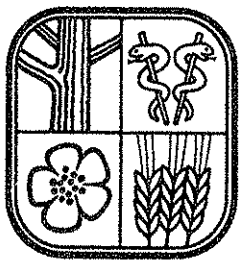

SVERIGES
LANTBRUKSUNIVERSITET

A Base Model for Discrete Event Simulation of Field Operations in Agriculture using Simula and Demos

Åke Axenbom

Institutionen för lantbruksteknik

Swedish University of Agricultural Sciences
Department of Agricultural Engineering

Rapport 143
Report
Uppsala 1990
ISSN 0283-0086
ISBN 91-576-4295-8

DOKUMENTDATABLAD för rapportering till SLU:s lantbruksdatabas LANTDOK, Svensk lantbruksbibliografi och AGRIS (FAO:s lantbruksdatabas)

Institution/motsvarande	Dokumenttyp	
Sveriges lantbruksuniversitet Inst för lantbruksteknik 750 07 Uppsala	Rapport	
	Utgivningsår 1990	Målgrupp Alla

Författare/upphov

Åke Axenbom

Dokumentets titel

A Base Model for Discrete Event Simulation of Field Operations in Agriculture using Simula and Demos

Ämnesord (svenska och/eller engelska)

machinery, equipment, labor, planning, scheduling, management, programming, operations, research, simulation, optimization, field operations, field work, systems analysis, systems dynamics

Projektnamn (endast SLU-projekt)

Serie-/tidskriftstitel och volym/nr	ISBN / ISSN
Sveriges lantbruksuniversitet, Instituttionen för lantbruksteknik. Rapport 143	ISBN 91-576-4295-8 ISSN 0283-0086

Språk	Smf-språk	Omfång	Antal ref.
Engelska		49 s + bil	31

Postadress	Besöksadress	Telefonnummer	Telefax
SVERIGES LANTBRUKSUNIVERSITET Ultunabiblioteket Förvärvssektionen/LANTDOK Box 7071 S-750 07 UPPSALA Sweden	Centrala Ultuna 22 Uppsala	018-67 10 00 vx 018-67 10 98 018-67 10 97	018-301006

Table of contents

PREFACE

This report describes a general package for discrete event simulation of field operations. The work is originally a side effect from the development of an integrated model of hay growth, harvesting and barn drying, a project financed by the Swedish Council for Forestry and Agricultural Research.

During the development work the model was found to be applicable to a wider range of problems. In particular, it was needed for the investigation of systems for harvesting of fuel straw. Consequently, the development of the base model into a stand-alone product has been financed by the Swedish Energy Administration.

The base model is strongly influenced by works by Ewoud van Elderen, IMAG. The reason not to use his model was the need for a simpler approach. Yet the concept and terminology is to a considerable extent adopted from van Elderen's models.

I am indebted to Dr. Anders Almquist for the patience shown during the final development of the model, during which time other important research within bio-energy have been set aside. I am also indebted to professor Jean-Marie Attonaty, for his hospitality at the very creative research environment at Station d'Économie et Sociologie Rurales, Paris-Grignon, during two intensive months when the base model was developed.

SUMMARY

Within the research project "Optimization of hay harvesting systems" a mathematical model of the entire hay production and utilization process was developed in order to enable the solution of dimensioning and management problems by means of simulations rather than making full-scale experiments.

As a part of this project, a model of field operations and management had to be developed, since no existing model fulfilled the demands on high resolution, combined with simplicity to use.

A comparison between linear programming (LP) and simulation showed that only the latter technique could yield the resolution necessary for accurately modelling the behaviour of the man-machine system, as well as of most biological processes that might be simulated along with the process of field operations.

Since the man-machine system is a typical discrete event driven one, whereas most biological processes are continuous, the model was designed so as to enable so called combined simulation, i.e. discrete event simulation combined with continuous simulation.

The model of field operations and management was given a multi-layer design, and was implemented in the object oriented programming language Simula. The simulation package DEMOS was used as the discrete event modelling environment.

The model design was kept as simple as possible. It mainly consists of three types of actors: the fields, the gangs, and the manager.

The fields contain exactly one material each. The gangs are different combinations of men and machinery and a process, converting the material of the field being processed into another field. The manager takes decisions about starting and stopping gangs.

There are also other actors: one keeping track of the starting time of the day, one keeping track of the hours for dawn and dusk, and one keeping track of the hours for rain starts and stops. The individual items of men and machinery however do not take actions of their own.

The package was designed as a base model, useful for simulating field operations in general. Specific applications are developed by means of further specifying properties of the actors included in the model. The discrete event sequencing is completely hidden within the model so that the user do not have to bother about it.

In order to maintain a flexible system, no decision model was included. This should be entered in the specific application.

The base model was validating by means of developing a test bed program which utilized the essential parts of the base model. The results were investigated in order to examine whether the program executed correctly.

The results showed that the synchronizations between all the actors involved in the simulation worked as expected.

It was concluded that the resolution of the model is high enough for most problems. An exception would be certain transport systems, which would require simulation of the position on the field or on the road for every vehicle.

The efficiency of the program with respect to computing speed showed to be satisfactory, but the memory requirements might limit the maximum problem size. It is however possible to execute realistically large problems on a personal computer under the MS-DOS operating system.

1 INTRODUCTION

1.1 Background

In 1981 the research project "Optimization of hay harvesting systems" was started with the objective of constructing one mathematical model of the entire hay-making process.

The model was built-up by several sub-models. The main parts were the growth model, the forage harvesting and conservation model and the hay to milk conversion model.

Again, the forage harvesting and conservation model was divided into several parts, for example a field drying model, a field loss model and a barn drying model. Another of the sub-models necessary to build up this large model would be a model of field operations and management (see figure 1).

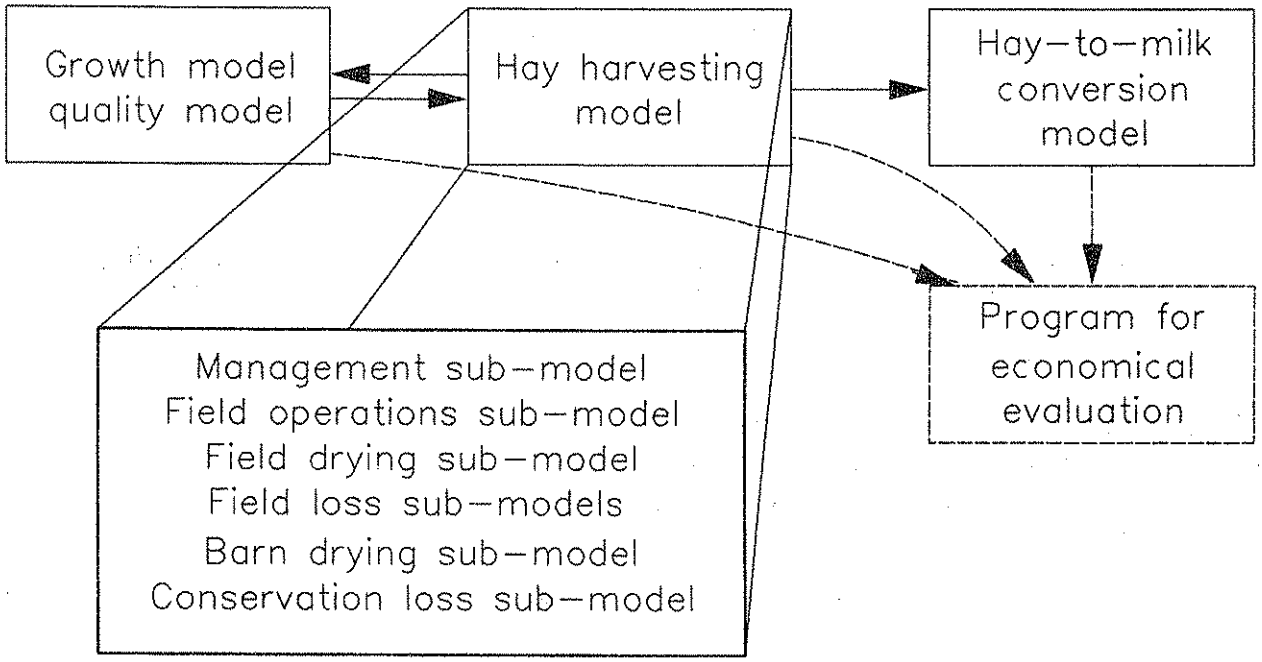

Figure 1. The structure of the hay production and utilization model. Source: Axenbom (1988).

For the model as a whole, as well as for each of the sub-models, the following requirements were set up:

- high resolution,
- modular design.

The following report concentrates on the development and validation of the field operations model and on the management (decision) model, while the model as a whole has been described in Axenbom (1988).

1.2 Problem

The overall purpose of the development of the hay production and development model was to solve dimensioning and management problems by means of experimentation with the model rather than making full-scale experiments.

It was evident that the man-machinery-system was a critical part of the model, since the actions taking place actually controls the entire harvesting process. Therefore it was necessary to pay certain attention to the field operations model with respect to its correctness and resolution.

Very little work has been done on this type of models. The only work living up to the resolution requirements was the works by van Elderen (1977; 1987). The latter work was however not available at the time the main work on this project was performed. Also, his model proved to be too large to be possible to execute on a personal computer.

Consequently, there did not exist any model of field operations and management that could be used in the hay production and utilization model.

1.3 Objective

The main objective was to construct a model able to simulate field operations with a resolution high enough to be useful in the hay production and utilization model.

This means that the model was required to consider

- the setup time and teardown time of man-machine systems,
- the effects of different management strategies,
- the consequences of events such as starts and stops of rainfalls,
- the availability of men, machinery and fields.

Some requirements on the design were also set up for the field operations model:

- modularity,
- compactness,
- generality,
- easy to use.

The objective behind these was the aim of being able to easily adapt the model for other applications than hay-making, without making it too large to be possible to use on personal computers.

1.4 Earlier work on field operations models

Work has been done on models of the entire hay harvesting system by for example Parke et al (1978), recently improved by McGechan (1986). Similar approaches have

been made by Lovering & McIsaac (1981) and by Savoie et al (1982). They all used a constant time step of one hour. Cunney & Von Bargen (1974) used a time step of half a day.

Discrete event models have not been much used in simulating field operations. The only examples found are Verma et al (1986) simulating forage harvesting experiments using GPSS, Chen et al (1976) simulating cucumber harvesting using Simscript II, Jonsen (1984) simulating the driving pattern on the field using Simula, Kindler et al (1983) simulating transport systems using Simula, and van Elderen (1987) simulating combine harvesting, also using Simula.

Personal communications with Jonsen (1983), Kindler (1984) and van Elderen (1984) confirmed that a simulation model of field operations should be made using a discrete event simulation language, since the development of an own discrete event simulator is a tedious and error-prone task.

1.5 Earlier work on decision models in agriculture

1.5.1 Heuristic concepts

A heuristic decision model (or strategy) consists of an algorithm or a set of rules which does not maximize or minimize any objective function in a true mathematical way. Therefore, its ability to find a good solution is not guaranteed, but dependent on the design of the model and of the problem.

Two entirely different heuristic concepts will be briefly covered here. They will be called

- the urgency concept,
- the decision rule concept.

Which concept to choose is dependent of the objective of the model. The two apparent alternative objectives of the decision model are:

- to model "real-world" decision behaviour,
- to find the optimal decisions.

Daily farm management is a typical application of decision-making under uncertainty. The major sources of uncertainty are the future weather, and the risk for machinery breakdowns and accidents. In addition, the decision-maker generally does not have exact information about the current field state. Nor is he able to predict its development with time. This is particularly valid for biological factors such as moisture content, ripeness etc.

The basis for decisions under uncertainty is that each alternative decision yields a probability distribution of the result, rather than a single value.

If the probability distribution can be transformed into a discrete number of possible events, then the manager's scheduling problem is appropriately described as a decision tree (see figure 2). In such a graph the possible sequences of decisions and outcomes are drawn, typically for a planning period including several decision points.

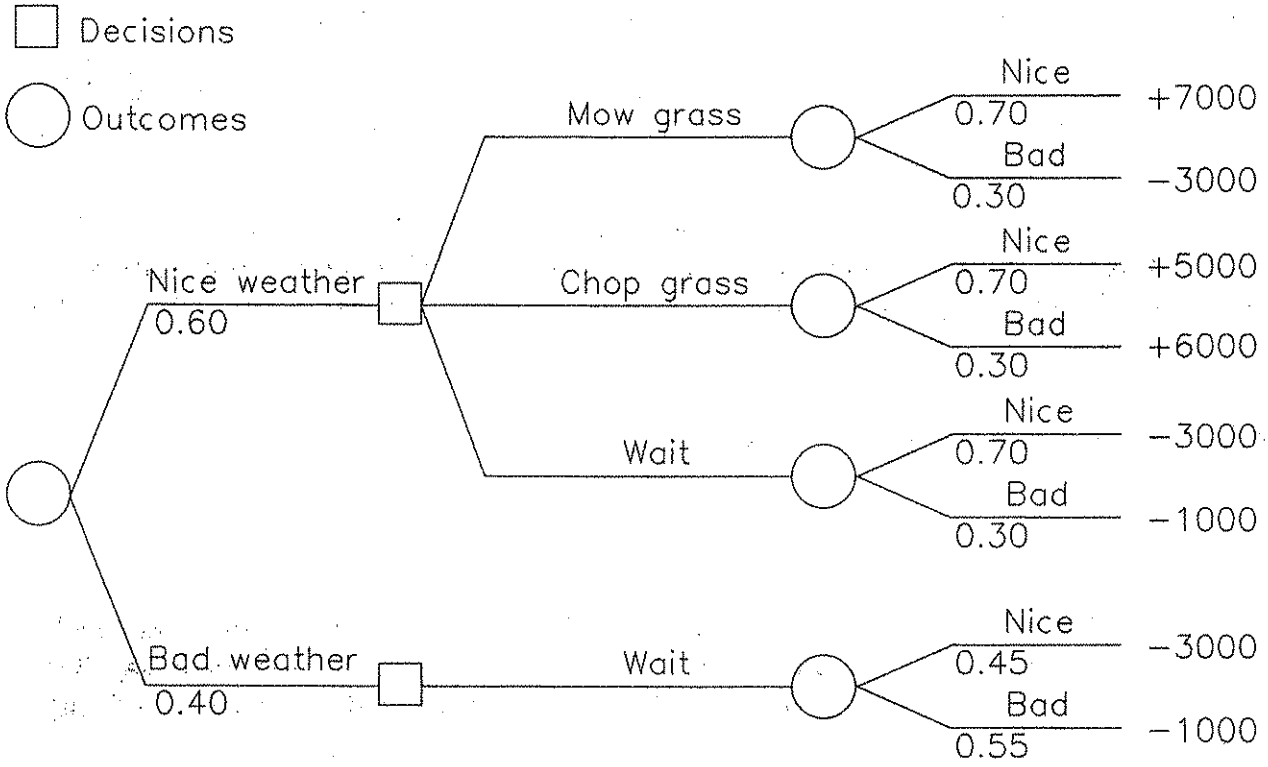

Figure 2. A simple decision tree of harvesting of wilted silage, with examples of probabilities and expected values.

Calculating the expected value of the alternatives at hand is straightforward. The expected value of each of the 'levels' in the decision tree is evaluated, and multiplied with the probabilities all the way from the current decision point. These expected values are summed for each alternative in the current decision point.

1.5.2 Optimizing concepts

Ranking of alternative actions under uncertainty means comparing the probability distributions of the expected consequences of them. There are at least two different ways of doing this.

The first one means that the entire curves are compared. If the cumulative probability curve of alternative A is to the right of the curve of alternative B over all their lengths, then A is said to be stochastically dominant to B. See figure 3.

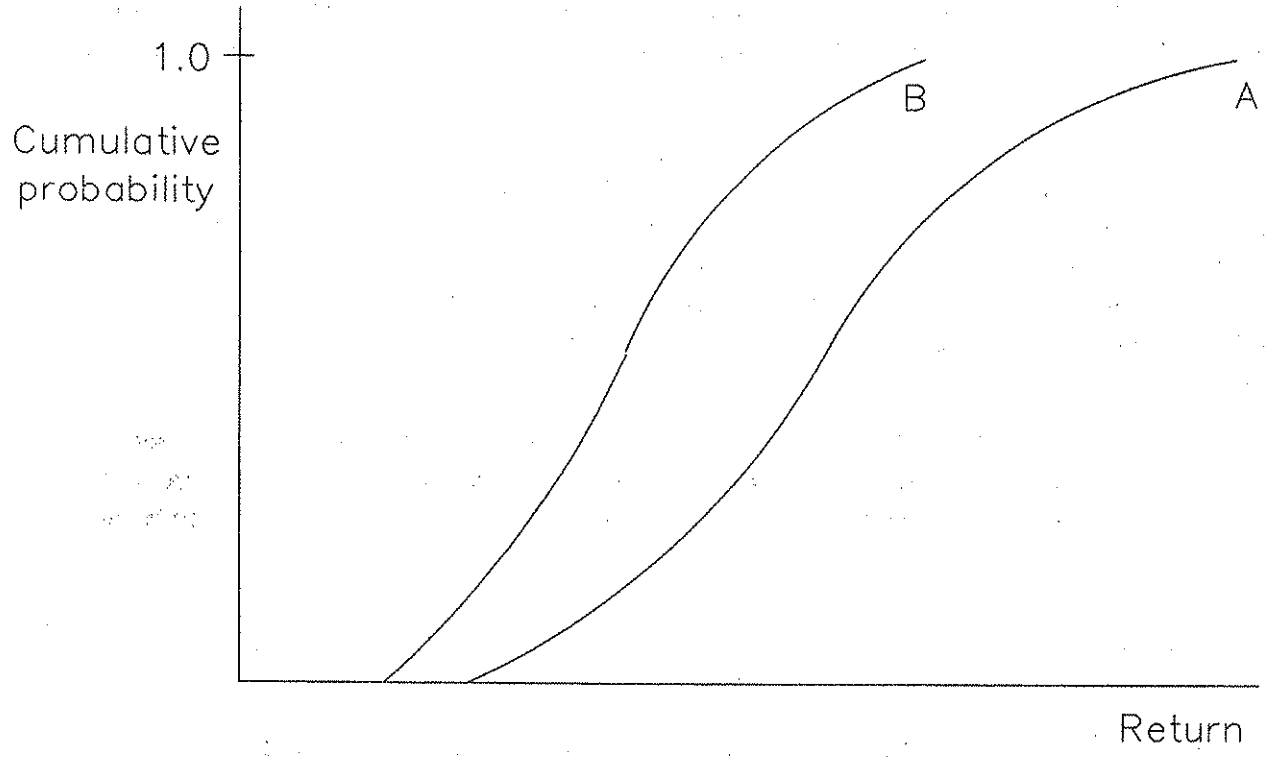

Figure 3. First-degree stochastic dominance. The alternative A is statistically dominant to alternative B.

The drawback with the stochastic dominance criterion is that it is not always possible to rank the alternatives. Russell et al (1982) states that the first-degree stochastic dominance criterion is quite useless for this reason, but that the second-degree stochastic dominance criterion, achieved by integrating the cumulative probability curves, has proved being able to present a unique choice in most cases, thus proving its usefulness.

The second way is to compare the expected (mean) value of the distribution. The simplest strategy of this kind is to choose the alternative with the highest expected value. For a risk-neutral decision-maker, relying in that good and bad outcomes take out each other in the long run, this is the optimal strategy.

The typical farmer is however not risk-neutral, since he possesses only a limited amount of capital. Therefore, he tends to choose a strategy decreasing the risk for very bad results, to the price of a slightly lower average result. In non-risk-neutral cases, it is therefore not appropriate just to compare the expected results. The variation or distribution must also be considered.

One conventional way of solving this problem is mean-variance (E-V) analysis. Herath (1982) however claims that the "maximum expected utility" criterion is the best one for decision-making under uncertainty. The latter method utilizes a "utility function" in conjunction with the probability distribution, to calculate the expected utility of each

alternative. The utility function transforms the probability distribution from a function of value into a function of utility. The major problem with this method is the difficulty of determining the shape of the utility function.

Gustafsson (1981) presents a practical method, called Split-simulation, to examine a decision tree by means of simulation, using Simula. All the branches of the tree are simulated simultaneously. This means that each path has to be simulated just once. At each node, the model is split into a number of identical copies, one for each path. To each copy is assigned a splitvalue, reflecting the probability of that branch being randomly selected. By omitting branches with very low splitvalues, the computer memory requirements are said to be reasonable.

Whan et al (1976) report that a Markov decision process may be solved using Linear Programming (LP). They also show that the computer storage requirements may be significantly reduced if Howards Policy Iteration Algorithm, a special case of the Simplex method, is used instead of the ordinary Simplex algorithm.

1.5.3 The urgency concept

van Elderen (1977) developed a heuristic strategy for scheduling of farm operations. The concept is that the operations with the highest urgency are to be performed first. The central factors involved in the calculation of the urgency are the timeliness effect and the capacity.

The urgency for performing a certain operation on a certain material (i.e. a crop) is affected by the timeliness effect of that material as well as of the other materials, the processing of which are delayed as a result of the choice of operation.

Every time an operation is finished, or some other event happens that might affect the urgency values, these are recalculated.

1.5.4 The decision rule concept

The most commonly used decision model for scheduling field operations first classifies the current workability of the field depending on the actual and preceding weather, and then ranks the possible operations due to some pre-determined priority order, see for example Konaka (1974), Cunney & Von Bargen (1974) or Parke et al (1978). The testbed program described below is another example of a simple rule-based decision model.

If the maximum number of gangs simultaneously active is very small, the number of possible combinations of gangs is also typically small. In that case the priorities may be programmed directly (as in the testbed program).

If however the number of possible combinations of gangs is very large, every operation should be assigned a value reflecting the importance of performing it. Summing these values for each combination of gangs makes it possible to rank the alternatives. (This is the principle used in the urgency concept.)

Sometimes a work-period (e.g. the autumn) can be regarded as a sequence of "sub-periods", each one having a relatively simple set of decision rules and a criterion for the period being finished. For such problems, decision-rules based on direct rules are more useful than otherwise.

An interesting development would be to design the strategy in such a way that the decision rules could be entered as indata. Such work is currently under development in France (Attonaty, 1988).

1.6 Systems analysis

In most agricultural operations, decisions and field operations on one hand and the weather and the state of the biological material on the other are highly interrelated. For example, in haymaking drying may be accelerated by a tedding operation. But the impact of the tedding depends on the moisture content of the hay, and its distribution within the swath, which in turn depends on the historical weather.

Obviously, a decision concerning field operations is preceded by collecting information about the current state of the fields.

Discrete event simulation models are used in particular for modelling different types of queuing systems. Since all future events are "booked" on an event list, the correct computing order when updating parallel processes is automatically maintained, contrary to constant-step simulation models. Furthermore, discrete event models are considerably faster, since time intervals during which nothing happens are actually simulated in zero time. See figures 4 and 5.

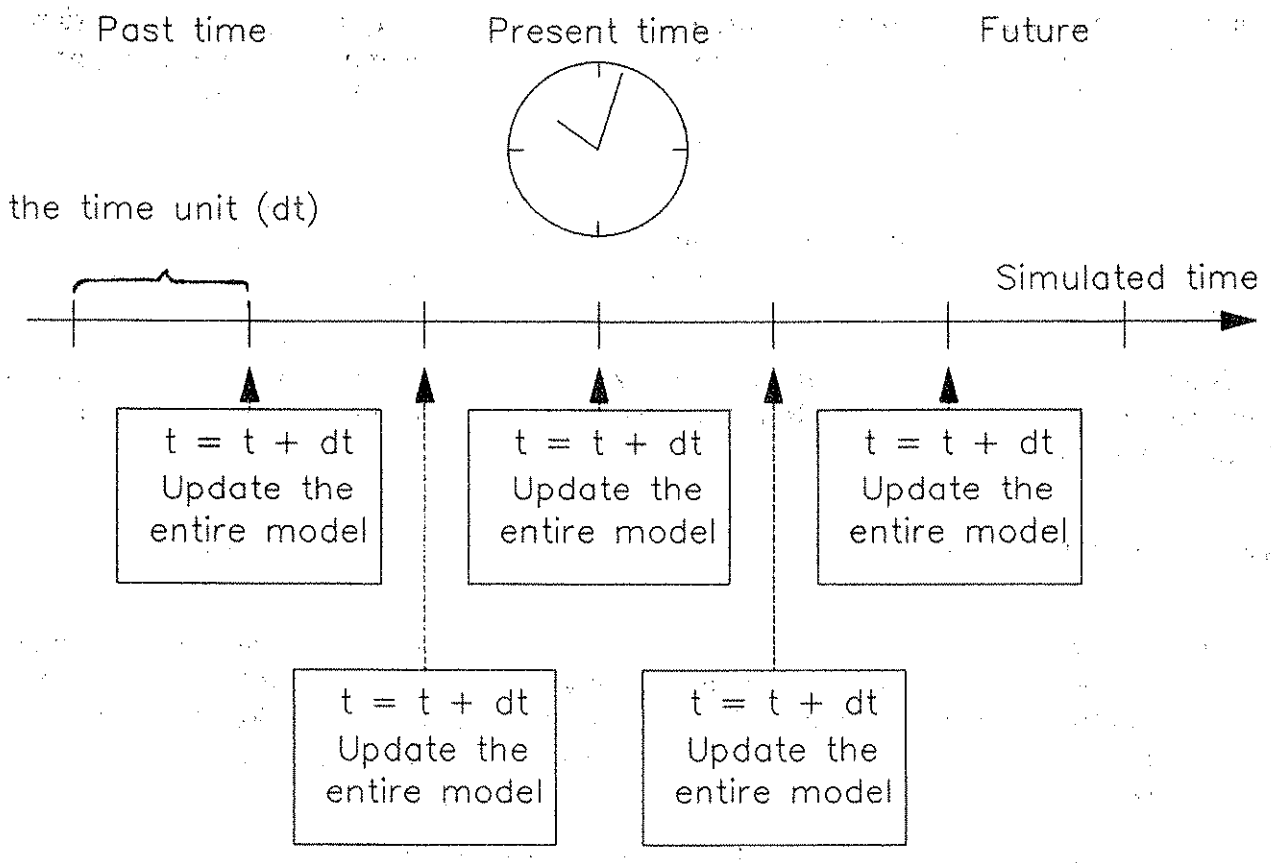

Figure 4. The principle of constant-step simulation. The simulated time is incremented in steps of constant length. Source: H. Islo, SIKOB.

Linear programming models, as well as constant timestep simulation models, restructures the actual processes into periods. It is obvious that this inevitably leads to severe sacrifices in resolution, unless the periods are made very short. And with the number of periods, the LP matrix grows and the execution time gets longer. LP models have the additional disadvantage of being completely restricted to linear equations.

These two properties make LP inappropriate for complicated dynamic models. The most appropriate model design would be a simulation model.

The man-machine system is a typical discrete-event-driven system, whereas the biological process in general should be characterized as a continuous one. Modelling the entire model as a continuous model would require very short periods if the resolution would not suffer. It would be possible also to design everything within an event-driven context, but it would create computational overhead and enforce a special design of the continuous model. It was therefore decided to use the technique of "combined simulation", where discrete event simulation is used for the man-machine system, while continuous simulation would be used for the biological models.

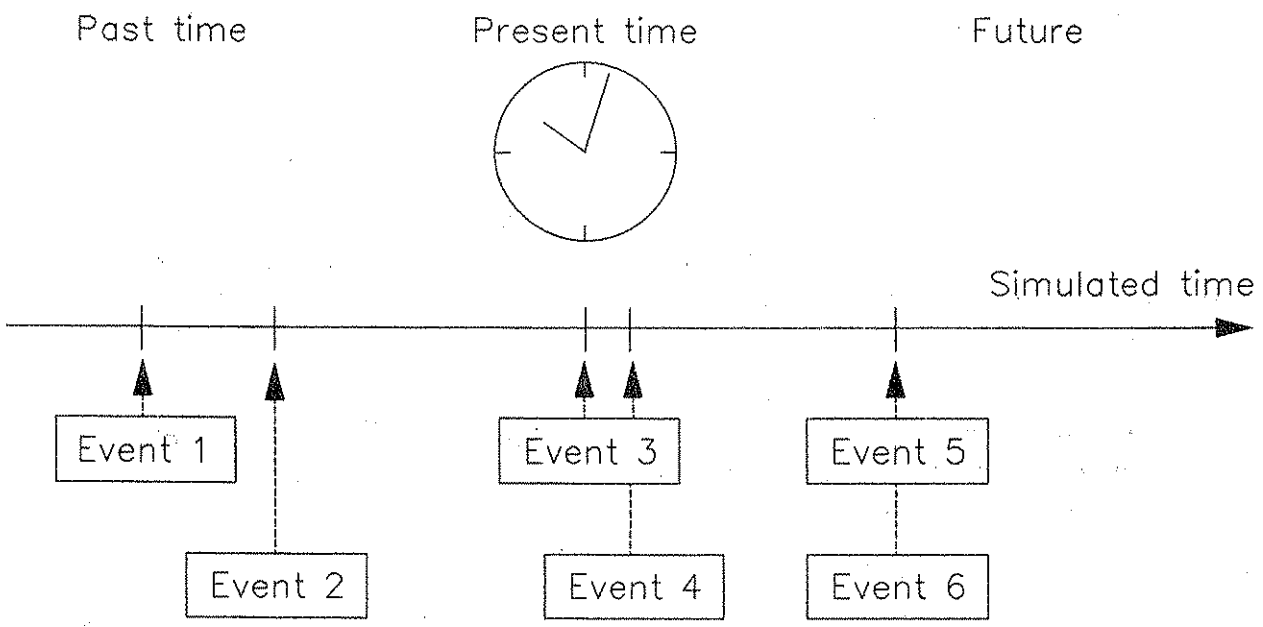

Figure 5. The principle of discrete event simulation. The simulated time is incremented in time steps of variable length, equalling the time interval between two subsequent events. Source: H. Islo, SIKOB.

Since the field operations are started and stopped due to a decision, it was naturally to place the field operations model as the "centre" of the entire model. The different biological models would be attached to, and executed from the field operations model.

2 MODEL DEVELOPMENT

2.1 Choice of programming language

There are a multitude of discrete event modelling packages on the market, see for example Franta (1977) or Savén (1988). In this case it was necessary to be able to attach continuous models of growing and barn drying to it. Thus the programming environment must permit ordinary programming in some 3rd generation language.

The choice fell on Simula. This is a general-purpose, structured 3rd generation programming language, yet with all discrete event modelling facilities available. Among the discrete event modelling packages available within Simula, DEMOS (Birtwistle, 1979) was chosen because of its power, simplicity, adaptability and portability.

Simula's "class" concept, encouraging an object-oriented style, was a pro as well as the highly portable code. A con was however that no MS-DOS based compiler was available at the time the work was started (1984). Such a compiler was however released in 1988 (Simula A.S., 1988) and has been used during the final development phase of this package.

2.2 General concept

2.2.1 Modularity and generality by means of a multilayer design

The term "field operations" corresponds to an enormous class of different types of equipment and work methods. As the word "class" implies, there exists a certain degree of similarity between all (or most) field operations. Consequently, models of different types of field operations may be designed with a certain degree of similarity.

It would be a waste of resources not to utilize these similarities in model design. By extracting the common, general part from the problem-dependent one, the general parts could be used as a basis for different models. This technique is known as "multilayer" programming.

As the name implies, the model is implemented as a number of logical layers, where the bottom layer is the programming language and the top layer is the application. One can say that the programming language is enriched with the definition of each layer. van Elderen (1987) demonstrated the usefulness of the multilayer technique in modelling farm operations.

Since the author believes that many different mechanization problems may be conveniently and appropriately described on the basis of a base model, it was decided to choose the multilayer design. It should be noted that such a programming style is strongly encouraged in Simula.

2.2.2 Generality by means of a flexible system representation

This aim has been realized by completely describing the structure of the farm by means of indata.

2.2.3 Compactness by means of a simple design

When this is written, one of few ways to get your computer program into use is to implement it under the MS-DOS operating system. Unfortunately, the hard restrictions on memory availability under this operating system suits very badly in conjunction with languages making heavy use of dynamic memory, like for example Simula. This meant that the general base model had to be kept as compact as possible, otherwise there would be no memory left for the application.

The base model was designed for compactness in two ways. The first one was to keep the basic concept as simple and straightforward as possible, by reducing the number of actors to a minimum and by simplifying their actions to a certain degree.

The other way was to make as much data input as possible non-interactive. This saves large amounts of code otherwise spent on menu, window and error handling. The drawback is of course that an erroneous data file could cause spurious run-time errors. The indata files should therefore in the future preferably be generated by a special indata generating program, executed before the very simulation program.

2.2.4 Easy to use in different applications

The concept of pseudo-parallel execution made possible in Simula, and utilized in DEMOS, is invaluable in the development of correctly executing discrete event simulation models. However, since this technique has more in common with multi-tasking programs than with "traditional" (procedural) programming, it is familiar only to a few people within Agricultural Engineering and Economics.

Therefore, to facilitate the adapting of the model to different applications by means of extending the model, it was decided to embed the entire discrete event scheduling function into the base model, transparent to the user.

To be able to embed the discrete event interactions between the different actors (objects) of the model, the general behaviour of most actors concerning interactions with other actors had to be pre-defined.

2.3 An object-oriented model approach

The obvious way to cope with a multitude of interactions between the different actors in a discrete event system is to make every actor an object of its own. This programming technique, commonly called "object-oriented programming", is encouraged in Simula. The objects needed to describe this system are the following:

2.3.1 The gang.

This term, borrowed from Oving (1971) and also used by van Elderen (1977, 1987) is defined as the combination of items of equipment needed for executing one operation according to one specific method. Note that "equipment" should be interpreted in a broad sense, including labor as well as machinery.

Adopting their approach, it is the gang (a set of men and machinery), rather than the individual men and machinery, which performs the field operation.

The general task of a gang is
1) to acquire the men and machinery required, as well as a field,
2) to perform the field operation,
3) to release the men and machinery, and the field.

A field operation means a time-consuming process from the gang's point of view. From the field's point of view, however, the field operation means a conversion of the material of the field into another material.

The set of gangs available to a certain field operations model is defined as a combination of some of the men and machinery. Furthermore, since a gang is defined so as to perform only one operation on the field, it only accepts fields with a certain material and always converts it into another specific material.

To become active, a gang must not acquire more than the number of any category of men and machinery currently available. Also, there must exist at least one field containing the appropriate material.

These are the most important simplifications built into the gang:

- the teardown time is appended to the setup time,
- the setup time is independent of transport distance,
- the setup time is constant even if the gang has been recently active,
- two gangs cannot work simultaneously at the same field.

2.3.2 The men and machinery

Their task is simply to be resources for the man-machine system. They need not take actions of their own.

2.3.3 The fields

The fields have a common, simple task: to keep track of their own states, and to be available only to one gang at a time.

2.3.4 The manager

The manager is an abstract object. It is included since it is convenient to separate the decision-making role of the men from their role as labour. The manager object takes care of all decision-making. The process of decision-making may however be modelled in widely different manners, dependent on the design of the rest of the model, or dependent on the objective of the simulations.

In general, all decisions concerning starting or stopping of gangs should be done by the manager. However, it was found that the manager's task could be quite simplified by permitting the work to be interrupted without a decision in certain conditions. These conditions are:

- when it starts raining,
- when the time of day exceeds the absolutely last possible work time.

If any of these conditions occur while there are gangs working, these will be stopped automatically.

2.3.5 The environmental actors

The objects mentioned above are considered to describe the farm sufficiently well. However, also the environment affects the work in certain ways.

The calendar starts the work in the morning, and stops it in the evening.

The weather affects the work in two different ways. On one hand it affects the continuous processes, on the other it generates events that may be decision dates. Such events are when it starts or stops raining, and when the sun goes up or down.

2.4 Should a general decision model be included?

The main reasons not to include an general decision model within the base model is the aim to retain the compactness and flexibility of the package. The price for this is the burden of having to develop a decision model for each application where the base model is used. Even an extremely simple strategy does require substantial development time. With a more complex behaviour of the decision model, the development effort increases considerably.

An attractive alternative is to construct a general decision model, whose behaviour can be "programmed" by means of indata. Unfortunately, there is no obvious concept in which to accomplish this.

The decision made was not to include any general decision model.

3 DESCRIPTION OF THE BASE MODEL

Some knowledge of Simula and DEMOS is advantageous when reading this chapter. It is however believed that the main content is understandable also with a basic knowledge of programming in general, since most of the elements discussed are briefly explained in the text.

3.1 Activity diagrams and DEMOS

Pseudo-parallel, discrete event driven systems are conveniently described by activity diagrams (Birtwistle, 1979). Quite similar to flowcharts, they should not be too hard to understand. Since one major objective with activity diagrams is to define the synchronization between different objects, there are typically several flowchart-like structures in one activity diagram. To facilitate the comprehension of this chapter, there will however be one activity diagram for each type of entity. Furthermore, the diagrams will be complemented with some explaining text boxes.

The elements used in the activity diagrams are showed in figure 6.

The universal building block in DEMOS for objects taking actions of their own is the ENTITY. All ENTITY objects have the ability to stay in queues, execute instructions, acquire and release resources, interact with other ENTITY objects and so forth.

In this program, the fields, the gangs and the manager are all implemented as ENTITY objects. More specifically, they are declared as different "sub-classes" to the common "super-class" ENTITY. A sub-class always inherits all the properties of its super-class, unless explicitly redefined. Within class FIELD, GANG etc the attributes needed for each type of actor are defined. So the set of attributes are the ones declared and defined in the class itself, and the ones defined in its super-class. The attributes are typically implemented as variables or member procedures.

The ENTITY may COOPT another entity, i.e. take another ENTITY as a "slave" for a period.

A RES object is a kind of counter. It achieves its initial value (the number of e.g. some category of men or machinery) at its creation.

From a RES object called TRACTORS, a tractor could be acquired, "borrowed", by calling TRACTORS.ACQUIRE(1). An already acquired tractor would be released calling the procedure TRACTORS.RELEASE(1). (The argument denotes the number of items acquired or released.)

Some reasons not to simply use an integer variable instead of a RES object are

- improved modularity,
- data abstraction,
- error checking at runtime performed by the object itself,
- automatic updating of statistics.

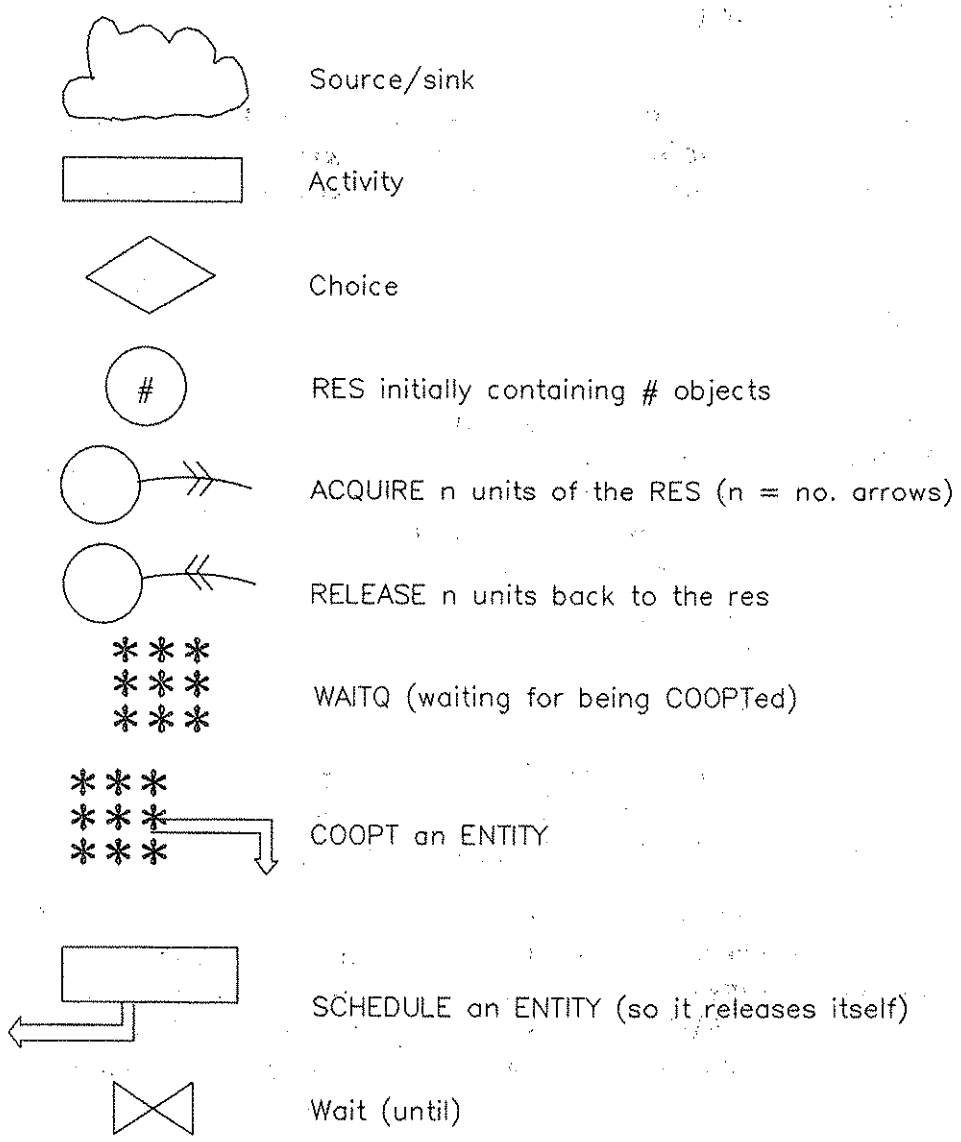

Figure 6. The elements used in the activity diagrams.

The dot between the object name TRACTORS and the procedure name should be interpreted as a genitive, denoting that the procedure is a member of the TRACTORS object. In object-oriented programming it is preferred to call an object's member procedure SomeObject.UpdateYourself rather than manipulating the object's attributes from outside. The aim is to keep the internal data organization of the object transparent to the environment. Therefore, inspections of an object's attributes are also made by means of procedure calls. For example, the current number of items available in a RES object is inspected through its member integer procedure Avail. (Member procedures and functions are sometimes called "methods". In this paper the term "member procedure" is used throughout.)

3.2 Implementation of the base model

3.2.1 Gangs

The complete algorithm of class GANG is showed in figure 7. Its general behaviour is: it acquires men and machinery and COOPTs a field (see below). Then it holds for the time it takes to perform the operation. Finally it releases the field and the men and machinery. After that, it becomes inactive until started again by the manager.

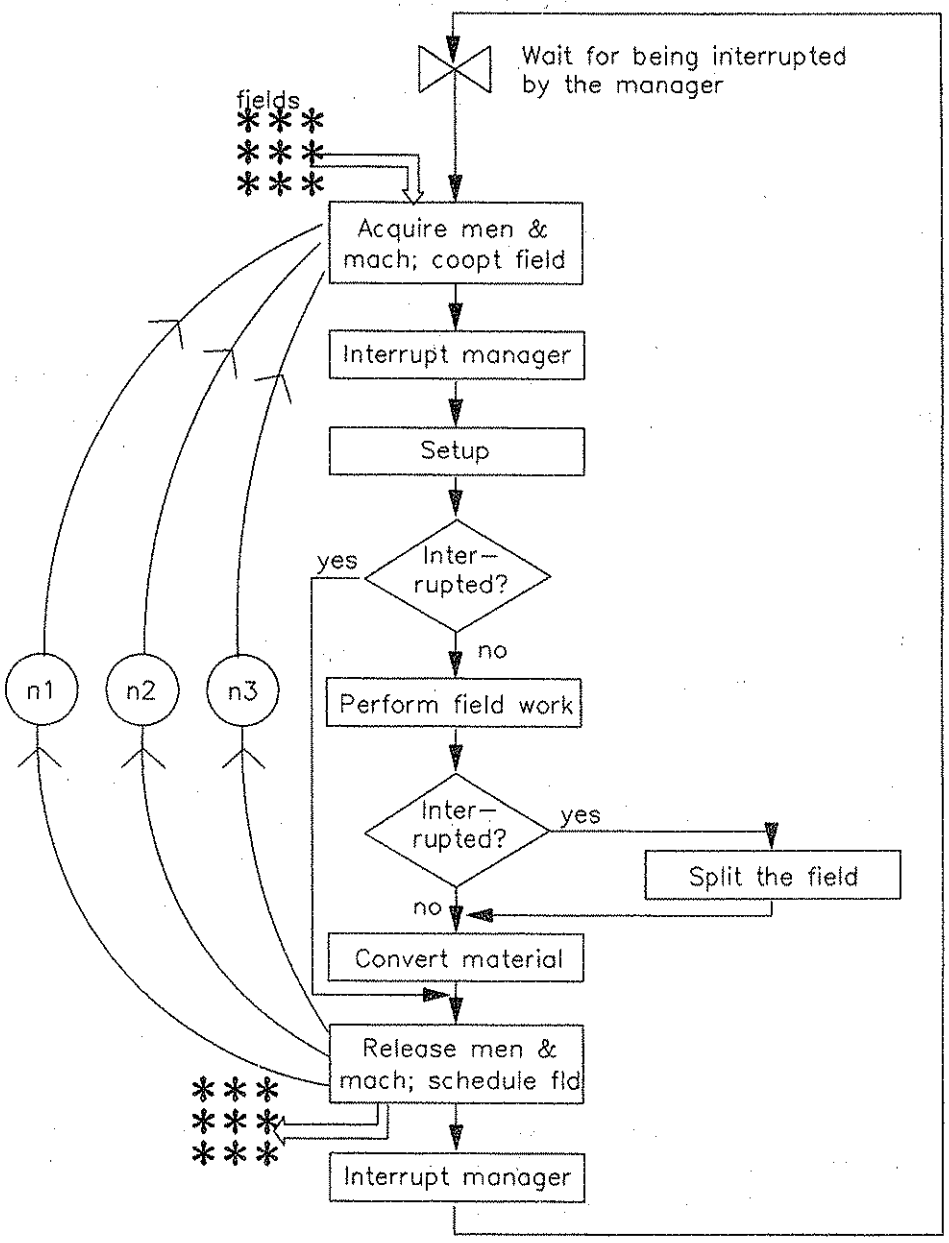

Figure 7. Activity diagram of the GANG object.

The data attributes of GANG objects are:

- gang title
- the number of categories of men and machinery used by the gang
- the categories (their ordinal number)
- the number needed of each of these categories
- mtrl_processed (i.e. the required initial MATERIAL of the field to be processed by this gang)
- mtrl_delivered (i.e. the final MATERIAL of the field processed by this gang)
- process name (a text string used in reports)
- setup time, teardown time and normal work time/ha
- a reference to the field currently processed (NONE if the gang is not active)

and a state variable having as value either LAZY, SETTING_UP or WORKING. (Since teardown time is appended to setup time, the state goes immediately from WORKING to LAZY when work is finished.

The GANG objects take actions by themselves and interact with fields and with the manager. Therefore class GANG is implemented as a subclass to class ENTITY.

There are three member procedures of class GANG. The boolean procedure MACH_AVAIL tests whether the men and machinery needed are available. The procedure FirstFieldAvail, returns a reference to the first field found, which is available to be processed by the particular gang. (If no field is available, NONE is returned.) Finally, the virtual procedure Capacity calculates the capacity of the gang.

3.2.2 Men and machinery

The only task of the men and machinery being to be available to the gangs, they take no actions of their own. Thus they are implemented using the simplest building blocks in DEMOS, as RES objects.

Since this package was designed to be as general as possible, the different RES categories have no predefined names (such as TRACTORS). Instead, an array of RES objects called MEN_MACH, having one element for each category of men and machinery, is created. If harrows were category 4, one harrow would be acquired by calling MENMACH(4).ACQUIRE(1).

3.2.3 Fields

A field contains exactly one material. The current material of the field is stored in the integer variable MATERIAL. For example, in haymaking the MATERIALs GROWING, MOWED, TEDDED and WINDROWED could be symbolized by the numbers 1, 2, 3 and 4. The meaning of different values of MATERIAL is not pre-defined, but must be specified in indata.

The fields normally stay in queues, one queue for each possible MATERIAL of the field. Thus a gang looking for a field with the appropriate MATERIAL need only look in one queue.

When being COOPTed (see figure 6) by a gang, the field is removed from the queue. When the operation is finished and the field is released, the field must place itself in the appropriate queue, which is typically not the same one as before, since its state has changed. See figure 8.

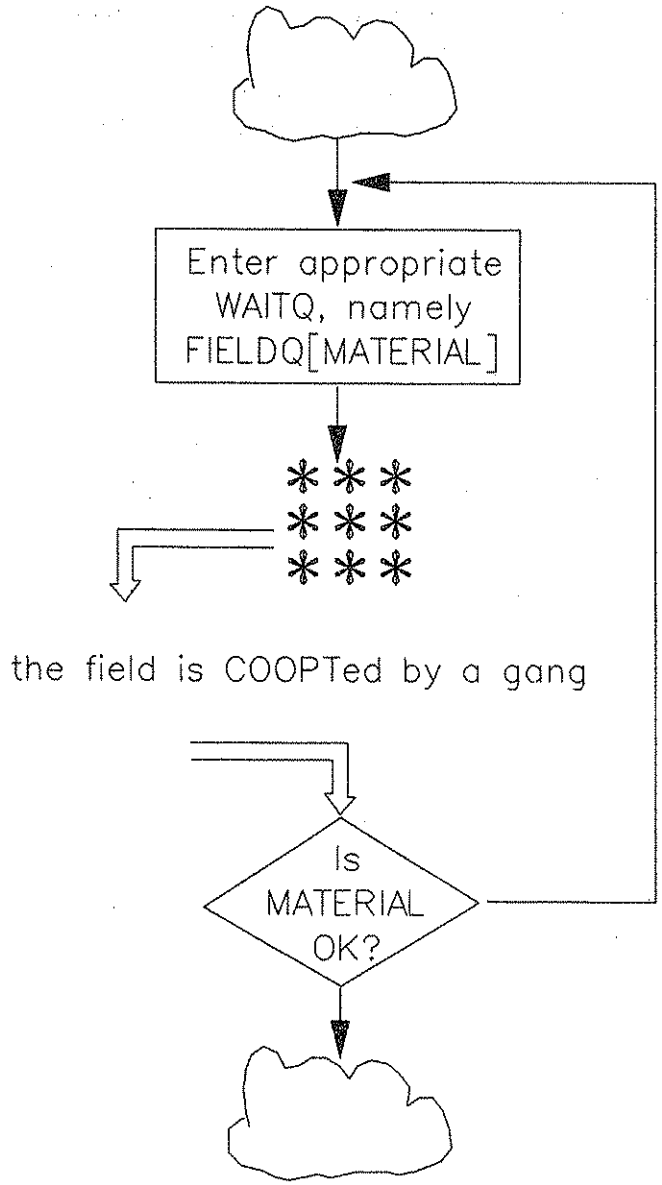

Figure 8. Activity diagram of the FIELD objects.

The attributes of the generic field objects are

- acreage (a real number)
- MATERIAL (an integer number)
- Split (a procedure dividing the field into two parts)
- ConvertMtrl (a procedure changing the value of MATERIAL)
- UpdateState (a procedure updating state variables)

The variable MATERIAL stores the current material of the field. It may be changed only by the procedure ConvertMtrl.

The field objects typically contain numerous state variables, depending on the application. Such variables, for example moisture content or yield, are not defined in class FIELD, since different problems require different state variables. Such additional attributes are appended by defining a sub-class to class FIELD in the specific application.

All the three procedures Split, ConvertMtrl and UpdateState are declared "virtual". This means that their contents may be redefined in different manners in different sub-classes. For example, the procedure UpdateState is empty in class FIELD, since there are no continuous variables to update. In a sub-class "unsown-field" it may be redefined to update certain tractability variables, in another sub-class "grain-field" it may be redefined to update the ripeness and moisture content of barley and wheat.

If a virtual procedure is not redefined in a sub-class, it retains the definition made in its super-class. If redefined, however, it loses all of its previous definition. The procedure ConvertMtrl should contain a call to the procedure UpdateState, and does so in class FIELD. If the former is redefined, the programmer must control that this call is made also in the redefined version.

3.2.4 The manager

Most events are decision dates. Whenever an object takes an action which generates a decision date, it calls the manager's member procedure INTERRUPT(n) to make him active. Since different events may require different types of decision, the integer argument (n) is given a certain value for each event classified as a decision date:

- a working day starts
- a field is processed and some men and machinery released
- the rain has stopped
- the sun has set
- a field is COOPTed and some men and machinery acquired.

The point of time when the setup is finished and the actual field work starts is not considered a decision date. Rather, the work goes on automatically.

The member procedure DetermineAction and the Boolean member procedure PeriodOver are both declared virtual and defined empty. The aim is that the decision strategy should be defined in a sub-class to manager, by means of redefining these procedures.

The other member procedures are

- StartGang (activating a gang)
- StopGang (interrupts a gang currently active)
- StopAllGangs (interrupts all gangs currently in action)

Finally there is a real variable brkpnt (breakpoint). If set to a positive number (e.g. in the procedure DetermineAction), the manager reactivates itself after the time indicated by the variable value. If it however is interrupted during the time interval, the breakpoint is cancelled.

The dynamic behaviour of the manager is rather trivial: it waits until it is interrupted or until the time reaches a breakpoint. Then it executes the (redefined) procedure DetermineAction. Finally it resets the interrupt "flag" and starts over again, in an endless loop. See figure 9.

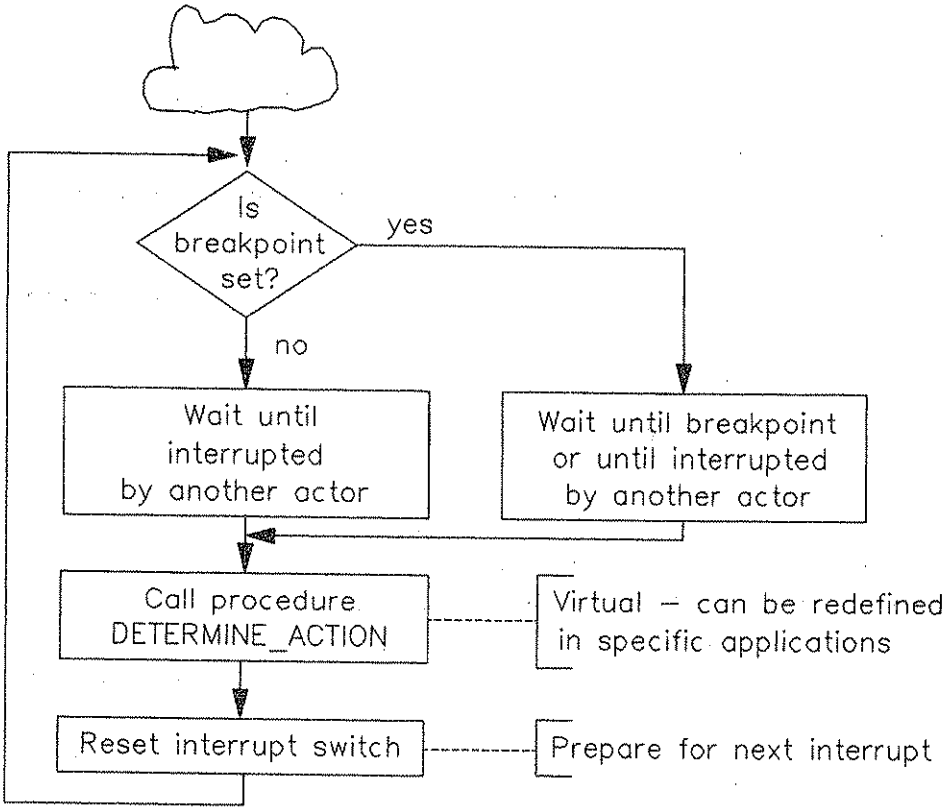

Figure 9. Activity diagram of the manager object.

3.2.5 The calendar

The calendar object has two major tasks: to keep track of the starting days of the work periods (there may be many periods within one simulation run) and to start the daily

work each morning within a work period. It may also stop the work in the evening if it is not already over. This facility is however just intended for preventing work to go on all night. Normally the manager should have stopped the work before that time.

The number of work periods and their starting days are read from indata as well as the points of time for the start and end of the working hours. The end of the work period, on the other hand, can not be determined in advance. Instead, the manager's Boolean member procedure PeriodOver is called every morning to check this.

The dynamic behaviour of the calendar is implemented by means of an outer (periodical) and an inner (daily) loop. At the beginning of each work period, the environmental actors (see below) are activated. At the beginning of each day the manager is interrupted to start the new day's work. At the very end of the day, all gangs still active are interrupted to teardown without engaging the manager. In order to enable other objects to inspect whether the time of day is between these points of time, its flag "night" is reset (i.e. FALSE) during this interval, and set (TRUE) otherwise. Similarly, the flag "workPeriod" is set during the work periods, and reset otherwise.

When the calendar finds that the last work period is over, the simulation is stopped.

Within the calendar object, there are some utility procedures, facilitating calculation and printing of current simulated day numbers and clock times. Day numbers may also be converted to month and day, and vice versa.

The calendar object is complete as it is, no complementing or redefinition in sub-classes is normally needed.

3.2.6 The environmental actors: RainReporter and Sun

Besides the continuously changing temperature, humidity etc. there are also some weather factors which can be considered as discrete, and which may interfere in the process of field operations, either by making it impossible to perform the operation (when it starts raining) or by causing a decision date due to changed work conditions (when the sun goes up or down, and when it stops raining). Typically, these events also affects the biological systems in different ways.

For this reason, two actors keeping track of, and reporting on rain start-stop and sunrise-sunset events were introduced into the model. (The reason for making two actors was that it is simpler to make two actors with simple tasks than to make one actor with a complicated task.)

The sun object has a member procedure returning the sun height for a given time of the year, latitude, longitude and time zone. From this one the equation for the time for sunrise and sunset is derived, see McGechan & Glasbey (1988).

The dynamic behaviour of the sun object is to calculate the points of time for the next sunrise and sunset, and to activate itself at these points of time. It updates the states of all fields, toggles its flag "sunUp" and (in the evening only) interrupts the manager with a message that the sun is now down. When the sun goes up the manager is not interrupted since field work almost never starts already at sunrise.

The rainreporter reads the points of time for the next rain start or stop from an indata file. (In a real time system, this could be an indata port.) Similarly to the sun, it activates itself at the point of time of the next rain start or rain stop, whichever comes first, updates the states of all fields, and toggles its flag "raining". If it is a rain start event, the rainreporter immediately interrupts all active gangs to make them tear themselves down. Otherwise it interrupts the manager, since it might be possible to start working again. It is up to the manager to check if it is at night when it is interrupted.

Both these objects are designed to be active only during the work periods. Whenever the flag "workPeriod" of the calendar is reset, they passivate themselves. When a new period starts, the calendar wakes them up.

These objects are complete as they are, normally needing no complementing definition in sub-classes.

3.3 Development of an application

3.3.1 Creating the application model

At the application level, the behaviour of certain actors have to be further specified or even changed. This is made possible in two ways.

The first one is that the parts of the behaviour of the actor subject to change are generally defined in procedures declared "virtual". A virtual procedure may be redefined in a sub-class. For example, the procedure UpdateState of the class FIELD is called in the appropriate place, but it is empty. In a sub-class HAYFIELD the procedure is redefined. Then the redefined version is called instead of the original one.

The other way is by appending extra initialization code. In this case, the original initialization code is not replaced, but complemented.

Both these ways of refining an actor's behaviour are quite uncomplicated, as will be shown later.

The output of the discrete event oriented results is conveniently produced by DEMOS' own report generator. For other output data, one will however have to write one's own report routines.

3.3.2 Environment of the base model

FIELD_OP is a class itself, embedding all the classes described above. In a practical problem, where at least the FIELD and MANAGER classes are redefined using the sub-class concept, this new definition will be embedded in a sub-class to FIELD_OP. See figure 10.

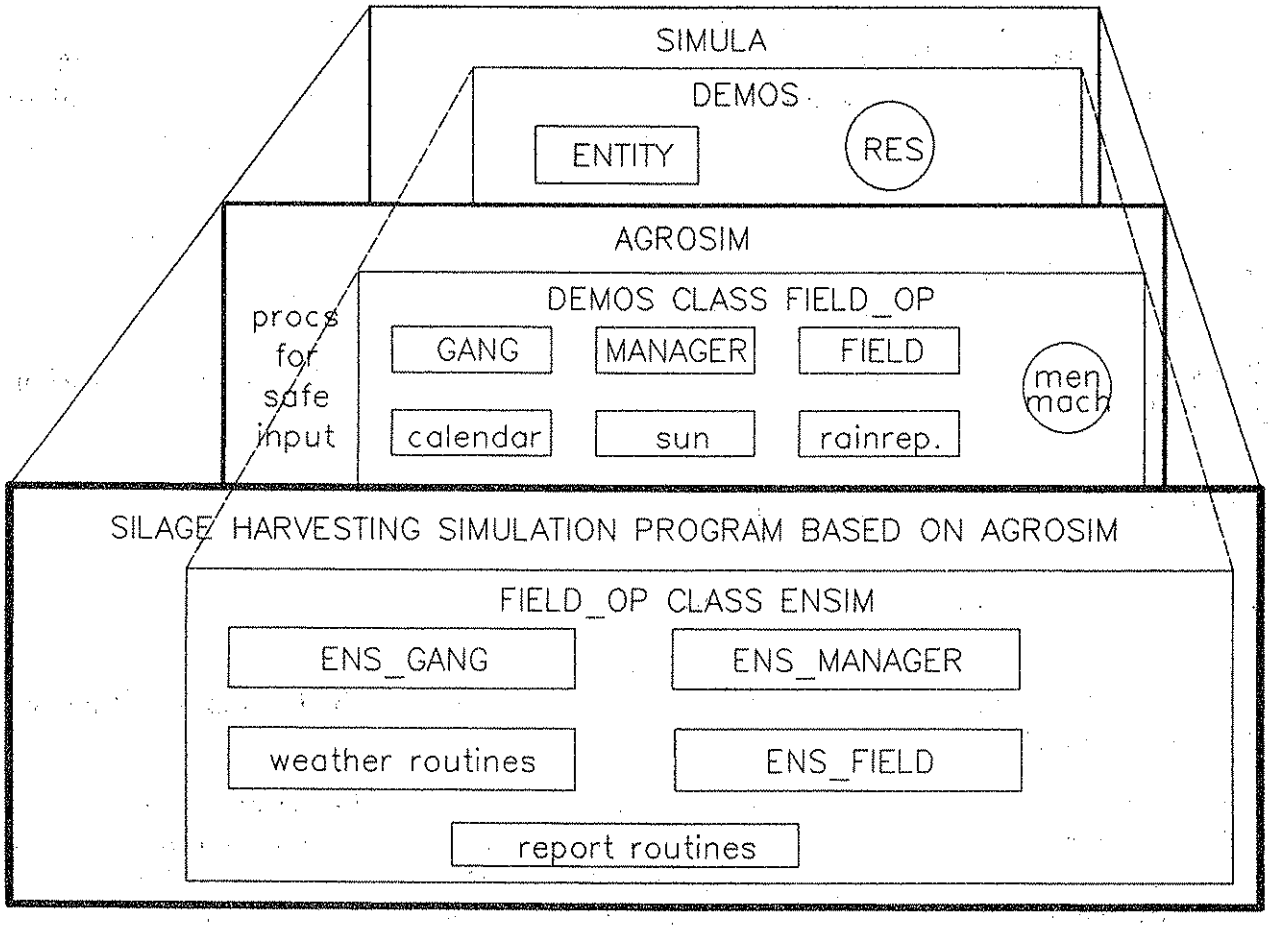

Figure 10. The environment of the base model. The figure shows the entire programming language of Simula enriched with DEMOS as the bottom layer, the class AGROSIM as an enrichment of Simula, containing FIELD_OP which is an enrichment of DEMOS, as an intermediate layer, and an application program based on AGROSIM and FIELD_OP.

Furthermore, the class DEMOS is surrounded by an "environment", a class called AGROSIM also containing a number of procedures to help to control the correctness of indata.

3.4 Initialization

3.4.1 Initialization data format

The general principle of initialization is that the design of the machinery system is to be specified at runtime as far as possible. This is accomplished by entering the dimensions of certain data structures as well as the values of certain attributes into the system as input data.

The general indata format is fixed in the sense that all indata must come in a predetermined order. It might however be freely divided into several lines. Empty lines may also be used freely. Comments are however not allowed in the indata file. The different indata sections should be entered in the following order:

1) general dimensions of the problem
2) attributes of the calendar object
3) attributes of the sun object
4) definition of categories of men and machinery
5) definition of field states
6) the fields and their attributes
7) the gangs and their attributes
8) miscellaneous

The "general dimensions" indata consists of the number of categories of men and machinery, the number of gangs and the number of field states. These are required first in the program to make up the dimensions of certain arrays. This list of indata may be extended by appending other variables declared in sub-classes to FIELD_OP.

The initialization of the objects embedded by FIELD_OP fall into two categories. The first one consists of the classes not needing any refinement in sub-classes, namely CALENDAR, SUN and RAINREPORTER. Also men and machinery fall into this category. The initialization of the objects of these classes is built-in into the object itself or into FIELD_OP, making it transparent to the application program. Thus the only problem in initializing these objects is to enter the indata properly.

The other category includes all the other classes, GANG, FIELD and MANAGER. The two latter ones must be refined to be useful. For one thing, they must be complemented with the attributes relevant for the specific problem. Thus the input of these indata must be appended to the obligatory ones. This problem is solved by means of placing all data input inside virtual procedures. If a sub-class to class FIELD is defined, a new indata procedure must also be defined, with the same contents as the original one, but extended with the extra lines necessary.

In the following paragraphs the exact formats of the indata for the different objects are described. The sun object and the rudimentary manager object need no indata for their initializations.

The indata items must be entered in the exact order specified below. Text strings must not exceed 10 characters or be split into several lines.

3.4.2 Initializing the calendar

The attributes required for the calendar object are as follows:

- the year,
- work start time (decimal - 7.5 means 07:30),
- definitive work stop time (decimal - 23.75 means 23:45),
- number of work periods within the simulation (normally 1),
- required name of the object in the simulation report.

For each work period is required:

- starting month,
- starting day of the month.

3.4.3 Initializing the sun

This object needs only three attributes:

- latitude (decimal),
- longitude (decimal),
- hour relative to Greenwich Mean Time (GMT).

In Sweden, the normal value of hour relative to GMT is 1. In summer it is however nowadays 2 due to the daylight saving time.

3.4.4 Initializing the men and machinery

The number of categories of men and machinery is already determined (see 5.3.1). For each category three attributes are required:

- category number (1 for the first category etc.),
- number of items available of this category,
- required name of the category in the simulation report.

Note that no distinction is made between men and machinery. They are simply regarded as different resources. If distinctions between different men, tractors etc are required, they should simply be divided into several categories. Then different gangs, with different capacities, are defined using these different categories.

3.4.5 Initializing the fields

Firstly, the necessary MATERIALs must be defined. Each MATERIAL is simply given an integer number starting with 1. For example, in a hay-making system, the MATERIALs could be GROWING (1), MOWED (2), TEDDED (3), WINDROWED (4) and maybe BALED (5). For each MATERIAL two items are required:

- MATERIAL code (starting with 1),
- required name of the MATERIAL in the simulation report.

This name is given to the field queue devoted to the fields currently in that particular MATERIAL. The field queues are initialized automatically.

Now the fields themselves can be created. To begin with, the initial number of fields is entered. Then, for each field, the following data are required:

- field name (used in the simulation report),
- acreage (normally hectares, but other units are also feasible),
- initial value of the state of the field.

The attributes of the field objects are read by means of a procedure NewField, which is called once for each field. It is a member procedure of class FIELD_OP. In any sub-class of FIELD_OP, when appending attributes to the field objects, this procedure must also be redefined to be able to read the additional data required. Normally, these data are simply appended to the obligatory data for each field in the indata file.

3.4.6 Initializing the gangs

Quite a large number of attributes are needed for each gang:

- gang code (starting with 1),
- the number of men and machinery categories utilized in the gang,
- gang name (used in the simulation report).

For each number of men and machinery, two figures are required:

- category code (starting with 1),
- initial number of items of the particular category.

Finally, some attributes are needed to describe the process performed by the gang:

- mtrl_processed (integer code of initial MATERIAL),
- mtrl_delivered (integer code of final MATERIAL),
- process name (used in the simulation report),
- setup time (in hours, decimal),
- teardown time (in hours, decimal),
- normal work time (hours per acreage unit).

If yet some attributes are to be appended to the gangs, for example concerning losses, the member procedure of FIELD_OP, NewGang, must be redefined just as is the case with the fields.

3.4.7 Output

In the "mother" class DEMOS, a report generator is included, optionally producing an event log and summary statistics. The report generator is reset for each period so that if the program simulates for example two hay harvests in a year, two reports may be produced.

In class FIELD_OP, no output is appended except for error messages and other important diagnostic messages. It is however possible to redirect both of these into dummy output files if they are not required. If some extra output is required, it is most convenient to control it from the redefined procedure DetermineAction of the manager object.

3.4.8 Miscellaneous

Somebody might want to add some additional objects not supported in FIELD_OP. One object possibly subject to inclusion would be a weather object, taking care of all weather data (historical, simulated or forecasted) typically required in this type of models. Since the availability of weather data may vary substantially, it was decided not to include it in the base model. It is probably just as simple to develop this part from scratch.

4 VALIDATION

4.1 Objective

The objective of validating the base model was to determine the correctness and usefulness of the base model. In particular, since the base model was originally developed from a model for simulating field operations in hay harvesting, it was considered important to prove that the model is applicable for a wider range of problems.

Among several different field operations considered, it was decided to perform the validation by means of constructing a testbed program, simulating the harvesting of wilted silage. The reason for this was that this application seemed to offer the best possibilities of testing the different parts of FIELD_OP, as well as giving a good possibility to show the potential of extending the base model in different respects.

4.2 A constructed problem

This testbed program simulates harvesting of wilted silage.

The machinery system consists of a trailed mower-conditioner, a trailed precision chopper, three trailers, three tractors and three men. (The man in the silo is omitted from the simulation.) The chopping operation may be performed by either 2 or 3 men. In the former case, mowing and chopping can be performed simultaneously, the 2-men chopping operation having a lower capacity than the 3-men chopping operation. The trailers are coupled to the chopper during chopping rather than to a tractor driving beside the chopper.

The problems to be solved by the testbed program, could be to investigate how

- the variation in moisture content,
- the length of the harvesting period,
- the man-hours and machine-hours used,

are affected by

- the mowing and chopping capacity,
- the management strategy used,

under different weather conditions.

Problems not to be covered by the program were, for example, the effects of machinery breakdowns or accidents; neither are the estimation of losses caused by rain.

4.3 Development of the testbed program

4.3.1 Overall structure

As have already been stated, developing an application program from the FIELD_OP base model consists essentially in developing, refining, the concepts already defined in FIELD_OP. This is done by means of defining sub-classes to some of the generic classes defined in FIELD_OP.

In the testbed program all the classes FIELD, GANG and MANAGER had to be refined. Furthermore, a class WEATHER, taking care of the weather records, had to be created.

In the same manner as the classes of the base model being encapsulated by a surrounding class FIELD_OP, the sub-classes are encapsulated by a sub-class to FIELD_OP, named ENSIM.

The class ENSIM has no additional arguments added, so its argument list is identical to the one of class FIELD_OP.

4.3.2 Refining class FIELD

The possible MATERIALs are GROWING, MOWED and HARVESTED.

A subclass of class FIELD called ENSFIELD was defined to enable additional attributes to be appended.

A continuous variable MCDB (moisture content dry base) was appended. For its updating, a very simple drying model was developed. This was made local to the redefined procedure UpdateState.

In order to enable a more precise calculation of the capacity of the gangs, the continuous variables trptDistance, yield and fieldWidth were appended. (The algorithm is described in the next paragraph.)

The drying model was inspired by earlier work by the author (Axenbom, 1983). It should be noted that this drying model was only used for the validation of the base model, the moisture content figures were not used.

4.3.3 Refining class GANG

Class GANG could have stayed as it was, but the simplistic procedure Capacity was considered yielding a too low resolution for the problem in question. Therefore the original Capacity procedure was replaced by a more detailed one, which takes into account the fact that the chop-gangs actually consist of one chopping part and one transporting part.

The field capacity is calculated as the minimum capacity of these ones. For the transporting part, it also takes into account the transport distance, transport speed, load size and yield.

Some of these parameters are field attributes and have already been mentioned. The other ones are gang attributes, namely

- loadWeight (ton),
- trptSpeed (km/h),
- unloadTime (hrs),
- changeTrailertime (hrs).

It should be noted that the parameter normWorkTime is still used, but is redefined as hours per ton instead of hours/ha.

4.3.4 Refining class MANAGER

Contrary to the classes FIELD and GANGDATA, a sub-class to class MANAGER always must be defined, since the initially empty procedures DetermineAction and PeriodOver have to be redefined.

The criterion used here for a period being over is that all fields have reached their final state (harvested).

The decision strategy used in the testbed program is kept as simple as possible. Basically, there are three rules:

- if it is daytime, and it is not raining, then it is OK to start working. (Daytime is defined as "the sun is up and timeofday < 21:00".)

- if mowing machinery is available and the maximum acreage mowed is not reached, mowing always has precedence to chopping. However, chopping is never interrupted in order to release resources for mowing.

- the high capacity chopping gang has precedence to the low capacity one.

"Precedence" here means that the operation with the highest precedence may allocate the resources and fields it needs before the other ones.

4.3.5 Creating class WEATHER

Since the testbed program contains a simple field drying model, some weather data is required. Even though the drying model requires only the potential evaporation, the interface to the weather data file was designed to make the actual data file format transparent to the rest of the program.

Transparency is achieved by directing all reading of weather data through member procedures of a class WEATHER. Two different procedures are provided: the real procedure DailyEvap, which returns the daily potential evaporation value, and the real procedure EvapDuring, which approximates the distribution of the potential evaporation over the day, and integrates it over the required time interval.

5 RESULT OF THE VALIDATION

The results of a simulation in DEMOS are of two different kinds. The one is an event log, tracing all events that have taken place. The other one is a report showing the usage of the different resources and queues.

5.1 The simulation log

The simulation log produced by DEMOS gives a very detailed "trace" of everything happening inside the program. From this log, useful output data such as capacity etc, may be derived. It is however a bit lengthy.

For convenience purposes, a second logfile is produced, containing only the events of interest to the user. In the testbed program, also the moisture content of the drying crop is printed in this file. The printout of the latter logfile is found in appendix 1, while the former one is in appendix 2.

5.2 The report

The report provided by DEMOS gives statistics on the usage of resources and queues. For the resources (i.e. the men and machinery) the following data are given:

- (RE)SET, the simulated point of time when the resources was created or reset
- OBS, the number of times the resource has been used
- LIM, the initial number of free units of the resource
- MIN, the minimum number of free units during the simulation period
- NOW, the current numbber of free units
- % USAGE, the proportion of the resource that has been used since (RE)SET

AV.WAIT and QMAX are irrelevant in the FIELD_OP context. An example printout (fetched from the silage-making validation run) is showed in table 1.

Table 1. DEMOS report on the usage of the RESources in the simulation.

TITLE	/ (RE)SET/	OBS/	LIM/	MIN/	NOW/	% USAGE/	AV. WAIT/	QMAX
workers	3607.500	21	3	0	3	31.129	0.000	1
tractors	3607.500	21	3	0	3	31.129	0.000	1
mower	3607.500	9	1	0	1	16.388	0.000	1
chopper	3607.500	12	1	0	1	30.296	0.000	1
wagons	3607.500	12	3	0	3	25.666	0.000	1

CLOCK TIME = 3679.500

The use in hours for any resource is easily derived by multiplying the usage (decimal) by the number of hours. The latter equals the current simulated clock time (printed at the header of the report) minus the (RE)SET time. (The unit of the simulation time is hours since January 1 at 00:00.)

The report of the queues is quite similar. It consists of the following data:

- (RE)SET, the simulated point of time when the queue was created or reset
- OBS, the number of times an entity (a field) has left the queue since (RE)SET
- QMAX, the maximum queue length during the simulation since (RE)SET
- QNOW, the current length of the queue
- QAVERAGE, the average queue length
- ZEROS, the number of leaving the queue immediately after entering it
- AV.WAIT, the average time spent in the queue (including ZEROS)

The printout from the silage-making run is shown in table 2.

Table 2. DEMOS report on queue usage.

TITLE /	(RE)SET/	OBS/	QMAX/	QNOW/	Q AVERAGE/	ZEROS/	AV. WAIT
growing	3607.500	9	7	0	1.066	0	2814.360
mowed	3607.500	12	2	0	1.140	2	6.839
harvested	3607.500	0	11	11	7.327	0	----------

CLOCK TIME = 3679.500

6 DISCUSSION

6.1 What do the validation results prove?

6.1.1 Correctness of the model

The first thing to discuss is whether the model executes correctly. For this purpose, the log of the first day's simulation will be discussed in detail.

As shown in appendix 1, first (07:30) field 1 is mowed. Thereafter (09:18) the mow-gang continues with field 2, while field 1 is being chopped by the 2-men chop-gang. At 09:53 the mow-gang continues mowing field 3, and 11:27 with field 4. After completing chopping field 1 at 12:09, chopping continues with field 2.

When field 2 is chopped at 13:20, mowing is halted, since the mowed acreage exceeds the maximum one, 7.5 ha. The 2-men chop-gang is halted in favour of the 3-men one, which is faster. The stopping of the mow-gang results in the field processed being split into two parts, one mowed (FIELD 4(1), 0.8 ha) and one not mowed (FIELD 4(2), 3.2 ha).

Now the 3-men chop-gang continues chopping field 3. When this is finished at 16:20, the mowed acreage is lower than the minimum one, 5.0 ha. At this time, the mow-gang is again activated, starting mowing field 4(2). The rest of the day, the two gangs work in parallel.

At 21:35 the sun sets, and both the active gangs are stopped. Both fields being processed are split up.

The example shows that the model behaves as intended. When a gang is activated, it acquires a field having the appropriate state, then it start workings after the set-up time and finally it releases the now processed field. An analysis of the more extensive DEMOS logfile in appendix 1 further proves that the interactions between gangs, fields and manager are correct.

6.1.2 The usefulness of the base model program

The second thing to discuss is whether the concept is useful in general, or rather, which information the testbed program gives us about the general usefulness of the program. This question will be covered from the following aspects:

- resolution (is the model detailed enough?)
- applicability (for which field operations is it useful?)
- efficiency (how efficiently are the problems solved?)
- design (is the design of the base model appropriate?)

The approach to answer these questions will go through a discussion about what the base model does not provide the possibility to model.

The resolution of the program is more or less determined by the resolution of the model of the gang, performing the field work. The concept chosen here is to calculate the capacity statistically, rather than modelling the actual work on the field and the transports between the farm and the field.

It has been shown that the resolution of the original capacity calculation is very simplistic, but that it can easily be improved, using a more sophisticated regression model. However, the interactions between for example the chopper and the transport function in a silage system, typically resulting in the one waiting for each other, can not be modelled.

These are the most important simplifications built into the gang:

- the teardown time is appended to the set-up time,
- the setup time is independent of transport distance,
- the setup time is constant even if the gang has been recently active,
- two gangs cannot work simultaneously at the same field.

The resolution of the fields is high enough as far as the discrete event system is concerned, since they do not take complex actions of their own. The resolution of the biological models is not limited by FIELD_OP, since these are defined on the application level.

The manager, the sun, the rainreporter and the calendar conforms the decision system of the program. The manager is very flexible, since the explicit strategy is supposed to be programmed rather than entered as indata. The drawback of this is the increased complexity in implementing the strategy. The main reason of choosing this solution was the aim to keep the base model compact and flexible.

There are some presumptions built into the decision system:

- the calendar starts work at the same time every day,
- no distinctions are made between different week days,
- leap years are not considered,
- the rainreporter always stops all current activity when rain starts.

6.1.3 Concluding remarks

The conclusion of the author about the resolution is that the simplifications and presumptions built into the model are not restricting its usefulness for most problems. However, for cyclic transport systems, as for example silage harvesting, it is clear that the model does not offer the possibility of making a detailed model of the coordination between the different parts of it. Consequently, the applicability is restrained with respect to such problems.

The efficiency of the program with respect to the problems to be solved (see 4.2) must be judged from the amount and quality of the output from the program. From the sample output earlier in this chapter, it is evident that the output data required can be produced. Let be that a whole series of simulations, which would have been required to solve the problem properly, has not been performed.

The layout of the output could easily be improved on, to the cost of extra program code. Since this would reduce the maximum problem size, it has not been done.

In terms of computer resources, the program is both efficient and inefficient. Execution speed is never any problem in conjunction with discrete event simulation. Simula is however extremely memory-consuming, which may restrict the maximum problem size. This problem has been avoided by designing the program as compactly as possible, making it quite efficient from Simula's point of view.

The quality of the design can be evaluated from the above. The author's impression is that the design of the base model is quite powerful. Much of its strength comes from retaining the elegance and simplicity of DEMOS, while extending it with tools making it useful for a multitude of problems.

The conclusions are unfortunately based on a very limited experience with the model. In particular, it would be valuable with experiences from other persons than the author himself.

6.2 Future use and development

6.2.1 Planned use

The original application for the base model was of course the simulation model of hay harvesting, since this is the one for which it was originally developed.

In the near future, it is planned to use the base model in a model of combine harvesting and fuel straw harvest, in order to investigate the impact of weather etc on the amounts of fuel straw available and how it may be altered by means of increasing the harvesting capacity and/or altering the harvesting strategy.

6.2.2 Need for development

For the presently planned applications, it is believed that the current version of the base model is appropriate. However, several suggestions of future development have already been considered. These goes in two directions.

In the scientific direction, there may be a need for detailed simulation of cyclic transport systems, and possibly also for simulating the driving pattern on the field. These

demands are probably not possible to fulfil within the FIELD_OP base model. A separately developed model could however benefit from the experiences from this work, as well as from the works of Elinder (1984) and Kindler et al (1983).

The other direction concerns the user interface. Here the options are unlimited. Listed below are however only suggestions in order to make an application program more functional to use.

A data pre-processor with a database would considerably simplify the task of producing a correct input data file. This should therefore have a high priority.

A pre-defined strategy enabling the decision rules or control variables to be entered as indata would simplify the programming task, but also restrict the flexibility of the program. For now, it seems wise to leave the strategy to the application program.

Interactive simulation with full-screen interface - this feature has already been used in the hay simulation program. A general approach would however preferably include a graphical interface using a pointer device. This is probably not possible under the memory limits of MS-DOS. It is however the author's belief that strategic games on computer, using interactive simulation, will become important training tools in the future, also in agriculture. So there is doubtlessly a need for this type of tool.

Animated simulation is a natural extension to the point above. This is realistic primarily using computer systems having a graphical interface, notably work stations.

Graphical output would be most valuable, and should be given high priority. For example, a Gantt Chart showing the activity of the different actors would simplify the interpretation of the results. Unfortunately, also this may be too memory-consuming to be possible to implement under MS-DOS.

7 CONCLUSIONS

The general base model for simulation of field operations has been shown to execute correctly, and to produce reliable results. The testbed program described has demonstrated most of the facilities of the base model for a deterministic application program.

The example with the testbed program shows that it is relatively simple to develop application programs, using the facilities of FIELD_OP and DEMOS. It is made believable that the concept of the base model is useful for a relatively large category of applications. However, it is probably not well suited for detailed simulation of cyclic transport systems and of the driving pattern during field work. If such high resolution is needed, it might be beneficial to develop a separate program, preferably built upon DEMOS.

DEMOS has proved its excellence for discrete event simulations. Its combination of simplicity and power is outstanding.

The Simula language, which is a prerequisite to use DEMOS, is close to ideal as a research tool for program development, since it is truly high-level, object-oriented, standardized and portable. For the development of commercial products it falls short compared to, for example, the languages C and Modula-2 in terms of compilation speed, execution speed and compactness of object code. As computer capacity becomes cheaper with time, the disadvantages mentioned might become less important, whereas Simula's advantages will grow in importance as these are efficient means of reducing program development costs.

The PC-Simula compiler may look ancient at a first glance, but it does offer advanced features such as for example a symbolic debugger. It is in fact basically the same compiler as the ones for VAX, PRIME and several other minicomputers. This guarantees its portability now and in the future.

Compared to van Elderen's (1987) model SFO_BASE, FIELD_OP has its advantages in its simplicity and compactness, making it available under MS-DOS and quite easy to learn. Some of the strengths of van Elderen's model are its conceptual excellence and the integrated decision model built upon the urgency concept. It is the author's belief that using the FIELD_OP model in some cases will be a step towards the more powerful tool SFO_BASE. The similarities in concepts and terminology between these two models should simplify the evolution of an application from the FIELD_OP base into the SFO_BASE one.

Discrete event simulation has been shown in this report to be an excellent operations research technique to evaluate man-machine systems in agriculture. However, it is only sparsely used for this purpose. The main reason for this is probably the initial difficulties often encountered in developing pseudo-parallel programs. The base model described in this report should minimize the efforts needed to develop discrete event models of field operations in agriculture, making it useful in education as well as for research purposes. It is hoped that it will be used for both of these.

REFERENCES

Attonaty, J.-M. 1988. Personal communication.

Axenbom, Å. 1983. Metoder att beräkna effekter av olika sätt för stråbehandling vid förtorkning av hö. (Methods to calculate the impact of different conditioning treatments in field drying of hay.) Report 87. Dept of Agricultural Engineering. Swedish University of Agricultural Sciences. Uppsala. 39 pp.

_____. 1988. An integrated simulation model of hay growth, harvesting and barn drying. Report 120. Dept of Agricultural Engineering. Swedish University of Agricultural Sciences. Uppsala.

Birtwistle, G.M. 1979. Discrete Event Modelling on Simula. 147 pp. Macmillan Press Ltd. London, UK.

Chen, L.H., Sowell, R.S. & Humphries, E.G. 1976. A simulation model for multiple harvesting of pickling cucumbers. J. agric. Engng Res. **21**, 67-75.

Cunney, M.B. & Von Bargen, K. 1974. Comparing hay harvest machinery by computer simulation. ASAE paper no. 74-1545. St Joseph, Michigan.

van Elderen, E. 1977. Heuristic strategy for scheduling farm operations. Pudoc, Wageningen, Netherlands.

_____. 1984. Personal communication.

_____. 1987. Scheduling farm operations: a simulation model. 226 pp. Pudoc, Wageningen, Netherlands.

Elinder, M. 1984. Arbetsbehov för växtodling - en system- och datamodell. Institutionsmeddelande 84:02. Dept of Agricultural Engineering. Swedish University of Agricultural Sciences. 84 pp. (In Swedish.)

Franta, W.R. 1977. The process view of simulation. North-Holland. 241 pp.

Garg, H.P. 1982. Treatise on solar energy. Vol 1. Fundamentals of solar energy. Wiley, Chichester.

Gustafsson, L. 1981. Split-simulation - an efficient method for studies of models with discrete choices. Mathematics and Computers in Simulation **23**, 245-252.

Herath, H.M.G. 1982. Decision making models with special reference to applications in agriculture: review and a critique. Oxford Agrarian Studies **11**, 139-157.

Jonsen, B.M. 1983. Personal communication.

_____. Simuleringsmodell for feltarbeid. Driftsteknisk informasjonsbank. Landbruksteknisk Institutt, Ås, Norway. (In Norwegian.)

Kindler, E., Chocol, S. & Prokop, C. 1983. Systems of material flow. Int. J. General Systems 9, 217-224.

Kindler, E. 1984. Personal communication.

Konaka, T. 1974. Simulation of forage crop harvesting system. The bulletin of the Faculcy of Agriculture no. 47, 365-413. Mie University, Tsu, Japan.

Lovering, J. & McIsaac, J.A. 1981. A timothy-beef model for Atlantic Canada. Agricultural Systems 7 (3), 219-233.

McGechan, M.B. 1986. OPerational Research study of forage conservation systems for cool, humid upland climates. Paper presented at Ag Eng-86, The Netherlands.

McGechan, M.B. & Glasbey, C.A. 1988. Estimates of solar radiation based on other meteorological parameters for use in simulations of swath drying. Dep. Note 5, Scott. Centre agric. Engng. Penicuic. 20 pp.

Oving, R.K. 1971. Farming task and machinery. Research report 3. Institute of Agricultural Engineering and Rationalization. Wageningen, Netherlands. 17 pp.

Parke, D., Dumont, A.G. & Boyce, D.S. 1978. A mathematical model to study forage conservation methods. Journal of the British Grassland Society 33, 261-273.

Pitt, R.E. 1982. A probability model for forage harvesting systems. Trans. ASAE, 549-555, 562.

Russell, N.P. et al. 1982. Alternative choice criteria in farm management models - a comparative study of forage machinery choice. Manuscript to be submitted in Am J. agric. Econ.

Savén, B. 1988. Produktions simulering. Mekanförbundets förlag. 240 pp. (In Swedish.)

Savoie, P., Parsch, L.D., Brook, R.C., Rotz, C.A. & Thomas, J.W. 1982. Simulation of alfalfa harvest alternatives. ASAE paper no. 82-1034. St Joseph, Michigan. 26 pp.

Simula A.S. 1988. PC-Simula under MS-DOS. Programmer's reference manual. Oslo, Norway. 350 pp.

Verma, L.R., Von Bargen, K. & Ballard, J.L. 1976. Planning forage harvesting research using simulation. Trans. ASAE, 1022-1024.

Whan, B.M., Scott, C.H. & Jefferson, T.R. 1976. Scheduling sugar cane plant and ratoon crops and a fallow - a constrained Markov model. J. agric. Engng Res. 21, 281-289.

APPENDIX 1: Compact simulation log

```
1990-06-28 21:31:57.330  Logfile of ENSIM session

81-05-31 07:30  Mow-gang    1  setting up   with mowing     of Field1     1
81-05-31 07:41  Mow-gang    1  working      with mowing     of Field1     1
81-05-31 09:18  Mow-gang    1  finished     with mowing     of Field1     1
mowed areal, ha: 4.0&+00
81-05-31 09:18  Mow-gang    1  setting up   with mowing     of Field2     1
81-05-31 09:18  choptrpt1   1  setting up   with choptrpt1  of Field1     1
81-05-31 09:30  Mow-gang    1  working      with mowing     of Field2     1
81-05-31 09:36  choptrpt1   1  working      with choptrpt1  of Field1     1
81-05-31 09:53  Mow-gang    1  finished     with mowing     of Field2     1
mowed areal, ha: 4.5&+00
81-05-31 09:53  Mow-gang    1  setting up   with mowing     of Field3     1
81-05-31 10:05  Mow-gang    1  working      with mowing     of Field3     1
81-05-31 11:27  Mow-gang    1  finished     with mowing     of Field3     1
mowed areal, ha: 5.5&+00
81-05-31 11:27  Mow-gang    1  setting up   with mowing     of Field4     1
81-05-31 11:39  Mow-gang    1  working      with mowing     of Field4     1
81-05-31 12:09  choptrpt1   1  finished     with choptrpt1  of Field1     1
mowed areal, ha: 5.7&+00
81-05-31 12:09  choptrpt1   1  setting up   with choptrpt1  of Field2     1
81-05-31 12:27  choptrpt1   1  working      with choptrpt1  of Field2     1
81-05-31 13:20  choptrpt1   1  finished     with choptrpt1  of Field2     1
mowed areal, ha: 7.6&+00
Areal processed of Field4     1 is      4.2 ha.
Remaining areal=     0.8 ha conforms Field4     2
81-05-31 13:20  Mow-gang    1  finished     with mowing     of Field4     1
81-05-31 13:20  choptrpt2   1  setting up   with choptrpt2  of Field3     1
81-05-31 13:38  choptrpt2   1  working      with choptrpt2  of Field3     1
81-05-31 16:20  choptrpt2   1  finished     with choptrpt2  of Field3     1
mowed areal, ha: 4.2&+00
81-05-31 16:20  Mow-gang    1  setting up   with mowing     of Field4     2
81-05-31 16:20  choptrpt1   1  setting up   with choptrpt1  of Field4     1
81-05-31 16:32  Mow-gang    1  working      with mowing     of Field4     2
81-05-31 16:38  choptrpt1   1  working      with choptrpt1  of Field4     1
81-05-31 16:50  Mow-gang    1  finished     with mowing     of Field4     2
mowed areal, ha: 4.6&+00
81-05-31 16:50  Mow-gang    1  setting up   with mowing     of Field5     1
81-05-31 17:02  Mow-gang    1  working      with mowing     of Field5     1
81-05-31 17:50  Mow-gang    1  finished     with mowing     of Field5     1
mowed areal, ha: 4.5&+00
81-05-31 17:50  Mow-gang    1  setting up   with mowing     of Field6     1
81-05-31 18:02  Mow-gang    1  working      with mowing     of Field6     1
81-05-31 18:40  choptrpt1   1  finished     with choptrpt1  of Field4     1
mowed areal, ha: 4.4&+00
81-05-31 18:40  choptrpt1   1  setting up   with choptrpt1  of Field4     2
81-05-31 18:58  choptrpt1   1  working      with choptrpt1  of Field4     2
81-05-31 19:14  Mow-gang    1  finished     with mowing     of Field6     1
  mowed areal, ha: 5.3&+00
81-05-31 19:14  Mow-gang    1  setting up   with mowing     of Field7     1
81-05-31 19:24  choptrpt1   1  finished     with choptrpt1  of Field4     2
mowed areal, ha: 5.0&+00
81-05-31 19:24  choptrpt1   1  setting up   with choptrpt1  of Field5     1
```

```
81-05-31 19:26  Mow-gang   1  working      with mowing     of Field7     1
81-05-31 19:42  choptrpt1  1  working      with choptrpt1  of Field5     1
81-05-31 20:53  choptrpt1  1  finished     with choptrpt1  of Field5     1
mowed areal, ha: 6.6&+00
81-05-31 20:53  choptrpt1  1  setting up   with choptrpt1  of Field6     1
81-05-31 21:11  choptrpt1  1  working      with choptrpt1  of Field6     1
Areal processed of Field7     1 is      4.8 ha.
Remaining areal=     1.9 ha conforms Field7     2
81-05-31 21:20  Mow-gang   1  finished     with mowing     of Field7     1
Areal processed of Field6     1 is      0.2 ha.
Remaining areal=     2.8 ha conforms Field6     2
81-05-31 21:20  choptrpt1  1  finished     with choptrpt1  of Field6     1
mowed areal, ha: 7.6&+00
81-06-01 21:21  choptrpt2  1  setting up   with choptrpt2  of Field6     2
81-06-01 21:39  choptrpt2  1  working      with choptrpt2  of Field6     2
Areal processed of Field6     2 is      1.4 ha.
Remaining areal=     1.4 ha conforms Field6     3
81-06-01 22:00  choptrpt2  1  finished     with choptrpt2  of Field6     2
mowed areal, ha: 6.2&+00
81-06-02 07:30  Mow-gang   1  setting up   with mowing     of Field7     2
81-06-02 07:30  choptrpt1  1  setting up   with choptrpt1  of Field6     3
81-06-02 07:41  Mow-gang   1  working      with mowing     of Field7     2
81-06-02 07:48  choptrpt1  1  working      with choptrpt1  of Field6     3
81-06-02 08:16  choptrpt1  1  finished     with choptrpt1  of Field6     3
mowed areal, ha: 6.2&+00
81-06-02 08:16  choptrpt1  1  setting up   with choptrpt1  of Field7     1
81-06-02 08:26  Mow-gang   1  finished     with mowing     of Field7     2
mowed areal, ha: 6.6&+00
81-06-02 08:26  choptrpt1  1  finished     with choptrpt1  of Field7     1
81-06-02 08:26  choptrpt2  1  setting up   with choptrpt2  of Field7     2
81-06-02 08:44  choptrpt2  1  working      with choptrpt2  of Field7     2
81-06-02 09:53  choptrpt2  1  finished     with choptrpt2  of Field7     2
mowed areal, ha: 4.8&+00
81-06-02 09:53  choptrpt2  1  setting up   with choptrpt2  of Field7     1
81-06-02 10:11  choptrpt2  1  working      with choptrpt2  of Field7     1
81-06-02 16:37  choptrpt2  1  finished     with choptrpt2  of Field7     1
mowed areal, ha: 0.0&+00
```

APPENDIX 2: Extensive simulation log

```
                        CLOCK TIME =      0.000
*****************************************************************************
*                                                                           *
*                  T R A C I N G   C O M M E N C E S                        *
*                                                                           *
*****************************************************************************

      TIME/ CURRENT      AND ITS ACTION(S)

     0.000 DEMOS         CANCELS ITSELF
  3600.000 TheCalendr 1  SCHEDULES The Sun 1 NOW
                         SCHEDULES The RainRe 1 NOW
                         HOLDS FOR 7.500, UNTIL 3607.500
           The Sun 1     HOLDS FOR 4.654, UNTIL 3604.654
           The RainRe 1  HOLDS FOR 118.017, UNTIL 3718.017
  3604.654 The Sun 1     HOLDS FOR 16.691, UNTIL 3621.346
  3607.500 TheCalendr 1  INTERRUPTS the Ensman 1, WITH POWER = 1
                         HOLDS FOR 14.500, UNTIL 3622.000
           the Ensman 1  SCHEDULES Mow-gang   1 NOW
                         CANCELS ITSELF
           Mow-gang   1  SEIZES 1 OF workers
                         SEIZES 1 OF tractors
                         SEIZES 1 OF mower
                         COOPTS Field1      1 FROM growing
                         INTERRUPTS the Ensman 1, WITH POWER = 5
                         HOLDS FOR 0.200, UNTIL 3607.700
           the Ensman 1  CANCELS ITSELF
  3607.700 Mow-gang   1  HOLDS FOR 1.600, UNTIL 3609.300
  3609.300               SCHEDULES Field1      1 NOW
                         RELEASES 1 TO workers
                         RELEASES 1 TO tractors
                         RELEASES 1 TO mower
                         INTERRUPTS the Ensman 1, WITH POWER = 2
                         CANCELS ITSELF
           Field1      1 CANCELS ITSELF
           the Ensman 1  SCHEDULES Mow-gang   1 NOW
                         CANCELS ITSELF
           Mow-gang   1  SEIZES 1 OF workers
                         SEIZES 1 OF tractors
                         SEIZES 1 OF mower
                         COOPTS Field2      1 FROM growing
                         INTERRUPTS the Ensman 1, WITH POWER = 5
                         HOLDS FOR 0.200, UNTIL 3609.500
           the Ensman 1  SCHEDULES choptrpt1  1 NOW
                         CANCELS ITSELF
           choptrpt1   1 SEIZES 2 OF workers
                         SEIZES 2 OF tractors
                         SEIZES 1 OF chopper
                         SEIZES 2 OF wagons
                         COOPTS Field1      1 FROM mowed
                         INTERRUPTS the Ensman 1, WITH POWER = 5
                         HOLDS FOR 0.300, UNTIL 3609.600
           the Ensman 1  CANCELS ITSELF
  3609.500 Mow-gang   1  HOLDS FOR 0.400, UNTIL 3609.900
```

```
3609.600 choptrpt1  1 HOLDS FOR 2.560, UNTIL 3612.160
3609.900 Mow-gang   1 SCHEDULES Field2     1 NOW
                      RELEASES 1 TO workers
                      RELEASES 1 TO tractors
                        RELEASES 1 TO mower
                      INTERRUPTS the Ensman 1, WITH POWER = 2
                      CANCELS ITSELF
         Field2     1 CANCELS ITSELF
         the Ensman 1 SCHEDULES Mow-gang   1 NOW
                      CANCELS ITSELF
         Mow-gang   1 SEIZES 1 OF workers
                      SEIZES 1 OF tractors
                      SEIZES 1 OF mower
                      COOPTS Field3     1 FROM growing
                      INTERRUPTS the Ensman 1, WITH POWER = 5
                      HOLDS FOR 0.200, UNTIL 3610.100
         the Ensman 1 CANCELS ITSELF
3610.100 Mow-gang   1 HOLDS FOR 1.360, UNTIL 3611.460
3611.460              SCHEDULES Field3     1 NOW
                      RELEASES 1 TO workers
                      RELEASES 1 TO tractors
                      RELEASES 1 TO mower
                      INTERRUPTS the Ensman 1, WITH POWER = 2
                      CANCELS ITSELF
         Field3     1 CANCELS ITSELF
         the Ensman 1 SCHEDULES Mow-gang   1 NOW
                      CANCELS ITSELF
         Mow-gang   1 SEIZES 1 OF workers
                      SEIZES 1 OF tractors
                      SEIZES 1 OF mower
                      COOPTS Field4     1 FROM growing
                      INTERRUPTS the Ensman 1, WITH POWER = 5
                      HOLDS FOR 0.200, UNTIL 3611.660
         the Ensman 1 CANCELS ITSELF
3611.660 Mow-gang   1 HOLDS FOR 2.000, UNTIL 3613.660
3612.160 choptrpt1  1 SCHEDULES Field1     1 NOW
                      RELEASES 2 TO workers
                      RELEASES 2 TO tractors
                      RELEASES 1 TO chopper
                      RELEASES 2 TO wagons
                      INTERRUPTS the Ensman 1, WITH POWER = 2
                      CANCELS ITSELF
         Field1     1 CANCELS ITSELF
         the Ensman 1 SCHEDULES choptrpt1  1 NOW
                      CANCELS ITSELF
         choptrpt1  1 SEIZES 2 OF workers
                      SEIZES 2 OF tractors
                      SEIZES 1 OF chopper
                      SEIZES 2 OF wagons
                      COOPTS Field2     1 FROM mowed
                      INTERRUPTS the Ensman 1, WITH POWER = 5
                      HOLDS FOR 0.300, UNTIL 3612.460
         the Ensman 1 CANCELS ITSELF
3612.460 choptrpt1  1 HOLDS FOR 0.887, UNTIL 3613.347
3613.347              SCHEDULES Field2     1 NOW
                      RELEASES 2 TO workers
```

```
                    RELEASES 2 TO tractors
                    RELEASES 1 TO chopper
                    RELEASES 2 TO wagons
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field2     1 CANCELS ITSELF
          the Ensman 1 INTERRUPTS Mow-gang   1, WITH POWER = 1
                    CANCELS Mow-gang   1
                      CANCELS ITSELF
          Mow-gang   1 SCHEDULES Field4     2 NOW
                    SCHEDULES Field4     1 NOW
                    RELEASES 1 TO workers
                    RELEASES 1 TO tractors
                    RELEASES 1 TO mower
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field4     2 CANCELS ITSELF
          Field4     1 CANCELS ITSELF
          the Ensman 1 SCHEDULES choptrpt2  1 NOW
                    CANCELS ITSELF
          choptrpt2  1 SEIZES 3 OF workers
                    SEIZES 3 OF tractors
                    SEIZES 1 OF chopper
                    SEIZES 3 OF wagons
                    COOPTS Field3     1 FROM mowed
                    INTERRUPTS the Ensman 1, WITH POWER = 5
                    HOLDS FOR 0.300, UNTIL 3613.648
          the Ensman 1 CANCELS ITSELF
3613.648  choptrpt2  1 HOLDS FOR 2.688, UNTIL 3616.335
3616.335            SCHEDULES Field3     1 NOW
                    RELEASES 3 TO workers
                    RELEASES 3 TO tractors
                    RELEASES 1 TO chopper
                    RELEASES 3 TO wagons
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field3     1 CANCELS ITSELF
          the Ensman 1 SCHEDULES Mow-gang   1 NOW
                    CANCELS ITSELF
          Mow-gang   1 SEIZES 1 OF workers
                    SEIZES 1 OF tractors
                    SEIZES 1 OF mower
                    COOPTS Field4     2 FROM growing
                    INTERRUPTS the Ensman 1, WITH POWER = 5
                    HOLDS FOR 0.200, UNTIL 3616.535
          the Ensman 1 SCHEDULES choptrpt1  1 NOW
                    CANCELS ITSELF
          choptrpt1  1 SEIZES 2 OF workers
                    SEIZES 2 OF tractors
                    SEIZES 1 OF chopper
                    SEIZES 2 OF wagons
                    COOPTS Field4     1 FROM mowed
                    INTERRUPTS the Ensman 1, WITH POWER = 5
                    HOLDS FOR 0.300, UNTIL 3616.635
          the Ensman 1 CANCELS ITSELF
3616.535  Mow-gang   1 HOLDS FOR 0.313, UNTIL 3616.848
```

```
3616.635 choptrpt1  1 HOLDS FOR 2.044, UNTIL 3618.679
3616.848 Mow-gang   1 SCHEDULES Field4     2 NOW
                      RELEASES 1 TO workers
                      RELEASES 1 TO tractors
                      RELEASES 1 TO mower
                      INTERRUPTS the Ensman 1, WITH POWER = 2
                      CANCELS ITSELF
         Field4     2 CANCELS ITSELF
         the Ensman 1 SCHEDULES Mow-gang   1 NOW
                      CANCELS ITSELF
         Mow-gang   1 SEIZES 1 OF workers
                      SEIZES 1 OF tractors
                        SEIZES 1 OF mower
                      COOPTS Field5     1 FROM growing
                      INTERRUPTS the Ensman 1, WITH POWER = 5
                      HOLDS FOR 0.200, UNTIL 3617.048
         the Ensman 1 CANCELS ITSELF
3617.048 Mow-gang   1 HOLDS FOR 0.800, UNTIL 3617.848
3617.848              SCHEDULES Field5     1 NOW
                      RELEASES 1 TO workers
                      RELEASES 1 TO tractors
                      RELEASES 1 TO mower
                      INTERRUPTS the Ensman 1, WITH POWER = 2
                      CANCELS ITSELF
         Field5     1 CANCELS ITSELF
         the Ensman 1 SCHEDULES Mow-gang   1 NOW
                      CANCELS ITSELF
         Mow-gang   1 SEIZES 1 OF workers
                      SEIZES 1 OF tractors
                      SEIZES 1 OF mower
                      COOPTS Field6     1 FROM growing
                      INTERRUPTS the Ensman 1, WITH POWER = 5
                      HOLDS FOR 0.200, UNTIL 3618.048
         the Ensman 1 CANCELS ITSELF
3618.048 Mow-gang   1 HOLDS FOR 1.200, UNTIL 3619.248
3618.679 choptrpt1  1 SCHEDULES Field4     1 NOW
                      RELEASES 2 TO workers
                      RELEASES 2 TO tractors
                      RELEASES 1 TO chopper
                      RELEASES 2 TO wagons
                      INTERRUPTS the Ensman 1, WITH POWER = 2
                      CANCELS ITSELF
         Field4     1 CANCELS ITSELF
         the Ensman 1 SCHEDULES choptrpt1  1 NOW
                      CANCELS ITSELF
         choptrpt1  1 SEIZES 2 OF workers
                      SEIZES 2 OF tractors
                      SEIZES 1 OF chopper
                      SEIZES 2 OF wagons
                      COOPTS Field4     2 FROM mowed
                      INTERRUPTS the Ensman 1, WITH POWER = 5
                      HOLDS FOR 0.300, UNTIL 3618.980
         the Ensman 1 CANCELS ITSELF
3618.980 choptrpt1  1 HOLDS FOR 0.432, UNTIL 3619.412
3619.248 Mow-gang   1 SCHEDULES Field6     1 NOW
                      RELEASES 1 TO workers
```

```
                    RELEASES 1 TO tractors
                    RELEASES 1 TO mower
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field6    1 CANCELS ITSELF
          the Ensman 1 SCHEDULES Mow-gang   1 NOW
                    CANCELS ITSELF
          Mow-gang  1 SEIZES 1 OF workers
                    SEIZES 1 OF tractors
                    SEIZES 1 OF mower
                    COOPTS Field7     1 FROM growing
                    INTERRUPTS the Ensman 1, WITH POWER = 5
                    HOLDS FOR 0.200, UNTIL 3619.448
          the Ensman 1 CANCELS ITSELF
3619.412  choptrpt1  1 SCHEDULES Field4     2 NOW
                    RELEASES 2 TO workers
                      RELEASES 2 TO tractors
                    RELEASES 1 TO chopper
                    RELEASES 2 TO wagons
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field4    2 CANCELS ITSELF
          the Ensman 1 SCHEDULES choptrpt1  1 NOW
                    CANCELS ITSELF
          choptrpt1  1 SEIZES 2 OF workers
                    SEIZES 2 OF tractors
                    SEIZES 1 OF chopper
                    SEIZES 2 OF wagons
                    COOPTS Field5     1 FROM mowed
                    INTERRUPTS the Ensman 1, WITH POWER = 5
                    HOLDS FOR 0.300, UNTIL 3619.712
          the Ensman 1 CANCELS ITSELF
3619.448  Mow-gang   1 HOLDS FOR 2.640, UNTIL 3622.087
3619.712  choptrpt1  1 HOLDS FOR 1.182, UNTIL 3620.894
3620.894              SCHEDULES Field5     1 NOW
                    RELEASES 2 TO workers
                    RELEASES 2 TO tractors
                    RELEASES 1 TO chopper
                    RELEASES 2 TO wagons
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field5    1 CANCELS ITSELF
          the Ensman 1 SCHEDULES choptrpt1  1 NOW
                    CANCELS ITSELF
          choptrpt1  1 SEIZES 2 OF workers
                    SEIZES 2 OF tractors
                    SEIZES 1 OF chopper
                    SEIZES 2 OF wagons
                    COOPTS Field6     1 FROM mowed
                    INTERRUPTS the Ensman 1, WITH POWER = 5
                    HOLDS FOR 0.300, UNTIL 3621.194
          the Ensman 1 CANCELS ITSELF
3621.194  choptrpt1  1 HOLDS FOR 2.412, UNTIL 3623.606
3621.346  The Sun 1    INTERRUPTS the Ensman 1, WITH POWER = 4
                    HOLDS FOR 2.664, UNTIL 3624.010
          the Ensman 1 INTERRUPTS Mow-gang   1, WITH POWER = 1
```

```
                    CANCELS Mow-gang   1
                    INTERRUPTS choptrpt1  1, WITH POWER = 1
                    CANCELS choptrpt1  1
                    CANCELS ITSELF
          Mow-gang   1 SCHEDULES Field7     2 NOW
                    SCHEDULES Field7     1 NOW
                    RELEASES 1 TO workers
                    RELEASES 1 TO tractors
                    RELEASES 1 TO mower
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          choptrpt1  1 SCHEDULES Field6     2 NOW
                    SCHEDULES Field6     1 NOW
                    RELEASES 2 TO workers
                    RELEASES 2 TO tractors
                    RELEASES 1 TO chopper
                    RELEASES 2 TO wagons
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS the Ensman 1
                    CANCELS ITSELF
            Field7     2 CANCELS ITSELF
          Field7     1 CANCELS ITSELF
          Field6     2 CANCELS ITSELF
          Field6     1 CANCELS ITSELF
          the Ensman 1 CANCELS ITSELF
3622.000  TheCalendr 1 HOLDS FOR 9.500, UNTIL 3631.500
3624.010  The Sun 1    HOLDS FOR 4.625, UNTIL 3628.635
3628.635               HOLDS FOR 16.729, UNTIL 3645.365
3631.500  TheCalendr 1 INTERRUPTS the Ensman 1, WITH POWER = 1
                    HOLDS FOR 14.500, UNTIL 3646.000
          the Ensman 1 CANCELS ITSELF
3645.365  The Sun 1    INTERRUPTS the Ensman 1, WITH POWER = 4
                    HOLDS FOR 2.645, UNTIL 3648.010
          the Ensman 1 SCHEDULES choptrpt2  1 NOW
                    CANCELS ITSELF
          choptrpt2  1 SEIZES 3 OF workers
                    SEIZES 3 OF tractors
                    SEIZES 1 OF chopper
                    SEIZES 3 OF wagons
                    COOPTS Field6     2 FROM mowed
                    INTERRUPTS the Ensman 1, WITH POWER = 5
                    HOLDS FOR 0.300, UNTIL 3645.665
          the Ensman 1 CANCELS ITSELF
3645.665  choptrpt2  1 HOLDS FOR 0.683, UNTIL 3646.348
3646.000  TheCalendr 1 INTERRUPTS choptrpt2  1, WITH POWER = 1
                    CANCELS choptrpt2  1
                    HOLDS FOR 9.500, UNTIL 3655.500
          choptrpt2  1 SCHEDULES Field6     3 NOW
                    SCHEDULES Field6     2 NOW
                    RELEASES 3 TO workers
                    RELEASES 3 TO tractors
                    RELEASES 1 TO chopper
                    RELEASES 3 TO wagons
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field6     3 CANCELS ITSELF
```

```
         Field6      2 CANCELS ITSELF
         the Ensman 1 CANCELS ITSELF
3648.010 The Sun 1    HOLDS FOR 4.607, UNTIL 3652.617
3652.617              HOLDS FOR 16.766, UNTIL 3669.383
3655.500 TheCalendr 1 INTERRUPTS the Ensman 1, WITH POWER = 1
                      HOLDS FOR 14.500, UNTIL 3670.000
         the Ensman 1 SCHEDULES Mow-gang   1 NOW
                      CANCELS ITSELF
         Mow-gang    1 SEIZES 1 OF workers
                      SEIZES 1 OF tractors
                      SEIZES 1 OF mower
                      COOPTS Field7     2 FROM growing
                      INTERRUPTS the Ensman 1, WITH POWER = 5
                      HOLDS FOR 0.200, UNTIL 3655.700
         the Ensman 1 SCHEDULES choptrpt1  1 NOW
                      CANCELS ITSELF
         choptrpt1   1 SEIZES 2 OF workers
                      SEIZES 2 OF tractors
                      SEIZES 1 OF chopper
                      SEIZES 2 OF wagons
                      COOPTS Field6     3 FROM mowed
                      INTERRUPTS the Ensman 1, WITH POWER = 5
                      HOLDS FOR 0.300, UNTIL 3655.800
         the Ensman 1 CANCELS ITSELF
  3655.700 Mow-gang   1 HOLDS FOR 0.742, UNTIL 3656.442
3655.800 choptrpt1   1 HOLDS FOR 0.467, UNTIL 3656.267
3656.267              SCHEDULES Field6     3 NOW
                      RELEASES 2 TO workers
                      RELEASES 2 TO tractors
                      RELEASES 1 TO chopper
                      RELEASES 2 TO wagons
                      INTERRUPTS the Ensman 1, WITH POWER = 2
                      CANCELS ITSELF
         Field6      3 CANCELS ITSELF
         the Ensman 1 SCHEDULES choptrpt1  1 NOW
                      CANCELS ITSELF
         choptrpt1   1 SEIZES 2 OF workers
                      SEIZES 2 OF tractors
                      SEIZES 1 OF chopper
                      SEIZES 2 OF wagons
                      COOPTS Field7     1 FROM mowed
                      INTERRUPTS the Ensman 1, WITH POWER = 5
                      HOLDS FOR 0.300, UNTIL 3656.567
         the Ensman 1 CANCELS ITSELF
3656.442 Mow-gang    1 SCHEDULES Field7     2 NOW
                      RELEASES 1 TO workers
                      RELEASES 1 TO tractors
                      RELEASES 1 TO mower
                      INTERRUPTS the Ensman 1, WITH POWER = 2
                      CANCELS ITSELF
         Field7      2 CANCELS ITSELF
         the Ensman 1 INTERRUPTS choptrpt1  1, WITH POWER = 1
                      CANCELS choptrpt1  1
                      CANCELS ITSELF
         choptrpt1   1 SCHEDULES Field7     1 NOW
                      RELEASES 2 TO workers
```

```
                    RELEASES 2 TO tractors
                    RELEASES 1 TO chopper
                    RELEASES 2 TO wagons
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field7     1 CANCELS ITSELF
          the Ensman 1 SCHEDULES choptrpt2  1 NOW
                    CANCELS ITSELF
          choptrpt2  1 SEIZES 3 OF workers
                    SEIZES 3 OF tractors
                    SEIZES 1 OF chopper
                    SEIZES 3 OF wagons
                    COOPTS Field7     2 FROM mowed
                    INTERRUPTS the Ensman 1, WITH POWER = 5
                    HOLDS FOR 0.300, UNTIL 3656.742
          the Ensman 1 CANCELS ITSELF
3656.742  choptrpt2  1 HOLDS FOR 1.150, UNTIL 3657.892
3657.892            SCHEDULES Field7     2 NOW
                    RELEASES 3 TO workers
                    RELEASES 3 TO tractors
                    RELEASES 1 TO chopper
                    RELEASES 3 TO wagons
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field7     2 CANCELS ITSELF
          the Ensman 1 SCHEDULES choptrpt2  1 NOW
                    CANCELS ITSELF
          choptrpt2  1 SEIZES 3 OF workers
                      SEIZES 3 OF tractors
                    SEIZES 1 OF chopper
                    SEIZES 3 OF wagons
                    COOPTS Field7     1 FROM mowed
                    INTERRUPTS the Ensman 1, WITH POWER = 5
                    HOLDS FOR 0.300, UNTIL 3658.192
          the Ensman 1 CANCELS ITSELF
3658.192  choptrpt2  1 HOLDS FOR 6.440, UNTIL 3664.632
3664.632            SCHEDULES Field7     1 NOW
                    RELEASES 3 TO workers
                    RELEASES 3 TO tractors
                    RELEASES 1 TO chopper
                    RELEASES 3 TO wagons
                    INTERRUPTS the Ensman 1, WITH POWER = 2
                    CANCELS ITSELF
          Field7     1 CANCELS ITSELF
          the Ensman 1 CANCELS ITSELF
3669.383  The Sun 1    INTERRUPTS the Ensman 1, WITH POWER = 4
                    HOLDS FOR 2.627, UNTIL 3672.010
          the Ensman 1 CANCELS ITSELF
3670.000  TheCalendr 1 HOLDS FOR 9.500, UNTIL 3679.500
3672.010  The Sun 1    HOLDS FOR 4.590, UNTIL 3676.600
3676.600            HOLDS FOR 16.800, UNTIL 3693.400

                  CLOCK TIME =   3679.500
```