CHAPTER 3

IMPLEMENTATION OF MICROANALYTIC MODELS USING MASH

The class of socioeconomic models to which simulation methods can be applied using MASH may be characterized as follows. The models consist of two components: (1) a microanalytic model consisting of processes, or operating characteristics, that are applied to individual entities in the micropopulation; and (2) one or more disjoint aggregate models that operate upon variables in a time series data bank. In particular, the research work at the Urban Institute which fostered the development of MASH (described in [O6]) consists of a primary microanalytic model of the U.S. household sector and a secondary aggregate model of moderate size of the U.S. economy.

The microanalytic model developed at the Urban Institute contains a detailed specification of individual and family behavior and is used to predict the effects of changes of state within the household, whether produced by other state changes within the entity or exogenously. The aggregate model serves two purposes: (1) it is a proxy for other economic sectors that are not represented in microanalytic form but might be so represented in an expanded model, e.g. the market for labor; and (2) it provides a mechanism for introducing government policy actions into the simulation in order to affect individual and household behavior. Both of these functions require that micro submodels include linkages between current and prior values of aggregate variables and micro unit changes of state.

The role of the aggregate model as proxy for a variety of other sectors is an important one. In the real world, individuals obtain employment from specific firms that participate in specific industries. Likewise, individuals and families obtain loans from specific credit institutions and invest in specific assets. In general, transactions between individuals and other sectors of the economy involve one or a few micro units in the other sector, although some units -- notably the Federal government -- are large enough to dominate their sector. By including the "behavior" of these entities and their markets in aggregate form, these sectors become homogeneous in their interaction with the household sector. Such treatment of these sectors is realistic for many investigations.

Thus, a model having this structure may be thought of as an aggregate model, with the aggregate equations representing the household sector replaced by a detailed microanalytic model that produces values for the same aggregate variables on the basis of micro relationships. In addition to including relationships that can be made dependent upon policy actions at the micro level, the microanalytic model can provide a wealth of distributional and cross-classified detail that cannot be obtained from an aggregate model.

To the extent that the essential microanalytic relationships are known, any sector of such an aggregate model may be represented by a microanalytic submodel based upon an adequate representation of units from within it. The remainder of the macroeconomic model then performs the functions of unifying the micro submodels and providing proxy aggregate relationships for sectors not having micro submodel representations.

The characteristics of such a mixed micro-macro model depend upon the specification of the interaction between all of its sectors. The nature of this interaction could range from completely independent submodels to a simultaneous solution of the various submodels, either analytically or by iterative procedures. The simulation system will allow almost any pattern of interaction between submodels, but the nature of the interactions is an integral part of the specification process and must be defined explicitly. While it is possible to specify the model in such a way that the simulated solution to the model is an approximation to a simultaneous and consistent solution, the simulation system does not require it and such a procedure is not a part of the current source model specification.

This chapter describes the process of translating models of this class into computer instructions and of "solving" such models, i.e. projecting their states in future time. This process involves translating the specification of each operating characteristic into a specific computer format, specifying the aggregate submodel(s), and connecting these modules together by means of a procedure for solving the model. This procedure is called an *agenda* for simulation.

The Model Solution Process

The technique of simulation is a method by which the implications of a model may be derived. Simulation is an efficient tool for "solving" models when the model cannot be solved by analytic methods due to intractability or size or both and when discrete state methods such as Markov analysis cannot be applied because of problems of size.

The model solution process, although an important part of economic research, is but one part of it. The model presented in [O6] is the result of a substantial amount of research in sociology, demography and economics, both by its authors and others. It is an analytic model that is independent of solution techniques.[1] In order to distinguish this form of a model from others, we denote it by the term *source*

[1] While the formulation of a model may be independent of the particular solution technique that is chosen to be applied to it, the method of modelling is influenced to a substantial degree by the existence of solution techniques available at the time. Interest in solving models can often lead to the development of more powerful solution techniques, such as in the development of mathematical programming algorithms and programming language translators.

model; it is the source of the specification which is to be transformed into a solvable form. Thus, the model specified in [O6] is the source description of a model that is implemented using MASH.

The *source model* consists of a consistent and complete set of written specifications that describe a set of dynamic processes that can be applied to both micro and macro entities whose future states are the objects of interest. Using these specifications of the source model and a complete description of the initial micropopulation and aggregate variables, it is theoretically possible to apply these processes manually to each microentity and to generate the future states of the micropopulation and aggregate environment implied by the model for future time. While such a process may be instructive for very small models having only a few entities, it becomes prohibitively slow and inaccurate when the model's size increases only moderately.[2] Automatic computing resources are required for solution by simulation of almost all useful microanalytic models.

A translation process is required in order to transform the source model into a form suitable for simulation using a computer system. When applied to the microanalytic model, the output of such a translation process is a series of computer program modules which are structured in a form acceptable for embedding within MASH. The set of computer program modules is referred to as the object form of the model or the *object model*. The object model is functionally equivalent to its source model in that solutions produced by simulations using the object model embedded within MASH are identical to those that would be obtained with a correct manual application of the source model to a micropopulation with identical initial composition and attribute states.[3]

In the following discussion of the model solution process, or simulation process, it is essential to distinguish between several concepts of *time*: (1) computational time; (2) simulated time or simulation state time; and (3) execution time, which has several dimensions.

---

[2] Such simulations have been found to be quite useful when the microentities are represented by individuals within a gaming environment. In such a scenario, the gaming model may or may not be implemented on a computer. The objective of such a gaming exercise generally differs from that of a simulation, in that it attempts to generate an understanding by the players of the dynamics of the model and the interrelationships between and the consequences of their actions.

[3] As in other disciplines, the act of translating model specifications from written form to a form in which they are executable automatically puts a burden upon both the specifier of the model and the translator. Incomplete specifications and inconsistencies that are not apparent in the derivation of and initial experimentation with the source model often become obvious and destructive when translated into object form. For this reason and others, the translation process is not a trivial task. It also involves feedback from the translator to the model specifier as the implications of both the model and its inconsistencies are observed.

Computational time is an ordinal concept that refers to the dynamic order in which computer program instructions are executed during the course of a simulation exercise. All events that are simulated during a run occur in a specific order that is determined by the object model and the simulation control program contained in MASH. The occurrences of these events are ordered in computational time.

Simulated time is a cardinal concept. It is the instant or interval in calendar time with which the current state of the model's entities should be associated. Simulated time may be measured relative to a zero point defined at the beginning of the simulation process or it may be measured in ordinary calendar time.

The simulation mechanism within MASH is limited to controlling fixed time period models in which the state of the model's entities is advanced in discrete, fixed periods of calendar time. The discrete time period chosen for the current model embedded in MASH is one calendar year. Thus, for these simulations, simulated time advances in discrete steps of one calendar year after all entities have been processed and have been moved forward in time one year by the object model.

Not all simulation states have a simulated time associated with them. For example, a simulation exercise that is defining new attribute values for population entities for the next year of simulated time may have some of its entities defined as of the next year, while some of its entities may reflect partially their state as of the "current" simulated calendar year. Since the computer program cannot -- and does not need to -- operate simultaneously and instantaneously upon all entities in the simulation population, the concept of simulated time generally only has meaning at the beginning or end of a simulated year. Further, during the computational time required to advance the population by one simulated year, any individual entity may be at the simulated time at the beginning of the period, at the end of the period, or somewhere in between and therefore undefined in simulated time.

Execution time is a cardinal concept that has at least two dimensions, central processing unit time and wall clock time. These times measure resource utilization and throughput associated with executing the MASH system with an object model, and they are not discussed in this chapter. Figure 3-1 illustrates graphically the relationships between these various concepts of time for a hypothetical simulation exercise.

Figure 3-1. Illustration of Different Time Concept for a Hypothetical Simulation

Simulated time is but one aspect of the characteristics of a simulation exercise or solution process that is in progress. These characteristics are denoted by the term *simulation state*, which refers to the totality of information defining a specific simulation exercise at any point during its execution. Included in the simulation state of the microanalytic submodel are: (1) all values of all attributes of all interview units, families, and persons that either exist or existed previously during the exercise; (2) the values of all parameters and the setting of all options within the micro and macro models; (3) all values of exogenous and endogenous variables used and generated by the macro model; (4) all summary statistics generated for the model up to the point in computational time when the simulation state is recorded; (5) the control sequence that has been used to direct the simulation exercise; (6) the versions of the operating characteristics and the macro model incorporated within the simulation; and (7) any other outputs of the execution of the simulation system that depend upon the particular simulation exercise being executed. In other terms, the set of PDP-10 machine readable files used by and generated during a simulation exercise may be divided into

two parts: (1) the "pure" or invariant part that is independent of simulation specifications; and (2) the "impure" or variable part that is a function of the specification of the run. The latter part, which consists of both entire files and collections of words within files, comprises the *state information* for that simulation run.

Construction of Simulation Agendas

The *simulation agenda* consists of the set of operations by which a simulation run is defined and executed. It consists of: (1) instructions that control the sequence in which micropopulation entities are processed; (2) ordered lists of micro submodel operating characteristics that govern the order of their application to micropopulation entities; and (3) instructions that involve aggregate submodels.

The three agenda components specified above are implemented in different ways within the MASH system. The instructions that determine the sequence in which micropopulation entities are processed are a fundamental part of the system and are subject to only limited change (without substantial loss of efficiency) by operating characteristics. The second agenda component consists of one sub-agenda for each processing pass made through the micropopulation. Each sub-agenda contains the name and order of each operating characteristic invoked during that pass and the computational time when it is invoked. Sub-agendas are specified in the form of Fortran subprograms which are loaded with other MASH program modules to form a run-time system specific to the microanalytic model. The third agenda component consists of the definition of one or more aggregate submodels. These submodels are easily specified at the MASH command language level using the PREPARE and associated commands. Such a specification is illustrated in detail in the previous chapter.

Micropopulation Entity Processing. Within a micropopulation, interview units are ordered according to names, which are chosen to be positive integers for technical convenience. [4] Families are ordered within each interview unit, and persons are ordered within each family. Each family and each person in the micropopulation has a name, which is also a positive integer. [5]

---

[4] In general, this ordering corresponds to the ordering of interview units in the survey file from which these population data were extracted. However, the ordering has no substantive meaning within the model, and has no relationship to any potential ordering or geographic location attributes within the micro population.

[5] The three level hierarchical structure of the micropopulation entities described here and upon which the simulation system is built has more general application than to just the household sector. Many entities of economic interest may be cast into a hierarchical mode. For example corporations may be regarded as containing a set of companies, each of which operates a set of production plants. Alternatively, firms own one or more plants, each of which manufactures a set of products. Hierarchical relationships abound in real life, and as a simulation system, MASH is capable of representing any such structures that can be characterized as a set of three level hierarchical trees. In this sense the acronym MASH is a misnomer, since simulation object models embedded in MASH may be chosen from a considerably larger class of models than household behavior models.

Within each interview unit, entities are processed in *left-list order*. This order is obtained by traversing the hierarchical structure found by the entities in the unit in such a way that the leftmost untraversed path is always chosen at each node. Figure 3-2 illustrates this traversal rule for two interview units. The numbers beside each micro entity indicate the relative sequence in computational time in which processing occurs according to this rule. Processing of family data can occur both before and after processing of data of persons within the family, and processing of interview unit data can occur both before and after processing of data of families within the interview unit.

Figure 3-2. Processing Micropopulation Entities in Left-List Order

The complete outline of the simulation control apparatus is shown by the schematic diagram in Figure 3-3. The basic control structure consists of five nested loops. In the outermost loop the object model is applied to the micropopulation, one or more aggregate models are solved, and the simulation state of the model is advanced by exactly one year. The next loop controls the several passes that are made over the micropopulation to cause this advance in the simulation state. Each pass may have an aggregate submodel associated with it.[6] Within each pass over the micropopulation, interview units are processed in order of name. Within each interview unit, families are processed in order of membership within the unit; and within each family, persons are processed in order of their membership within the family. The last two innermost processing loops correspond to the left-list processing order illustrated in Figure 3-2.

[6] The present model described in [O6] is implemented using three passes over the micropopulation with one aggregate submodel being solved between the second and third pass.

Figure 3-3. Schematic Diagram of Simulation Control Flow

The simulation control apparatus illustrated in Figure 3-3 establishes a structure in which both object forms of microanalytic operating characteristics and aggregate submodels can be embedded. Microanalytic operating characteristics are embedded in the micro sub-agendas, one of which exists for every pass over the micropopulation. Within each sub-agenda, an operating characteristic may be scheduled at any of 11 instances, or scheduling points, in computational time corresponding to the 11 numbers in Figure 3.3. [7] The state of simulation progress corresponding to these 11 scheduling points is as follows:

[7] Scheduling points 1, 2, 10, and 11 occur outside the loop over micro submodel passes and therefore are indistinguishable in computational time among the micro sub-agendas corresponding to the passes over the micropopulation. The existence of these scheduling points is maintained in each sub-agenda so that operating characteristics can be grouped by function to display the logical structure of the model more clearly. Alternatively, the redundant scheduling points need not be used.

1. The simulation of the microanalytic model has just begun, and the calendar year is equal to the year corresponding to the initial simulation state. Initial conditions for an operating characteristic can be specified at this time.

2. A new year of simulation is about to begin. Initial calculations for an operating characteristic that are time dependent can be performed now.

3. A new pass over the micropopulation is about to begin. If an operating characteristic is included in more than one pass, any initialization that is pass dependent can occur now.

4. Operating characteristics in the current pass are now to be applied to the next interview unit. Initial processing for simulating the changes of state of this unit can be performed now, as well as defining state changes at the interview unit level.

5. Operating characteristics in the current pass are now to be applied to the next family within the current interview unit. Initial processing for simulating changes of state of this family can be performed now, as well as defining state changes at the family level.

6. Operating characteristics in the current pass are now to be applied to the next person in the current family. The computation of changes of state of this person's attributes can now be performed.

7. Operating characteristics in this pass have been applied to all persons in the current family. Additional processing and changes of state can now take place for the current family.

8. Operating characteristics in this pass have been applied to all families in the current interview unit. Additional processing and changes of state can now take place for the current interview unit.

9. Operating characteristics have been applied to all interview units in the micro population. Summary calculations for this pass can now be performed.

10. The simulation state has advanced by one year. Summary computations for the year can now be performed.

11. The simulation has been terminated. Final results of this simulation exercise can now be computed.

Many operating characteristics will be scheduled to be applied at only one of the scheduling points. For example, the education operating characteristic would be expected to be scheduled only at the person level (scheduling point 6). However, an operating characteristic that performed calculations over multiple micro units might be more conveniently translated into object form by using two or more scheduling points. For example, an operating characteristic that computed total wages earned within an interview unit might use scheduling point 4 (initial interview unit processing) to set initially the value of the total wage attribute to zero, and then use scheduling point 6 (person processing) to add the computed wages of individuals to form a running total. If average wages of members in the labor force were to be computed as an output variable, than the number of members in the labor force could be aggregated in the same manner, and scheduling point 8 (interview unit final processing) could be used to calculate the average wage by division.

Operating characteristics have access to the aggregate time series data bank for input values, and they may also calculate aggregate values and store them in the data bank. For example, if the microanalytic model is to determine an initial estimate of unemployment based upon operating characteristics defining labor force participation, scheduling point 3 can be used to set aggregate counters for labor force participation and unemployed workers, scheduling point 6 can be used to increment the appropriate counters, and scheduling point 9 can be used to multiply the entity counts by the current population weight and augment the appropriate time series in the data bank.

The existence of the specific simulation control apparatus displayed in Figure 3-3 does not imply that micropopulation entities must be accessed and their states updated in exactly the order specified. The simulation control program cycles through micropopulation entities in the order specified, and it makes available "current" entities at the appropriate scheduling points. Thus, at scheduling points 4 through 8 there is a current interview unit; at scheduling points 5 through 7 there is a current family; and at scheduling point 6 there is a current person. Entities that are current are known to have their attribute states in main memory, so that no data transfers from secondary memory are required to fetch and store data of a current entity. Subject to the limitations discussed in Chapter 4 regarding thrashing, an object model that processes entities strictly as they become current will minimize the number of input-output transfers of population data required to execute a simulation run.

Some microanalytic operating characteristics may be translated to object form in a more straightforward or efficient manner if population entities are accessed out of the standard computational sequence. The disposition of assets at death provides one example. When an interview unit is destroyed because its last living person has died, assets exist in the unit. In general, linkages will exist in the micropopulation between that interview unit and persons who were members of the unit at one time. These linkages can be examined by the operating characteristic, and assets can be transferred between the current unit and the interview units containing some or all of the previous members of the unit. Another example is provided by a possible treatment of alimony payments to divorced spouses who are by definition no longer part of the same family or interview unit as their ex-partner. The existence and amount of the payment could depend upon relative marital circumstances and levels of income of the ex-partners; if so, the alimony payment operating characteristic would access and change the values of attributes of two persons concurrently.

Deviations from the standard computational sequence over entities in the micropopulation are permitted and may be included in the object form of microanalytic operating characteristics with ease. Relatively infrequent deviations cause a minor degradation in the operating efficiency of the simulation system, and the degradation is gradual as the deviations become more numerous. Thus, while infrequent

events that require concurrent access to separate interview units can be beneficially modelled in this manner, frequent events should be translated differently into object form, lest the resulting system degrade performance so substantially as to make experiments prohibitively expensive to perform. An example of different treatment in the current model is provided by mate selection for marriage, which is performed outside the standard simulation control sequence in a separate module.

A subset of the standard simulation control sequence is executed when a micropopulation is created by the CREATE POPULATION and related commands. After the micropopulation has been extracted from the survey file and the population, membership, containment and documentation files have been created, a special pass is made through the micropopulation so that one-time initializing operating characteristics may be applied to population entities. Scheduling points 3 through 9 are available for the set of operating characteristics that perform such initial processing. Except for being associated with and loaded into an initializing module of the system, the object form of these operating characteristics is identical to the object form of operating characteristics of the microanalytic model.

Microanalytic Operating Characteristics. The microanalytic submodel consists of an interrelated set of processes that are applied to entities in the micropopulation. These processes are called operating characteristics in that they describe "the relationships between the characteristics of a component (micropopulation entity) and its behavior."[8]

Each operating characteristic embodies a set of behavioral or accounting relationships that define as outputs -- for each unit of simulation -- simulated future states, or outputs, from present states, or inputs. More rigorously, an operating characteristic is a transformation which when applied to a unit in the population alters the states (values) of its attributes according to the process it represents. Operating characteristics may include stochastic components.

The microanalytic model described in [O6] consists of five major groups of operating characteristics: (1) *demographic,* including birth, death, marriage, divorce, and family formation and dissolution; (2) *educational attainment*; (3) *labor force participation,* including occupation selection, wage determination, unemployment experience, and hours worked; (4) *income generation,* including income from labor, business and farming; income from capital, public and private transfers, and eventually some non-monetary returns from labor and capital; and (5) *income disposal* including consumption, tax payments, and type and amount of asset accumulation (saving). In addition, there exists a set of initializing operating characteristics that are used for making adjustments in newly created micropopulations prior to any simulation activity.

---

[8] G. H. Orcutt, et. al., *Microanalysis of Socioeconomic Systems: A Simulation Study.* New York: Harper and Brothers, 1961, p. 7.

The decomposition of a microanalytic submodel into specific operating characteristics is not a unique process. For functional clarity, the resulting characteristics should reflect the substantive divisions inherent in the model itself. However, there still exists much scope for differences based upon style and taste.

The experience of the research team engaged in implementing the model described in [O6] is that a high degree of modularity is both desirable and essential. It provides a means to subdivide a large and complex task into manageable subsets, and it allows experimentation with and replacement of individual modules based upon different source models as the development of the model progresses, both at the source and at the object level and often in parallel. Furthermore, as research modules are refined, they have a tendency to grow both in size and complexity, so that only modules that address and implement simple relationships are likely to remain manageable and understandable throughout the life of the development of a model. [9]

While the desirability of modularity was recognized during the creation of both the MASH system and the microanalytic model described in [O6], the extent of its necessity was underestimated. After several years of development, the size of the model -- both in terms of the number of attributes and complexity of specification -- was putting a considerable burden upon the work of the project. Boundaries of existing modules were found to be disadvantageous and required restructuring. Inadequate attention to the specification of interfaces between modules caused misspecifications that were not easily detectable.

While issues of module size, boundaries, and interfaces were solved for MASH and for the model in [O6] on an ad hoc basis, the lack of an adequate conceptual framework for specifying the order and interrelationships of operating characteristic modules will inhibit the effective development of more complex models. The following framework is suggested as a beginning step toward evolving a mechanism that will allow flexible and accurate module specification and that will permit the consistency of complex microanalytic models to be investigated.

Each operating characteristic has access to certain information which it may use as input, and each characteristic may produce a variety of outputs. Inputs may consist of any value of entity attributes that are defined at the point in computational time at which the operating characteristic is scheduled, values of time series variables from the data bank, and stochastic terms. The outputs of an operating characteristic may

---

[9] The analogy is strong between this experience in model building and the general experience of the computing community that modular development is essential to efficient programming. The construction of a large microanalytic source model is not dissimilar to the construction of a major programming system. When the source model is translated into its object form, the similarity becomes complete.

consist of updated values of entity attributes (moved forward one time period in simulated time) and newly defined values of time series in the aggregate data bank.

More formally, each operating characteristic may be characterized in terms of its inputs and outputs as a set of vectors of attribute names with associated time periods, i.e. for each scheduling point utilized, a vector may be defined as:

$$\{ -a_{i1}(t_{i1}), \ldots, -a_{im}(t_{im}), \; +a_{j1}(t_{j1}), \ldots, +a_{jn}(t_{jn}) \}$$

containing the *m* attributes $\{a_i\}$whose values are required as inputs and the *n* attributes $\{a_j\}$ whose values are produced as outputs. [10] Each of these attributes has associated with it a specific time dimension. Suppose that an operating characteristic is applied when simulated time is being advanced from time T_1 to time T_2. The $\{t_i\}$ will then correspond to period T_1 for attributes that have not yet been updated in computational time or to period T_2 for attributes that have already been updated in computational time, depending upon the input requirements of the characteristic. Output attributes having time dimension $\{t_j\}$ will correspond to period T_2.

A *structure definition table* T could then be constructed from these vectors and the micropopulation attribute set which would display the interrelationship between operating characteristics and assist in determining the consistency of their ordering. An example of such a table is illustrated in Figure 3-4. The rows of the matrix correspond to the entire set of attributes associated with each entity type within the micropopulation unit. Each column of the matrix consists of an operating characteristic in the simulation procedure that either requires values of attributes as input or produces new values of attributes as output, or both. [11]

---

[10] For the purpose of this exposition, it is assumed that the attribute values required and defined are within a single micropopulation entity, i.e. that the domain and range of each operating characteristic are both contained within a single micropopulation unit. The framework proposed here must eventually be extended to describe the inputs and outputs of more complex, inter-unit operating characteristics, since this assumption is violated in specifying models.

[11] The part of the marriage operating characteristic that actually performs the formation of new families occurs between passes in computational time. This is logically no different than if it occurred at the end of the proceeding pass or at the beginning of the following pass.

Figure 3-4. Structure Definition Table T for Microanalytic Submodel

The structure definition table may be partitioned vertically to correspond to the computational sequence in which the simulation procedure is executed:

$$T = (S \quad I \quad U \quad P_1 \quad P_2 \quad .... \quad P_m)$$

The column *S* corresponds to the process of extracting attributes from the survey micro data file. *I* corresponds to the set of operating characteristics applied during the initializing pass over the micropopulation. The partition *U* consists of processes that update micro time series attributes in the population. These processes are transparent to both the user of MASH and the builder of the object form of the micromodel, but are included in any complete micromodel specification. The partition P_I corresponds to the set of operating characteristics applied during the I'th pass over a micropopulation in the process of advancing it one simulated year. Within each partition, processes are ordered from left to right by computational time.

Each cell in the table corresponds to the interaction of one operating characteristic C (corresponding to the column in which the cell is located) with one attribute, *a*, (corresponding to the row in which the cell is located). The value of this cell is based upon possible interaction conditions:

Cell Value	*Condition*
0	C makes no reference to *a*
1	C defines $a(T_1)$ in the initial population
2	C uses $a(T_1)$ as input
3	C uses $a(T_1)$ as input and produces $a(T_2)$ as output
4	C produces $a(T_2)$ as output, not using a previously computed intermediate value of $a(T_2)$ as input
5	C produces $a(T_2)$ as output, using a previously computed intermediate value of $a(T_2)$ as input
6	C uses $a(T_2)$ as input only

For example, in Figure 3-4, attribute a_i is defined by extraction from the survey file and is not modified by characteristic C_0 during initialization. It is used as input by operating characteristic C_1 and it is defined as output for the next simulated time by characteristic C_2. Attribute a_j is not extracted from the survey file, but is defined by operating characteristic C_0 during initialization. Its updated value is defined by characteristic C_1, and it is not referenced by characteristic C_2.

Basic conditions for consistency of attribute definition of a micropopulation model can now be stated as follows. These conditions apply to each row of the table T independently.

1. There must be one or more 1 entries in sub-matrix (S I).
2. All non-zero entries in (U P_1 ... P_m) must appear in non-decreasing order. Furthermore, the value 3 may occur at most once.[12]

These conditions are believed to be sufficient to ensure consistency of the microanalytic submodel, provided that all operating characteristics are intra-unit only, the computational sequence used is identical to the one provided by MASH, and provided that the subsets of each operating characteristic executed at different scheduling points do not overlap. This last requirement is discussed in the next section.

The usefulness of determining consistency of the microanalytic submodel automatically increases as the submodel's size and complexity grows. The model reported in [O6] contains currently almost 100 attributes and well over 100 entries to operating characteristics in all the scheduling points, including initialization. The task of ensuring consistency manually is long and tedious and is prone to error during a period of model development.

Automatic consistency checking can be facilitated if each operating characteristic module contains within itself in machine readable form for each of its entry points a complete list of attribute names referenced and the manner in which they are used. Such text, entered in a specific syntax, is certainly a part of good documentation of the module; in addition, the module may be used by a computer program which translates this information into one or more columns of the structure definition table T. Applying this program to all operating characteristic modules yields all columns of the partitions (I P_1 ... P_m) of T. The codebook describing the micropopulation provides column S, and the associated attribute table contains the information regarding micro time series that can be used to generate partition U. The ordering of operating characteristics within I and P_1 ... P_m could be derived by requiring that the micro sub-agenda files conform to a syntax that may be read by a program to determine the order of invocation of operating characteristics for each scheduling point of each pass, including initialization.

Thus, the table T could be created and the micro submodel could be checked for consistency automatically provided that specific program documentation guidelines have been followed.[13] In addition,

---

[12] lt should be noted that many attributes will remain the same for a series of simulated time periods or even for the life of the micro entity. For example, sex of person is now unchanged in our model. Year of first marriage is defined at most once for a person. Marital status changes at various rates for different persons, but may remain constant for many time periods. The operational definition of consistency specified is based upon the assumption that if an attribute is not explicitly redefined for an entity during a simulated interval, it is implicitly redefined to have the same value.

[13] This procedure requires that internal object model documentation in each operating characteristic model accurately reflect the content of the characteristic. While it would be possible to create a program that would analyze Fortran source programs to determine some aspects of its interaction with attributes, it is not possible in

the table T itself serves as a useful schematic diagram of the structure of the model and the interrelationships between the various operating characteristics.

The structure definition table T as pictured in Figure 3-4 does not include any reference to the time series data bank. It could be augmented by additional rows for each aggregate time series used, and by a set of columns for each macromodel involved, one per equation. The matrix structure becomes more complex due to the multiple variable length lags allowed in macromodel equations, but the extension is possible and could be useful in providing an overall diagram of the structure of the combined submodels.

Object Operating Characteristic Specification

The object form of an operating characteristic consists of a computer program module that is executable within the framework of a system for simulation and that is functionally identical to the characteristic's source specification. The source form of an operating characteristic consists of an analytic specification of a relationship and is independent of both the technique and the mechanism of solution. However, the object form of the characteristic consists of a computer program module that is constructed according to specific rules that are dependent upon the system for simulation in which the module is to be embedded. The following description of the specification of the object form of operating characteristics is therefore a specific description of how object modules are constructed for a microanalytic object model that is to be solved using MASH.

The standard flow of control through the micropopulation that is embedded in MASH is described in the previous section and is illustrated in Figure 3-3. For each pass over the micropopulation, 11 scheduling points are provided for invoking object forms of operating characteristics or parts thereof. An operating characteristic may make use of any one of these points, any combination of them, or all of them. In general, only one or a few scheduling points will be used. The scheduling points are provided within a specific sub-agenda module for each pass over the population. Appendix 4 contains an example of the program structure of a sub-agenda module.

Each operating characteristic module must have a point of entry, or *entry point*, corresponding to each point at which it is scheduled in any sub-agenda. These entry points appear at the beginning of

---

general. A more promising approach might be to create a specification language for operating characteristics that would allow the external interfaces of characteristics to be specified in a compact, unambiguous manner that would guarantee consistency with its object module content, and then implement the module by translating automatically the specification language into a commonly used high level language. This type of approach was used by the creators of SIMSCRIPT I; its source language was first translated into Fortran and then passed to the Fortran translator for further processing. Such an approach was considered early in the development of MASH, but was dropped because of lack of sufficient understanding of the requirements of such a specification language, and a lack of appreciation (since gained) of the substantial complexity in large microanalytic models and the need to provide computer based management tools to cope with their machine readable modules.

subsets of the module that contain executable program steps corresponding to that subset of the operating characteristic. Each subset terminates with an exit point that returns control to the sub-agenda.

The entire module is composed of a common communication and data area for access by all module subsets followed by the subsets with entry and exit points, each corresponding to one scheduling point in the sub-agenda. The subsets of the module do not overlap. The common data area binds the subsets of the module together through data interchange, and is a necessary part of the module structure for the translation of many operating characteristics into object form. For example, characteristics that gather summary statistics generally have at least three entry points for the operations of initializing, incrementing, and reporting their data. These entry points are then called by the corresponding scheduling points in the sub-agenda -- at the beginning of a time period, at a micro entity level, and at the end of the time period. If no such information interchange is needed between subsets of the module, then there is generally little justification to include them in the same object module.

The creation of the exact form of an object module is in part a matter of style, since alternative implementation strategies generally exist. For example, suppose it were desired to define an attribute at the highest (interview unit) level whose value was a count of lowest level (person) entities contained within it at the simulated time corresponding to the end of a simulation period. One could define an object module with three entry points: (1) during initial interview unit processing, initialize a count variable in the common data area to zero; (2) during person processing, and after death and birth have been invoked, increment the counter by 1; and (3) during final interview unit processing, transfer the value of the count to the attribute at the interview unit level. Alternatively, one could define two object modules, the first of which would define the interview unit attribute as zero at initial interview unit processing time and the second of which would increment the attribute by one at person processing time. The former alternative was generally chosen in the implementation of the model described in [O6], since it was both a more efficient implementation and it grouped interrelated procedures within one module. While it would be possible to eliminate multiple entry point modules by replacing them with single entry modules and additional micro entity attributes, the loss of convenience and efficiency could be quite high.

Figure 3-5 illustrates the interrelationship between sub-agendas and operating characteristics by depicting the flow of control through a sub-agenda and a hypothetical operating characteristic object module called LABOR. The module has three entry points that are invoked at scheduling points 1, 6, and 10. A common communication and data area exists in the module for communication between the submodels. [14]

[14] The common data area for a specific module may be accessible from all submodules, but it is local to the module.

The procedure part of the module consists of 3 disjoint sets of executable statements, i.e. there is no overlap between different procedures associated with different entry points.

Figure 3-5. Flow of Control Through Subagendas and Operating Characteristic Object Modules

The MASH system for simulation provides a program environment in which microanalytic object operating characteristics are programmed using the general capabilities of an existing programming language, Fortran IV. Substantial data management assistance is provided by MASH. The programmer who translates operating characteristics into their object form does not need to have thorough knowledge of the entire system, but rather uses existing system functions to retrieve and store data either belonging to any entity in the micropopulation or belonging to any series in the user's aggregate time series data bank.[15]

[15] The time series data bank is at present implemented as a logical subsystem within the MASH user's dictionary.

Each object operating characteristic module is a Fortran IV subroutine. [16] The subroutine begins with the subroutine name, which has one integer argument attached to it. This argument has range 1 through 11 and is set in the micro sub-agenda to correspond to the scheduling point from which it is called. Following the subroutine statement is a series of COMMON blocks that contain global variables useful to the programmer. The first executable statement of the program consists of an (up to) 11 way branch to the logical entry points within the subroutine. In skeleton form, these statements are

```
      SUBROUTINE name (I)

      IMPLICIT INTEGER (A-W)
      IMPLICIT REAL (X-Z)
      REAL RDM

      COMMON /SIMULP/ IN, FN, PN, SPOPID
      COMMON /SIMCON/ IA, FA, PA, NY, IY, BY, CY,
     1                 PS, IB, IC, NF, NP

      GO TO (100,200,300,400,500,600,700,800,900,1000,1100),  I

100   program for entry point 1
      -----
      RETURN

200   program for entry point 2
      -----
      RETURN


1100  program for entry point 11
      -----
      RETURN
      END
```

The meaning of the variables in COMMON blocks SIMULP and SIMCON is as follows:

IA	Address of current interview unit
FA	Address of current family
PA	Address of current person
IN	Name of current interview unit
FN	Name of current family
PN	Name of current person
NY	Number of years through which the simulation is to proceed.
IY	Year, relative to zero, of current simulation state
BY	Calendar year associated with initial micropopulation
CY	Calendar year of current simulation state
PS	Number of current pass over micropopulation

[16] These subroutines may, of course, call other subroutines and functions that are required to execute it.

NF	Number of families in current interview unit, ascertained at initial interview unit processing time
NP	Number of persons in current family, ascertained at initial family processing time
SPOPID	Name (number) of current micropopulation
IB	Index for iterating over families in the current interview unit
IC	Index for iterating over persons in the current family

Operating characteristic object modules access micropopulation attributes through functions provided by MASH. This set of functions allows attributes to be referenced by name [17] and entities to be identified by address. For interview unit data, the function and subroutine:

```
K = INTUN ('attribute-name', intunit-address)
CALL STIU ('attribute-name', intunit-address, K)
```

retrieve and store attribute values in the interview unit virtual data array. All attribute values are stored as integers. The variable type used for storage and retrieval is therefore integer. Similarly, the two function-subroutine pairs:

```
K = FAMLY  ('attribute-name', family-address)
CALL STFAM ('attribute-name', family-address, K)

K = PERS   ('attribute-name', person-address)
CALL STPER ('attribute-name', person-address, K)
```

retrieve information from and store information into the family and person data arrays, respectively.

The link between entity names and addresses is straightforward. Interview units have identical names and addresses. Family and person addresses may be derived from names by using the following system functions:

```
family-address = FAMAD (1, family-name)
person-address = PERAD (1, person-name)
```

[17] As discussed in Chapter 4, the call by name facility is currently implemented by performing entry point and initial argument substitutions in the machine level code generated by these function and subroutine calls. Thus, if an attribute is never referenced during a simulation exercise, it may appear in a non-executed section of code within a program module, but it need not appear in the micropopulation. However, the current implementation produces self-modifying non-reentrant code and precludes the use of any form except a literal as the first argument of any of these functions or subroutines.

If the address is zero, the entity never existed. If the address is negative, the entity once existed but was removed from the population; and the absolute value of the result is the address that was occupied by the entity when it ceased to exist.

Membership in interview units and families may also be determined using system functions. The statements:

```
no.-of-families-in-IU = FMINU (1, interview-unit-name)
no.-of-persons in family = PRINF (1, family-name)
```

may be used to obtain a membership count. If the count is zero, the entity has no members contained in it. If the count is positive, then their names may be retrieved by increasing the value of the first argument in steps of 1. [18] For example, to extract all of the names of persons who are members of the family whose name is 57 and store them in array PNAME, the following code may be used:

```
      NOPERS = PRINF (1, 57)
      IF (NOPERS EQ. 0) GO TO other code
      DO 10 I = 1, NOPERS
10    PNAME(I) = PRINF (I + 1, 57)
```

The MASH system contains functions that allow an operating characteristic to retrieve information from and store information in the aggregate time series data bank. The modules GETVAL and PUTVAL accomplish this. Each has as its arguments: (1) the name of the owner of the series, or zero if the current user's name is to be used; (2) the name of the series; (3) the year associated with the value; (4) an operation status code which is returned denoting success or failure of the requested option, and if failure, the reason for it; and (5) the value to be retrieved or stored.[19] Values may be retrieved from series belonging to any owner, but they may only be stored in series owned by the present user.

In order to illustrate the process of translating an operating characteristic from its source to its object form, we will specify a hypothetical specification for the educational attainment characteristic and display its object form. This characteristic, called EDUCAT below, is created for the purpose of this example and resembles only superficially the corresponding operating characteristic in [O6].

---

[18] The function names here are actually virtual array references. Both FMINU and PRINF currently have 20 rows each; row 1 contains the current count of members and the members names appear in consecutive rows following the count. Depending upon the nature of prior simulation activity and the availability of space in the column, prior members of the entity may have their names also in the column. Prior members' names are distinguished by being included as negative names, i.e. negative numbers.

[19] The operation of retrieving many values from the time series data bank or storing them in it is at the present time relatively expensive. Therefore, if aggregate data are to be used as input to a microanalytic operating characteristic, it should be retrieved at the beginning of a pass, year, or simulation exercise, not for each micro entity.

EDUCAT is applied to each person in a micropopulation. It requires the updated values of three person attributes, AGE, RACE, and SEX, and one interview unit attribute, CPSAREA, as inputs, and it both uses as input and updates the values of two attributes of each person, GRADCOM and EDSTAT.[20] The entries for these attributes in the micropopulation codebook appear below. It is assumed that the values of attributes EDSTAT and GRADCOM were generated in a consistent manner when the micro population was initialized.

The assumptions underlying our hypothetical operating characteristic, EDUCAT, are simple. Each person enters the educational system with certainty at age 5, 6, or 7. Each person always advances one grade each year in school, but the person may drop out at the end of the school year after advancing that grade. The probability of advancing through any specific grade (assuming that the previous grade has been completed) is solely dependent upon the race, sex, and region of residence of the person.

The cumulative probability function describing entry into the educational system as a function of age is as follows:

Age of Person	Probability of Having Entered School
5	0.4
6	0.9
7	1.0

The region-independent probabilities of progression from one grade to the next in grades 1 through 8 are given in the following table:

	Male	Female
White	.99	.995
Non-white	.98	.99

Table 3-1. Probability of Continuing in Elementary School

In addition, it is hypothesized that the probability of dropping out of elementary school in the west (census region) is only one-half of the probability in other regions of the country.

---

[20] All of these attribute definitions except EDSTAT are used in the microanalytic model described in [O6]. In place of EDSTAT there exist a small, more complex, set of attributes for each person.

Figure 3-6. Micropulation Attribute Definitions Used in Hypothetical Educational Attainment Characteristic, Page 1

Figure 3-6. Micropulation Attribute Definitions Used in Hypothetical Educational Attainment Characteristic, Page 2

The probabilities of completing grades 9 through 18 is solely a function of race and sex and are given in the following table.

Probability of Finishing Grade	Male		Female	
	White	Non-White	White	Non-White
9	.98	.95	.99	.98
10-11	.99	.95	.995	.99
12	.95	.90	.90	.90
13	.50	.50	.30	.20
14-16	.80	.90	.90	.70
17-18	.90	.95	.95	.90

Table 3-2. Probability of Continuing in School Above Elementary Level

We require three more tasks of our hypothetical operating characteristic EDUCAT. First, it should maintain a count of the expected number of continuations and the actual number produced over all grades by use of the pseudo-random number generator. Each year, the probabilities of continuing should be normalized by this ratio over all past years so that variations in the expected number due to stochastic effects can be corrected as the simulation progresses. This procedure is often called tracking; it is a simple feedback mechanism that ensures that the actual number of continuations imputed during simulation will follow the expected number.[21] Second, EDUCAT should generate four aggregate time series on a yearly basis containing the enrollment in elementary school (grades 1-8), high school (grades 9-12), college (grades 13-16), and graduate school (grades 17+).

The third requirement is that EDUCAT have two options built into it for testing policy alternatives. The first option, if selected, is to give women the same probability of remaining in the educational system as men. The second option, if selected, is to give every person the same probability of remaining in the educational system as white males. The existence of these options will give policy analysts an opportunity to determine the effect of differential probabilities of remaining in the educational system -- for whatever reasons that the differences exist -- upon a variety of attributes such as employment history, lifetime earnings distribution, and wealth accumulation.

The following pages contain the Fortran IV source code for the object module for EDUCAT. Comments and explanations are contained in interspersed text.

```
      SUBROUTINE   EDUCAT   (POINT)
```

[21] Such a tracking mechanism is only one of several methods of minimizing the stochastic variation that arises from use of such a generator.

The variable POINT has domain 1,2, ..., 11 corresponding to the 11 possible scheduling points from which EDUCAT may be invoked. The standard variable typing statements used throughout MASH also apply here.

```
      IMPLICIT INTEGER (A-W)
      IMPLICIT REAL (X-Z)
      REAL RDM
```

Several standard COMMON blocks are included that contain global variables for all operating characteristic object modules.

```
      COMMON /PROTOl/ BUFFER(81),INSTR(16),0UTSTR(16),
     1                QMARK, NFEEDS
      COMMON /SIMULP/ IN, FN, PN, SPOPID
      COMMON /SIMCON/ IA, FA, PA, NY, IY, BY, CY, PS,
     1                IB, IC, NF, NP
      COMMON /POPWGT/ XWGT
```

The variable XWGT contains the weight implicitly assigned to each person in the micropopulation. It is computed as the inverse of the implicit sampling frequency when the micropopulation is created and is adjusted yearly to account for the effect of net immigration.

The principal data array containing the probabilities of remaining in the educational system are stored in array XPR. It has 3 dimensions, race, sex and grade.

```
      DIMENSION  XPR(2,2,18)

      DATA      XPR           /
     1           0.99,     0.98,     0.995,     0.99,
     2           0.99,     0.98,     0.995,     0.99,
     3           0.99,     0.98,     0.995,     0.99,
     4           0.99,     0.98,     0.995,     0.99,
     5           0.99,     0.98,     0.995,     0.99,
     6           0.99,     0.98,     0.995,     0.99,
     7           0.99,     0.98,     0.995,     0.99,
     8           0.99,     0.98,     0.995,     0.99,
     1           0.98,     0.95,     0.99,      0.98,
     2           0.99,     0.95,     0.995,     0.99,
     3           0.99,     0.95,     0.995,     0.99,
     4           0.95,     0.90,     0.90,      0.90,
     1           0.50,     0.50,     0.30,      0.20,
     2           0.80,     0.90,     0.90,      0.70,
     3           0.80,     0.90,     0.90,      0.70,
     4           0.80,     0.90,     0.90,      0.70,
     5           0.90,     0.95,     0.95,      0.90,
     6           0.90,     0.95,     0.95,      0.90  /
```

The array ADV will be used to count the number of advances into (and therefore through) each grade.

```
      DIMENSION  ADV(18)
```

The use of attributes by this operating characteristic could be specified in rigidly formatted comment lines. These lines could then be read by a program for building a structural and consistency matrix for the micro submodel, as suggested earlier in this chapter.

```
C====MICROPOPULATION ATTRIBUTES USED BY EDUCAT
C====AGE(6), ENTRY-POINT 6
C====RACE(6), ENTRY-POINT 6
C====SEX(6), ENTRY-POINT 6
C====CPSAREA(6), ENTRY-POINT 6
C====GRADCOM(3), ENTRY-POINT 6
C====EDSTAT(3), ENTRY-POINT 6
C====PERSEED(4), ENTRY-POINT 6
C====END MICROPOPULATION ATTRIBUTES USED BY EDUCAT
```

The number in parenthesis following the attribute name would correspond to the value to be used in the matrix definition. The number following ENTRY POINT would be the number of the scheduling point at which the operating characteristic is invoked when the attribute is referenced.

The first executed statement of the program tests an option to determine whether to execute this operating characteristic or not. Option 101 is keyed to the ALLOW EDUCAT and SUPPRESS EDUCAT commands. If the option is on, EDUCAT is suppressed.

```
      IF (OPTION (101))  RETURN
```

The next executable statement of the program module separates the flow of control into the separate calls to EDUCAT from scheduling points 1, 3, 6 and 9. No other calls are required to implement the operating characteristic. If other calls are detected, they result in transfer of control to an error procedure.

```
      GO TO (100, 5, 300, 5, 5, 600, 5, 5, 900, 5, 5), POINT
5     GO TO errorprocedure
```

At the beginning of a simulation exercise, it is only necessary to initialize to zero the count for the expected and actual number of advances in grade. The ratio of these counts will be used to adjust the probabilities used to impute advances in grades so that the actual number imputed will track the expected number.

```
100   XEXP = 0.0
      XACT = 0.0
      RETURN
```

At the beginning of the simulation pass (or year) in which the EDUCAT object module is invoked, the counts of expected and actual advances in grade must be initialized to zero. The attendance in grade array, ADV(18), must also be set to zero. As persons are processed, ADV(I) will contain the number of persons who have entered (and therefore completed) grade I in the current simulated year.

```
300   XE = 0.0
      XA = 0.0
      DO 310  I=1,18
310   ADV(I) = 0
```

The tracking factor for this year, XT, is computed as the ratio of the cumulative actual advances to the cumulative expected number of advances. At the beginning of the simulation, the expected number will be zero and the tracking factor is initially set to 1.0.

```
      IF (XEXP .EQ.  0.0)       XT = 1.0
      IF (XEXP .GT.  0.0)       XT = XACT/XEXP
      RETURN
```

The majority of processing is performed at the person level. First, the educational status of the person is checked. If the person has already left the school system, no further processing is required. The address of the current person, PA, is used to retrieve all attributes.

```
      E = PERS ('EDSTAT', PA)
      IF (E .EQ. 3)    RETURN
```

Otherwise, the person is either in school or has not yet entered the educational system. These alternatives are processed separately.

```
      IF (E  .EQ. 2)  GO TO 625
```

Age is the only attribute that determines when a person enters school. The following statement obtains the age of the current person.

```
      AGE = PERS ('AGE', PA)
```

If the person is less than 5 years old, no change in status is made.

```
      IF (AGE .LT. 5)   RETURN
```

Otherwise, entry into the school system is determined stochastically according to the probabilities specified in the source version of the operating characteristic. To impute a stochastic event, the person's current

random number seed, which is an attribute, must be fetched and a new pseudo-random number generated using generator RDM.

```
RN = PERS ('PERSEED', PA)
XRN = RDM (RN)
CALL  STPER ('PERSEED', PA, RN)
```

Initial enrollment is a function of age and the random number drawn.

```
IF (AGE .EQ.5 .AND. XRN .LE. 0.4) GO TO 605
IF (AGE .EQ.6 .AND. XRN .LE 0.9) GO TO 605
IF (AGE .GE.7) GO TO 605
RETURN [22]
```

Enrollment in the system implies a change in educational status and entry into and completion of grade 1. Tally the new enrollment also.

```
605   CALL STPER ('EDSTAT', PA, 2)
      GC  = 1
      ADV(l) = ADV(l) + 1
      ADVANC = .TRUE.
      GO TO 675
```

For persons already in school, continued attendance is dependent upon the person's race and sex and for grades 1-8, the region of the person's residence.

```
625   RACE = PERS ('RACE', PA)
      IF (RACE .EQ. 3) RACE = 2
      SEX = PERS ('SEX', PA)
      GC = PERS ('GRADCOM', PA)
```

Continued attendance is also determined stochastically, so that a random number must be extracted from the person's random number stream.

```
RN = PERS ('PERSEED', PA)
XRN = RDM (RN)
CALL STPER ('PERSEED', PA)
```

The policy experiment options are now applied. They are implemented by artificially changing the local variables defining race and sex of the current person within this operating characteristic. If option 201 is selected, then women are given the same probability of remaining in the educational system as men.

---

[22] The Fortran IV source code used in this example is meant to illustrate the translation of the source specification of an operating characteristic into its object form. The code presented here is not optimized either for minimum storage or for minimum execution time, but is chosen to illustrate the result of the translation process. Nor is the code presented well structured, since pedagogy is more important than style in this explanation.

```
      IF (.NOT. OPTION (201))    GO TO 630
      SEX = 1
      GO TO 650
```

Option 202, which is mutually exclusive from option 201, gives every person the same probability of remaining in the educational system as white men.

```
630   IF (.NOT. OPTION (202))  GO TO 650
      RACE = 1
      SEX = 1
```

Next, retrieve the probability of continuing in school. If the last grade completed is 7 or less, then retrieve the person's region of residence (an interview unit attribute) and adjust the probability to account for the assumption that persons in the west census region are only half as likely to drop out.

```
650   XP = XPR(RACE, SEX, GC)
      IF (GC .LE.7 .AND. INTUN('CPSAREA', IA) .EQ. 4)
     1      XP = XP + (1.0 - XP)/2.0
```

Now compare the random number drawn, adjusted by the simulation tracking factor, with the probability of staying in school and advancing a grade. If the person does not advance a grade change his or her educational status to reflect this.

```
      IF (XRN*XT .LE. XP)  GO TO 660
      CALL STPER ('EDSTAT', PA, 3)
      ADVANC = .FALSE.
      GO TO 670
```

If the person does advance a grade, alter his or her "grade completed" attribute to reflect the new grade. If the person has completed 18 grades, remove him or her from the educational system. Tally the enrollment in the revised grade.

```
660   GC = MINO(GC + 1, 18)
      IF (GC .EQ. 18) CALL STPER ('EDSTAT', PA, 3)
      CALL STPER ('GRADCOM', PA, GC)
      ADVANC = .TRUE.
      ADV(GC) = ADV(GC) + 1
```

For "successful" events, i.e. for those persons who advanced one grade, increment their count.

```
      XA = XA + 1.0
```

Increment the total of expected number of advances by the probability selected.

```
670   XE = XE + XP
```

The event has now been imputed or not, and all statistics and partial sums have been updated. The last set of options determines whether or not micro entity information is typed displaying attribute and variable values involved in this event. Options 401, 402 and 403 are keyed to the WATCH ALL, WATCH SUCCESSFUL, and WATCH THWARTED commands for this operating characteristic.

```
675   IF (OPTION (401))  GO TO 680
      IF (ADVANC .AND. OPTION (402))  GO TO 680
      IF (.NOT. ADVANC .AND. OPTION(403))   GO TO 680
      RETURN
```

The output string to be typed is encoded in the array OUTSTR, and the MASH system subroutine TYPE transmits it to the user's terminal.

```
680   ENCODE (80, 690, OUTSTR) PN, PA, RACE, SEX,
     1                            GC, XP, XT, XRN
690   FORMAT ('SCHOOLING', 5I6, 3F7.4)
      CALL TYPE
      RETURN
```

At the end of the simulation pass (or year) in which the EDUCAT object module is invoked, the expected and actual advance in grade counts for the year are added into the partial sum for the simulation exercise.

```
900   XEXP = XEXP + XE
      XACT = XACT + XA
```

Attendance is then computed for grade school levels, high school levels, college levels and post graduate levels. Micropopulation attendance is weighted by the current population weight and 4 aggregate time series are augmented. Error checking and recovery are not included for simplicity of presentation.

```
      Al = 0
      DO  910  I = 1, 8
910   Al = Al + ADV(I)

      A2 = 0
      DO 920  I = 9, 12
920   A2 = A2 + ADV(I)

      A3 = 0
      DO 930  I = 13, 16
930   A3 = A3 + ADV(I)

      A4 = ADV(17) + ADV(18)

      CALL PUTVAL (0, 'ELEMATT', CY, STAT, Al*XWGT)
      CALL PUTVAL (0, 'HIGHATT', CY, STAT, A2*XWGT)
      CALL PUTVAL (0, 'COLLATT', CY, STAT, A3*XWGT)
      CALL PUTVAL (0, 'UNIVATT', CY, STAT, A4*XWGT)
      RETURN
```

This completes the Fortran IV specification of the object form of the operating characteristic. The statement:

```
      END
```

signals the end of the program module.

Stochastic Components

Modelling of stochastic processes is an essential part of the translation from source model specification to object program computer module. Stochastic processes are an integral part of many of the microanalytic submodel operating characteristics. In addition, initial populations may be selected stochastically from a micropopulation survey file and surveys of a micropopulation may be selected using a stochastic component. Equations contained in an aggregate submodel may also have a stochastic "disturbance" term.

Simulation of stochastic events is accomplished through the use of psuedo-random number generators which yield a stream of random numbers having properties that one would anticipate observing of a random process. These properties are: (1) the distribution of a large sequence of random numbers approximates a desired distribution, generally uniform in the interval (0, 1); (2) the serial correlation of the sequence differs insignificantly from zero; and (3) the periodicity of the pseudo-random sequence is large relative to the expected number of numbers to be used.

The pseudorandom number generator initially chosen for the simulation of stochastic events was the one originally used by Greenberger in previous microanalytic simulation work [O4]. It generates the sequence $\{u_i\}$ using $u_o = 0.501695437$ and

$$u_{i+1} = (2^{18} + 3)u_i \qquad (\text{mod } 2^{35})$$

This generator has a known period of 2^{35}, and has the advantage that only a few computer cycles are required to compute successive numbers in the pseudo-random sequence. However, due to irregularities discovered in the generator and the decreasing cost of computation, this generator will soon be replaced with another having more rigidly acceptable statistical properties. [23]

---

[23] Unfortunately this generator has one annoying property not detected by Greenberger in 1961. The pseudo-random number sequence exhibits substantial second order autocorrelation, which is vividly displayed graphically in [G6]. Applying a X^2 test to a contingency table formed by the first 100,000 pairs $\{u_i, u_{i+2}\}$, the hypothesis of independence was rejected conclusively; the value of X^2 was approximately 38 standard deviations away from zero.

The stochastic functions are implemented in the following manner. When a micropopulation is created, each entity within it is given an initial "seed" for random number generation. These seeds are stored as attributes, presently named IUSEED, FAMSEED and PERSEED. To draw a pseudo-random number and save the seed for further drawings, the following skeleton of statements is required, for example, at the family or intermediate hierarchical level:

```
ISEED = FAMLY ('FAMSEED', FA)
.....................
XRAND = RDM (ISEED)
.....................
CALL STFAM ('FAMSEED', FA, ISEED)
```

Once a microentity has been given a pseudo-random number seed, that seed is used to generate a stream of numbers that is unique to that entity. [24] The seeds are themselves drawn from the original stream but quite far apart, so that the streams generated by each assigned seed are for all practical purposes independent.

Operating characteristics that are used to initialize the initial population may perform stochastic calculations. Each such characteristic has its own independent generator. Thus, the addition to or modification of the set of initializing characteristics will not affect the random numbers generated when the population is advanced through simulated time. These random number streams may be controlled to some extent by the user to provide alternative initial imputations. [25] Thus the effect of stochastic variability of the initial imputations may be separated from other sources of variability and measured. A new MASH command, DISPENSE NEW SEEDS, can be used to dispense a different and independent set of seeds to entities in the micropopulation, so that the effect of seed assignments can also be measured.

Other stochastic processes, such as sampling microentities for inclusion in a population or for export in a survey, and for computing disturbance components in an aggregate submodel, each have their own pseudo-random number seed and sequence. Several MASH commands have been extended using a phrase

---

[24] The credit for this improved treatment for stochastic components goes to Guy Orcutt. It.was implemented by Kenneth Jacobs. A previous version of MASH contained a simpler treatment of dispensing random numbers, in which one generator dispensed numbers for all entities. This process led to unnecessary and avoidable variability in the results. Any model change that caused any change in the order of assignment of random numbers to entities would precipitate an entire series of Monte Carlo changes over the entire population. This variance was due solely to the single pseudo-random number generator and weakened the sharpness of the model solution if it was not altogether submerged. The present treatment, involving individual seeds, avoids this problem.

[25] An ADVANCE command has been implemented whereby one can command: ADVANCE n RANDOM NUMBERS FOR INITIALIZATION OF LABOR, ... This causes n numbers to be discarded and for any moderately large n the random number streams will be independent and therefore the results of the alternate applications of stochastic processes will be independent.

"ADVANCING n RANDOM NUMBERS" so that independent solutions of these subsystems can be derived and the Monte Carlo variability measured. [26]

Aggregate Object Model Specification and Linkages to Micro Submodel

The specification of aggregate object models within MASH is a straightforward task and is performed totally at the MASH command language level. In part, this simplicity of specification is due to the simplicity of structure of aggregate models relative to microanalytic models. Furthermore, the importance of aggregate models for policy modelling has stimulated the development of a variety of computer based systems for their specification and solution, such as CANDIDE [M4], SIMULATE II [H3], and TROLL [M1]. The latter system has served as a model for part of the aggregate submodel specification and solution within MASH.

The class of aggregate models that can be easily implemented within MASH is limited to recursive models. Block recursiveness is permitted provided the blocks are explicitly recognized by the modeller and the iterative process for solving the block is made an explicit part of the model. Ordering the equations into proper recursive order is currently the responsibility of the modeller.

The MASH system is more limited and more primitive than other systems such as TROLL in a number of ways. First, only recursive models are allowed. Second, explicit ordering and iteration instructions are required of the user. Third, execution of the solution algorithm is currently interpretive and therefore more inefficient in the use of machine resources than a compiler based solution algorithm. The justification for these limitations is: (1) the aggregate models expected to be implemented within MASH for mixed micro-macro simulations are expected to be approximately 10-20 equations long and their execution time is expected to be dominated by the execution time of the microanalytic submodel; and (2) the most important aspect of the aggregate submodel system is that the user interface be easy and flexible to manipulate.

Macroeconomic submodels implemented within MASH may therefore consist of a system of recursive equations which may include stochastic terms. No restriction is placed upon the form of the right hand side of any equation. These models are annual. Endogenous variables, lagged exogenous variables, and stochastic terms may be used to compute the predicted values for the endogenous variables in the model.

---

[26] Credit for the improved modularity of treatment of stochastic functions as well as the design and implementation of the new commands and command extensions necessary for its support should go to Kenneth Jacobs.

Macro model equations are defined using DEFINE EQUATION commands. Each equation is identified by its name and owner. Macromodels are then defined as ordered lists of equation names with the DEFINE MACROMODEL command, and they are initialized for execution with the PREPARE MACROMODEL command. Equations may contain symbolic parameters, and lists of parameters may be searched in any specified order with the SEARCH PARAMETER LIST command. Variables in the model are dynamically linked to aggregate series in the annual time series data bank by means of ASSOCIATE and TRACK commands. Examples of these commands are given in Chapter 2.

The time series data bank provides the data linkage between micro and macro submodels. Operating characteristics in the micro submodel can access any series in the data bank for retrieval of aggregate time series data, and they may generate by aggregation new values to be stored in the current user's time series collection. Similarly, the macro submodel may retrieve values of required exogenous and lagged endogenous variables from any series in the data bank, and may record the time series values it generates in the bank. Thus, the data banks serves both as a dynamic bidirectional information interface between the submodels and as a storage device for recording the results of simulation exercises.

Macro submodel execution and micro submodel "passes" are interleaved in the simulation agenda and therefore in the computational sequence. A mixed macro-micro model may consist of a separate macro submodel after any or every micro submodel pass. The interleaved structure therefore provides an elementary capacity for iteration between micro submodels and aggregate relationships. The partial or general equilibrium implications of simulation results obtained by such a process depend upon the nature and extent of the interaction among modules in the macro and micro submodels; indeed, the specification of such interaction is a part of the model specification process. The simulation system allows certain interactions to occur, but does not require that any interaction be specified explicitly. Nor does the system either require that the solutions to submodels be consistent or guarantee that consistent solutions are produced.

Conclusion

The implementation of models using the MASH system has proceeded along rather conventional lines. Certain features, such as referencing attributes by name and use of functions referencing virtual arrays for data management, simplify the process of micromodel specification. However the complexity of micromodel specification defies easy treatment using any existing computer oriented tools known to the author. Macromodel specification, in contrast, is relatively straightforward. The lack of knowledge of basic and powerful methods of expressing microanalytic operating characteristics succinctly and the absence of

good micromodel management tools are fundamental barriers to significant progress in specifying such models.