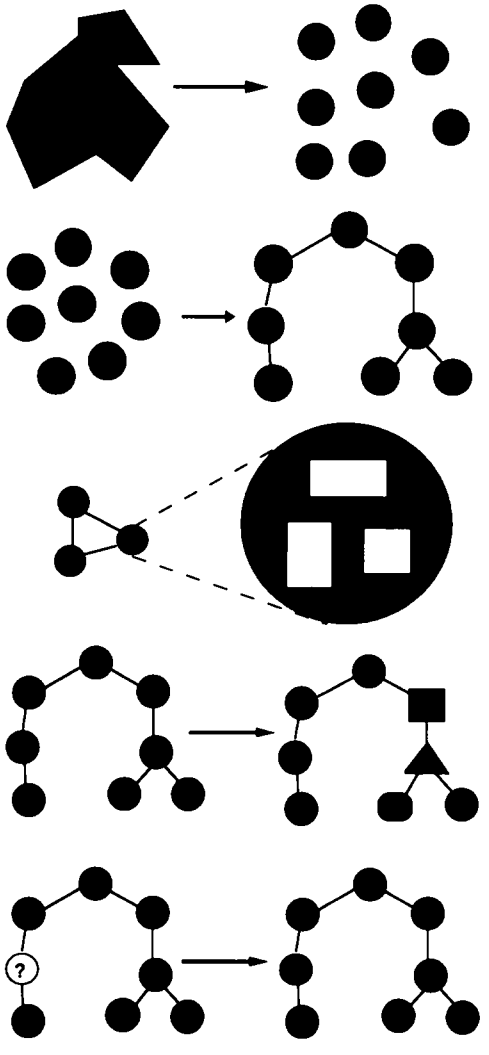

Fig. 1. Schematic representation of Meyer's (Meyer, 1990) criteria of good modular design. From top to bottom: (1) decomposability, problem can be decomposed into smaller independent subproblems (modules); (2) composability, existing modules can be recombined to model new systems: (3) understandability, individual modules are separately understandable; (4) continuity, small changes in specification changes only one or a few modules; (5) protection, effects of a run-time error confined to one or a few modules.

2.1.1. Decomposability

The decomposability criterion states that a good modular design should decompose the problem into smaller, independent subproblems (modules), each of which can then be solved separately. The critical feature is that the subproblems should be independent enough that different modelers can work on them separately. This is a matter of degree. In fact, it is always possible to have different modelers work on various subproblems, if the interactions between the subproblems are sufficiently well defined. In a good design the interactions between the modules should be readily apparent. Decomposability is, to some extent, a function of the problem being solved. A system consisting of discrete, sequential, independent processes is easier to decompose than a system of continuous, simultaneous, interacting processes. Plants and ecosystems are of the latter type (Acock and Reddy, 1997).

2.1.2. Composability

The advantage of decomposing a problem into subproblems is that the modules created to solve the subproblems may be reused. Modular composability requires that it should be possible to recombine existing modules to model new systems. The key factor is how easy it is to use the modules in other models. For example, math library modules are very easy to reuse in solving a range of problems, hence they have high composability.

2.1.3. Understandability

The criterion of understandability states that modules should be separately understandable. It should be possible for a reader to understand the function of a module without referring to other modules. The main issues here are the time and the amount of information outside the module required to understand it. If the correct function-

ing of a set of modules depends on their being activated in a certain order, they will not be separately understandable. Understandability and decomposability are closely linked.

2.1.4. Continuity

The continuity criterion states that a small change in the problem specification should lead to modifications in one or (at most) a few other modules. These changes should not require changes in the basic structure of a model.

2.1.5. Protection

The protection criterion is satisfied when the effects of a run-time error are confined to the module in which it occurs or, at worst, to a few other modules. This criterion does not address error avoidance or correction but rather the minimizing of error propagation. For example, an error in a photosynthesis module that causes a negative photosynthesis rate at noon should not then go on to reduce plant dry weight in a growth module. Meyer's example of good modular protection is to have each module validate input data at its source. A module that validates its own input data protects itself from input data errors.

2.2. Modular design rules

Modular design criteria are, by themselves, merely goals; to accomplish these goals we need a set of guidelines. To this end, Meyer (1990) proposed five design rules: (1) use linguistic modular units, (2) use few interfaces, (3) use small interfaces, (4) use explicit interfaces, and (5) use information hiding. Each rule supports one or more of the modular design criteria.

2.2.1. Linguistic modular units

Modules should correspond to the syntactic units in the language used to design the model. For plant modelers unaccustomed to talking about design language, this rule has little meaning. However, the idea is easily understood when applied to model implementation. Subroutines are syntactic units in FORTRAN. In implementation languages, this rule means that the modules should be separately compilable. They should not be a loose collection of procedures and variables within a syntactic unit, e.g. lines X–Y in subroutine Z. This rule follows from the criteria of decomposability, composability, understandability, and protection. Separating, combining and understanding modules is easier if the modules are well-defined; similarly, limiting the scope of errors to a module is difficult if it is not clearly delimited. This rule removes any hope that modularity can be achieved without the support of an appropriate implementation language (see examples in Acock and Reddy, 1997; Lemmon and Chuk, 1997; Sequeira et al., 1997).

2.2.2. Few interfaces

All five criteria require the rule of few interfaces—every module should communicate with as few others as possible. In Meyer's terminology, interfaces are communication channels. Increasing the number of channels between modules increases the likelihood that a change or error in one module will affect another module. Increasing the number of channels between modules also makes it harder to understand how each module operates.

2.2.3. Small interfaces

Communication between any two modules should involve a minimal exchange of information. As more information is exchanged, the chance of errors increases.

2.2.4. Explicit interfaces

Communication between modules is constrained even more by the explicit interfaces rule: If modules A and B communicate, this must be obvious from the text of A or B, or both. Hidden communication between modules violates all of the modular design criteria. For example, understanding a module can be very difficult if data are changed by some 'invisible' means, which happens when global variables are used. Anyone who has attempted to read and understand a large FORTRAN-77 program has faced this problem.

2.2.5. Information hiding

All information about a module should be private unless specifically declared public. We have

already stated that interfaces between modules should be small and explicit. This rule states that everything else should be hidden from other modules.

3. Genericness defined

The dictionary definition of generic is: "relating to or characteristic of a whole group or class" (Webster, 1983). We believe that the design of plant and ecosystem models should be generic—they should be generally applicable to a range of plants or ecosystems (Acock and Reynolds, 1989, 1990; Reynolds et al., 1986, 1989, 1993; Tenhunen et al., 1989). Innis et al. (1980) suggested that by recognizing certain 'classes' of processes, modeling efforts would be greatly reduced and researchers would begin to see the unique and different as exceptions, rather than the rule itself. A similar observation was made earlier by Forrester (1970) who, working with economic models, wrote: "...one should start not by building a model of a particular situation, but by modeling the general class of systems under study. This may seem surprising, but the general model is simpler and initially more informative than a model of a special case".

Computer programmers have adopted a narrower definition of genericness than the dictionary definition. Meyer (1990) considers a module to be generic if it can simulate several functionally equivalent systems just by using different values of the parameters. This definition is widely accepted by programmers and they apply it to modules and whole models. Biological modelers use the use term 'generic' loosely; sometimes in the sense of Meyer's definition, and sometimes in the sense of the dictionary definition. We think Meyer's definition is too narrow and restrictive.

The principal appeal of a generic design is economy—in terms of both modeling effort and understanding. Economy of effort is obtained when a generic design provides an alternative to developing ad hoc models for each plant or ecosystem of interest, and economy of understanding results from having to learn only one model structure. Few would disagree with this. However, a definition of genericness in biological modeling has not previously been proposed.

3.1. Generic design criteria

We propose four generic design criteria for a good generic design: (1) transferability; (2) additivity; (3) separability; and (4) testability.

3.1.1. Transferability

A good generic design must be suitable for application to any member of the target group of ecosystems or plants of interest by the use of different model parameters or different modules.

3.1.2. Additivity

A good generic design must be able to simulate functionally similar, yet different systems, by the addition (or subtraction) of modules. For example, a generic design developed for non-legumes should be able to simulate legumes by adding modules for nodule growth and nitrogen fixation.

3.1.3. Separability

Individual modules should be readily recognized by experts in the field as separate parts or processes of the system under study. A model design is not truly generic if it contains modules whose purposes are not readily apparent. Modules that combine several functions not normally considered together may be more difficult to parameterize for a new system. The generic separability criterion is complementary to the modular decomposability and understandability design criteria.

3.1.4. Testability

The parameterization and validation of each module should be independent of all other modules. A model composed of such modules is more likely to simulate a new system successfully.

3.2. Generic design rules

3.2.1. Common components

The transferability and additivity criteria are served by the rule of common components. The principal modules should represent components

common to as many systems as possible in the group of systems under study. In developing models to simulate a range of plants, for instance, the principal modules should correspond to components or functions common to all the plants under study.

3.2.2. Disciplinary separation. Scientists are trained and organized into various disciplines, and most of them work entirely within a single discipline. To make modules accessible to them, the modules must be separated along disciplinary lines (e.g. plant physiology, soil physics, biogeochemistry, etc.). Disciplinary separation will facilitate the interaction of specialists by allowing them to critique and contribute to the development of the modules from their discipline without requiring them to understand all parts of a larger model. Experimental scientists can update modules that correspond to their discipline as new knowledge becomes available. This leads to gradual improvement of the modules, simpler maintenance, and a greater likelihood that the model will outlive its original creator. The combining of modules across disciplinary lines is realized through the composability design criterion.

3.2.3. Input/output measurability. The input and output variables of a module should, as far as possible, be measurable properties of the system. This supports the testability criterion. It facilitates the parameterization and validation of modules, and the testing of alternative hypotheses via changes in formulation of the module structure and function.

4. Achieving genericness

We suggest several ways to achieve genericness in models:

Case 1: A *single* model is developed that is intended to simulate functionally- and structurally-equivalent systems solely through the use of different parameter values. There are two examples of this: (1) where the model is intended to represent widely different types of systems (e.g., saltmarshes, forests, deserts, croplands, etc.); and (2) where the model is intended to represent different case studies of a single system type, e.g., shortgrass prairie, alpine grassland, desert grassland, etc. In the former, the details of the system must be grouped together into relatively simple (but general) formulations, e.g. describing biomass accumulation as a function of total light intercepted over the growing season rather than as a function of instantaneous leaf photosynthesis. In the latter, more of the details uniquely associated with the system of interest may be included in the model.

Case 2: Essential components and/or processes of the system are defined as modules. For a model of plant growth these might include modules for photosynthesis, carbon allocation, nutrient uptake, etc.; for a model of ecosystem dynamics these might include modules for nutrient cycling, productivity, decomposition, etc. In keeping with the modular composability criteria described above, these modules can serve as building blocks for creating full models. However, the user must devise appropriate interfaces between the modules—following the modular design rules described above.

Case 3: Essential components and/or processes of a system are defined as modules, but unlike Case 2, the interfaces between them are also defined. These modules can be used to create full models because the interfaces have already been defined. The generic plant growth model (GePSi) developed by Chen and Reynolds (1997) is an example. GePSi encapsulates functions and properties common to most plants, has been parameterized and tested for predicting growth of loblolly pine under elevated CO_2 (Chen and Reynolds, unpubl.) and has been applied to study an annual grassland under elevated CO_2 (Luo et al., 1997). The latter required some reparameterization (as in Case 1) and modifications in the details of carbon storage and dynamics modules (Case 2). The model structure described by Acock and Reddy (1997) is another example. This model structure was developed for a suite of crop models with the intention that, as each crop model is developed, some modules will be reused without alteration, some will be reparameterized, and some will be completely rewritten, although they will retain the same interfaces.

The genericness achieved in Case 1 fits Meyer's definition, but the genericness in Cases 2 and 3 is closer to the dictionary definition. We propose another way to achieve genericness that builds on all of the above:

Case 4: Develop a generic, modular structure (GMS) as in Case 3 above, but one that represents a consensus of the scientific community. The GMS—which would be based on the criteria of modularity and genericness—should include explicit specifications of the modules, input and output variables, all of the linkages and feedbacks, etc. Any GMS based on consensus would have to be revised regularly as new knowledge and ideas became available. Is this a feasible goal? We offer some opinions below.

5. Where will this lead?

We see two possible outcomes of applying the modular and generic design criteria and rules enumerated above: (1) a collection of essential plant and ecosystem modules with the function(s) of each defined (see Case 2 above); and (2) a single GMS for plant and ecosystem models, with both module functions and interactions defined, agreed, and implemented (Case 4). These are actually successive steps along the same path, but either one could be the end point.

A collection of modules will give maximum freedom to modelers because they can chose the input and output variables, as well as the algorithms to perform the functions. This freedom will not result in complete anarchy because the functions being modeled will place some constraints on the input and output variables. However, it could result in modules that cannot be used together to construct a model. For example, if a photosynthesis module requires leaf area index and leaf age as input, it must be used with other modules that supply those variables as output. The disadvantage here is that it might not be possible to compare various hypotheses if they are encoded in modules with different input and output variables. Also, obtaining agreement on the list of modules is not a trivial undertaking.

A GMS model that defines module functions and interfaces would facilitate the testing of alternative hypotheses because the input and output variables for each module would be defined, making the modules 'plug-compatible'. Ultimately, we think obtaining agreement on a single structure is virtually impossible. 'Generic' and 'modular' are issues that everyone agrees are 'good things', yet there is no agreement on the precise form they should take. Hence, everyone develops their own model. Given that achieving a consensus GMS is an idealistic goal, we envision that steps towards this objective will be accomplished in an incremental fashion. For example, based on our experience, it might be relatively straightforward for crop modelers working on a single species like cotton to develop a consensus GMS for that species. Initially, this could build on the design and implementation schemes for cotton presented in this volume by Sequeira et al. (1997) and Lemmon and Chuk (1997). The demonstrated success of such a GMS for cotton modeling throughout the world would be an incentive to adapt it to other species (or the development of a GMS for other species), which could eventually lead to a consensus GMS for crop plants. A GMS is more likely to be adopted if it is developed by one or more modeling groups and made available for other scientists (the next generation?) to assess its strengths and weaknesses.

6. Summary

Many ecologists have commented on the problems with current models, noting in particular that they tend to be complex and difficult to understand, maintain, debug, and modify (Reynolds et al., 1989). The authors of the models readily admit to some of these difficulties. Potential users are obliged to accept the models as they stand. This has led to the current trend of staging modeling 'competitions' or comparisons, as noted in Section 1. The problem is that we do not have a method for continuously advancing and improving our models. The reason for this can best be understood by contrasting the different ways in which modelers and experimenters operate.

Ideally, experimenters design each experiment to test a specific hypothesis based on earlier results and hypotheses. The experiment is reproducible and the results are published along with the experimenters' interpretations and hypotheses. This procedure continuously advances our collective understanding. On the other hand, modelers rarely build on the work of others; instead, we tend to prefer to build from the ground-up, which is primarily because existing models are not designed for incremental improvement.

A well-defined modular generic structure for plants that is accepted by many modelers could be populated with modules by scientists with the relevant expertise. Such a model would belong to the scientific community rather than an individual, in much the same way as our understanding of plant processes is acquired and held collectively. Alternative hypotheses could be tested by replacing modules, and the model would advance as our collective understanding advances. Models for various plants or ecosystems might have different modules, but modules like those for soil and weather processes could be common to many models. There would be several modules for some processes, such as photosynthesis and carbon allocation. At various times there might be several versions of a module representing conflicting hypotheses where experimental data do not permit a final choice. It should be possible to develop new models by assembling existing modules. For example, a new model might consist of modules for C3 photosynthesis, a storage root, nitrogen fixation, etc., as appropriate. The whole modeling enterprise would become much more intelligible to both modelers and nonmodelers.

Models for different purposes require different levels of detail and comprehensiveness. A plant growth model is different from a community dynamics model, which is different from an ecosystem biogeochemistry model. At first it might appear that such models necessarily require different structures. However, with careful choice of modules even this might be avoided. If the modules in our structure are arranged in a hierarchy of increasing detail (e.g. Acock and Reddy, 1997; McMurtrie and Wang, 1993) we could choose modules with the level of detail appropriate to a particular application. At the simplest level, the structure might consist of modules for plant, soil and atmosphere, with the plant module containing a simple logistical curve describing growth in biomass. At a higher level of detail the plant module would call other modules dealing with photosynthesis, transpiration, etc. However, both models could use the same soil and weather modules. Adopting such a hierarchical structure would also tend to increase code reuse.

We recognize that implementing and accomplishing these goals will not be easy. Producing a module that can be used by others—that is, one that is easier to use and understand than it is to recreate—is a major challenge. Meeting the modular and generic design criteria we have proposed will not always be easy. Implementation always introduces new twists and problems. Computer programmers have learned that many years of development (and experience) are needed to produce reusable modules (Johnson and Foote, 1988). It is one thing to agree that a module should be understandable; it is another to know how to design one that is. In this volume, there are several examples of models that have utilized the object-oriented paradigm to implement genericness and modularity. This reflects a major shift towards the use of object-oriented programming in both agricultural and ecological modeling (e.g. Baveco and Lingman, 1992; Bossel, 1991; Folse et al., 1989; Kolstrom, 1991; Larkin et al., 1988; Sekine et al., 1991; Sequeira et al., 1991, 1997). As this trend in agriculture and ecology continues, we strongly recommend close cooperation with software engineers. Because of the close link between efficient programming and efficient science, we believe that this will facilitate the adoption of the high standards of modularity and genericness that we advocate.

Acknowledgements

The authors wish to thank Mary Acock, George Hess, and Yakov Pachepsky for valuable comments on the manuscript, and Roger Whitney for bringing the work of Meyer to our attention. This research was supported by Interagency

Agreement DE-AI05-88ER69014, DOE grant DE-FG05-92ER61493, U.S.D.A. Specific Cooperative Agreement 58-1270-3-070, and NSF grants DEB 9524058 and DEB 92-40261 (Jornada LTER). Additional support provided by the Forest-Atmosphere Carbon Transfer and Storage (FACTS-1) project at Duke University under DOE contract numbers DE-AC02-76H00016 at Brookhaven National Laboratory and DE-FG05-95ER62083 at Duke University.

References

Acock, B. and Reddy, V.R., 1997. Designing an object-oriented structure for crop models. Ecol. Model., 94: 33–44.

Acock, B. and Reynolds, J.F., 1989. The rationale for adopting a modular generic structure for crop simulators. Acta Hortic., 248: 391–396.

Acock, B. and Reynolds, J.F., 1990. Model structure and database development. In: R.K. Dixon, R.S. Meldahl, G.A. Ruark and W.G. Warren (Editors), Process Modeling of Forest Growth Responses to Environmental Stress. Timber Press, Portland, Oregon.

Baveco, J.M. and Lingman, R., 1992. An object-oriented tool for individual-oriented simulation: host-parasitoid system application. Ecol. Model., 61: 267–286.

Bossel, H., 1991. Modelling forest dynamics: moving from description to explanation. Forest Ecol. Manage., 42: 129–142.

Chen, J.-L. and Reynolds, J.F., 1997. GePSi: A generic plant simulator based on object-oriented principles. Ecol. Model., 94: 53–66.

Folse, L.J., Packard, J.M. and Grant, W.E., 1989. AI modelling of animal movements in a heterogeneous habitat. Ecol. Model., 46: 57–72.

Forrester, J.W., 1970. Systems analysis as a tool for urban planning. IEEE Trans. Syst. Sci. Cybernetics, 6: 259–265.

Houghton, J.T., Jenkins, G.J. and Ephraums, J.J. (Editors). 1990. Climate Change: The IPCC Scientific Assessment. Cambridge University Press, Cambridge, 365 pp.

IGBP. 1990. International Geosphere-Biosphere Programme. A Study of Global Change: The Initial Core Projects. International Council of Scientific Unions, Stockholm, Sweden.

Innis, G.S., Noy-Meir, I., Gordon, M. and Dyne, G.M.V., 1980. Total system simulation models. In: A.I. Breymeyer and G.M.V. Dyne (Editors), Grasslands, Systems Analysis and Man, IBP Synthesis Vol. 19. Cambridge University Press, Cambridge.

IPCC, 1991. Intergovernmental Panel on Climate Change: The IPCC Response Strategies. Island Press, Covelo, CA. 270 pp.

Johnson, R.E. and Foote, B., 1988. Designing reusable classes. J. Object-Oriented Programming, (June/July): 2–35.

Joyce, L.A. and Kickert, R.N., 1987. Applied plant growth models for grazinglands, forests and crops. In: K. Wisiol and J.D. Hesketh (Editors), Plant Growth Modeling and Resource Management. CRC Press, Boca Raton, FL.

Kolstrom, T., 1991. Modelling early development of a planted pine stand: an application of object-oriented programming. Forest Ecol. Manage., 42: 63–78.

Larkin, T.S., Carruthers, R.I. and Soper, R.S., 1988. Simulation and object-oriented programming: the development of SERB. Simulation, 52: 93–100.

Lemmon, H. and Chuk, N., 1997. Object-oriented design of a cotton crop model. Ecol. Model., 94: 45–51.

LIDET. 1995. Long-Term Intersite Decomposition Experiment Team. Meeting the challenge of long-term, broad-scale ecological experiments. LTER Network Office, Seattle, WA.

Luo, Y., Chen J.-L., Reynolds, J.F., Field, C.B. and Mooney, H.A., 1997. Physiological adjustments cause disproportional increases in photosynthesis and plant biomass in elevated CO_2: a case study in a California grassland with the plant simulator GePSi. Functional Ecol., in press.

McMurtrie, R. and Wang, Y.-P., 1993. Mathematical models of the photosynthetic response of tree stands to rising CO_2 concentrations and temperatures. Plant Cell Environ., 16: 1–13.

Melillo, J.M., Callaghan, T.V., Woodward, F.I., Salati, E. and Sinha, S.K., 1990. Effects on Ecosystems. In: J.T. Houghton, G.J. Jenkins and J.J. Ephraums (Editors), Climate Change: The IPCC Scientific Assessment. Cambridge University Press, Cambridge.

Meyer, B., 1990. Object-Oriented Software Construction, 2nd edition. Prentice-Hall, Englewood Cliffs, New Jersey.

Reynolds, J.F. and Acock, B., 1985. Predicting the response of plants to increasing carbon dioxide: a critique of plant growth models. Ecol. Model., 29: 107–129.

Reynolds, J.F., Acock, B., Dougherty, R. and Tenhunen, J.D., 1989. A modular structure for plant growth simulation models. In: J.S. Pereira and J.J. Landsberg (Editors), Biomass Production by Fast-Growing Trees, NATO ASI Series, Applied Science. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Reynolds, J.F., Acock, B. and Whitney, R., 1993. Linking CO_2 experiments and modeling. In: E.-D. Schulze and H.A. Mooney (Editors), Design and Execution of Experiments on CO_2 Enrichment, Report No. 6. Ecosystems Research Series of Environmental Research Programme, Commission of the European Communities, Brussels.

Reynolds, J.F., Bachelet, D., Leadley, P. and Moorhead, D., 1986. Assessing the effects of elevated carbon dioxide on plants: toward the development of a generic plant growth model. U.S. Department of Energy: Washington, D.C.

Reynolds, J.F., Kemp, P.R., Acock, B., Chen, J.-L. and Moorhead, D.L., 1996. Limitations, uncertainties and challenges in modeling the effects of elevated CO_2 on plants and ecosystems. In: G. Koch and H.A. Mooney (Editors),

Carbon Dioxide and Terrestrial Ecosystems. Academic Press, San Diego.

Reynolds, J.F. and Leadley, P.W., 1992. Modeling the response of Arctic plants to changing climate. In: F.S. Chapin, III, R. Jefferies, J.F. Reynolds, G. Shaver and J. Svoboda (Editors), Arctic Physiological Processes in a Changing Climate. Academic Press, San Diego, CA.

Ryan, M.G., Hunt, E.R., Ågren, G.I., et al., in press. Comparing models of ecosystem function for temperature conifer forests and grasslands. In: J.M. Mellilo and A. Breymeyer (Editors), Global Change Effects on Coniferous Forests and Grasslands. Wiley, London.

Sekine, M., Nakanishi, H., Ukita, M. and Murakami, S., 1991. A shallow-sea ecological model using an object-oriented programming language. Ecol. Model., 57: 221–236.

Sequeira, R.A., Olson, R.L. and McKinion, J.M., 1997. Implementing generic, object-oriented models in biology. Ecol. Model, 94: 17–31.

Sequeira, R.A., Sharpe, P.J.H., Stone, N.D., El-Zik, K.M. and Makela, M.E., 1991. Object-oriented simulation: plant growth and discrete organ to organ interactions. Ecol. Model., 58: 55–90.

Tenhunen, J.D., Reynolds, J.F., Rambal, S., Dougherty, R. and Kummerow, J., 1989. A physiologically-based growth simulator for drought adapted woody plant species. In: J.S. Pereira and J.J. Landsberg (Editors), Biomass Production by Fast-Growing Trees, NATO ASI Series, Applied Science. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Vangrinsven, H.J.M., Driscoll, C.T. and Tiktak, A., 1995. Workshop on comparison of forest-soil-atmosphere models: preface. Ecol. Model, 83: 1–6.

VEMAP, 1995. Vegetation/ecosystem modeling and analysis project: Comparing biogeography and biogeochemistry models in a continential-scale study of terrestrial ecosystems responses to climate change and CO_2 doubling. Global Biogeochemical Cycles, 9: 407–437.

Webster, 1983. Webster's Ninth New Collegiate Dictionary. Merriam-Webster, Springfield, Mass.

Whisler, F.D., Acock, B., Baker, D.N., et al., 1986. Crop simulation models in agronomic systems. Adv. Agron., 40: 141–208.