FUNDAMENTAL ISSUES IN COMPUTER DESIGN AND IMPLEMENTATION FOR THE SOCIOECONOMIC MICROSIMULATION COMMUNITY

George Sadowsky, New York University, and Paul Cotton, Fulcrum Technologies, Inc.[1]
George Sadowsky, 251 Mercer Street, New York, NY 10012-1185 (sadowsky@nyu.edu)

KEY WORDS: Computer, Microanalytic, Simulation, Tax and Transfer Programs, Policy Evaluation, Hardware, Software Architecture.

1. Introduction[2]

Since the early 1960’s, microanalytic simulation models have provided an important methodology for assessing the impact of economic and social policy alternatives for a number of important areas in public policy. In particular, the analysis of tax and transfer systems applied to families and individuals now depends critically upon the construction and evolution of such models, and their use is routine and even expected in public agencies and research organizations dealing with these issues.

The term microsimulation model describes in this context a microanalytic model of some universe of elements which is solved under a variety of conditions using computer based simulation techniques. In the context of social and demographic analysis, the microanalytic units of such models are individuals and small groupings of individuals such as families, households, or tax filing units.

The pathbreaking work creating the field of socioeconomic microsimulation, *Microanalysis of Socioeconomic Systems: A Simulation Study*, was performed by Guy Orcutt and his colleagues in the late 1950’s. The underlying behavioral model was dynamic and stochastic, and the simulation system was implemented in assembly language on an IBM 704 computer system.

Initial public policy analysis based upon this new methodology was first applied to the federal individual income tax system in the United States [Pechman 1965] and Canada [Bossons 1967]. Their original models were strictly static accounting models that embodied neither behavioral assumptions nor forward projections in time. Additional Canadian models were subsequently built using similar methodology by Bossons for tax reform analysis for the Province of Ontario (1973) and by Fortin for the Province of Quebec (1981).

Another initial use of microanalytic simulation methodology was to project the economic status of the retired aged [Schulz 1968]. The underlying microanalytic model was long term, dynamic and stochastic, and emphasized labor market participation and accrual of private and public pension rights.

Interest in the late 1960's in the areas of welfare reform and negative income tax proposals led to the creation of the RIM model [Wilensky 1970] for use by the President's Commission in Income Maintenance in the United States. RIM embodied a a static model used to project the effects of alternative tax and transfer policies upon families in the United States. The success of RIM in supporting the work of the Commission led to the development first of TRIM in 1971 and then of TRIM2 in 1979 to support continued exploration of tax and transfer policy alternatives focused upon the lower end of the income distribution.

At the same time in the early 1970’s, another approach was undertaken under the leadership of Guy Orcutt to develop DYNASIM, a dynamic microanalytic simulation model embodying expanded household sector submodels [Orcutt 1976]. The initial underlying computer system, MASH, was written in Fortran for a DECsystem-10 [Sadowsky 1977]; a later implementation, MASS, was created at Yale University by Amihai Glazer and his colleagues in PL/I for an IBM System 370. In 1981 the DYNASIM model was reimplemented as DYNASIM-2 for reasons of efficiency and increased portability. Developments in the area of dynamic socioeconomic simulation models have been continued recently by Steven Caldwell at Cornell University, a member of the original DYNASIM research group.

The development of TRIM2 spawned several other microeconomic modelling developments. They included MATH [Doyle 1979] developed by Mathematica Policy Research, Inc.; KGB [Betsen 1980] developed within the Department of Health and Human Services, and HITSM, a proprietary model used by, *inter alia*, ASPE (the Office of the *A*ssistant *S*ecretary for *P*olicy and *E*valuation) in 1987.

In 1986, Statistics Canada initiated its own microanalytic simulation model, SPSD/M [Statistics Canada 1989], a microanalytic model of the Canadian household sector, implemented on an MS-DOS based microcomputer platform. This choice of hardware platform was a major departure from previous choices for such applications. The SPSD/M development contains a static model oriented toward assessment of the revenue and distributional effects of Canadian household tax and transfer policies.

Microsimulation models may be either single or multiple period models. While such models implicitly have an element of future prediction in them, this aspect is not necessary to the concept of such modelling. When such models have neither behavioral change nor future projection elements, they are sometimes referred to as *static accounting models*. However, the use of such models in a public policy setting is oriented toward providing useful estimates of the future impact of alternative programs and legislation. Such a process, however implemented, is often referred to as *aging the population* so that it will be representative of future years and will take into account a predicted pattern of social, demographic, and economic changes.[3]

Microsimulation models are often categorized as static or dynamic with respect to the method that they employ to predict future outcomes. *Static models* are characterized by lack of direct interaction of microanalytic units within the context of the model during the time period simulated. Static models may be either deterministic or stochastic. Dynamic models are characterized by varying degrees of direct interaction between micro population units within the simulation process. Such interaction includes the birth, death and recombination of micropopulation units in a manner intended to simulate accurately those processes in the entire population. Dynamic microanalytic models rely upon an accurate knowledge of the dynamics of such interactions.[4]

There is general acceptance that dynamic models provide a more realistic representation of micro population unit behavior. However, static models are regarded as more effective at times for specific short run projection purposes because of their greater simplicity and the often lower costs associated with building such models and obtaining computer generated model solutions.

2. The National Academy of Sciences Study

One of the grand challenges facing the United States is applying appropriate quantitative techniques and computational science in formulating and administering effective economic and social policy in a wide variety of areas. Meeting these challenges is important for the economic and social health, if not the viability, of our nation. While it is difficult to obtain a precise measure of the opportunity cost of *not* providing better such instrumentation for policy formulation, surely such a cost is there and could be very large and multidimensional.

Although many of the nation's problems are not easily amenable to computer analysis, a subset of them dealing with individual taxation and transfer policies have benefitted from it in the past. Microsimulation models have been used consistently since 1963 to provide estimates of the revenue and distributional impact of a wide variety of tax and transfer proposals. Such models have been used in the private sector, by social scientists doing research, by administrative departments and by agencies of the Congress. During periods of active consideration of legislative initiatives in this area, use of such models has been intense and has pervaded the analytical process behind evaluation of proposed legislation.

The quality and effectiveness of models built to support the development and evaluation of these economic and social policies is of substantial importance. The direct cost of national involvement in the U.S. in the development, implementation and use of such models is easily in the millions of dollars. The indirect costs, including time spent analyzing and acting upon the results of such models, is much higher. The opportunity cost of not having effective models and tools for analysis is difficult to assess, but is likely to be largest of all using different metrics.

Realizing the importance of such models for policy formulation and evaluation in the United States, in 1989 the National Academy of Sciences convened a Panel to Evaluate Microsimulation Models for Social Welfare Programs. The panel concentrated primarily upon the substantive aspects of model formulation, since much of the challenge of building such models rests in incorporating within them credible administrative and behavioral content. Nevertheless, the Panel understood the importance of effective exploitation of computer technology for use of such models, and commissioned a study on the topic of the implications of emerging computer hardware and software technology for microsimulation models, which is being published as [Cotton and Sadowsky, 1991].

The study was commissioned in part because of a concern regarding the effectiveness of the current major model used by the Department of Health and Human Services, TRIM2, for supporting current work in transfer payment analysis. Despite the importance of TRIM2 and similar models, the computing environment in which they are embedded is relatively limited. They have generally been implemented as batch processing programs, which involve significant delays.[5] This is true for two major reasons:

1. For realistic experiments, microanalytic simulation populations need to be relatively large. Populations of 100,000 units are often required to obtain the desired amount of detail in the results.
2. The amount of computing power required to apply a complex micromodel's operating characteristics can be significant. Using traditional computer architectures, compute time for one time step of a model is proportional to the product of the number of population units and the number of instructions required per unit. Even when interaction during the course of the experiment is possible, it generally only adds modestly to the compute-intensive part of the simulation process.[6]

The batch model of computing was adopted early in the history of computing to maximize computer utilization, often at the expense of the productivity of the programmers and users of the time. It reflected a high cost of computing and a low cost of programmers and users on a relative basis. Early computer users adopted a variety of compensating strategies, such as working on several projects in parallel, in order to maintain their own productivity in the face of what was (in retrospect) at best a user-indifferent computing environment. The cost of working in such an environment was that the user would have to switch mental contexts between tasks, often with resulting loss of time and insight.

With the recent strong reversal of the factor costs of production, computers are now used more inefficiently in order to maximize the productivity of users. The batch processing paradigm has been relegated to an increasingly smaller segment of commercial computing, while interactive computing models with more elaborate and "friendly" user interfaces have become more numerous and popular.

However, no microanalytic simulation models have been implemented within a modern computing environment having a friendly user interface, and no existing models offer near real time feedback because of their substantial raw computing requirements.

Realizing that these concerns impact the policy formulation and analysis process negatively, the Panel wanted to know whether the current structure, mode of operation and user interface of TRIM2 should be modified and moved to a microcomputer platform, following the direction established by Statistics Canada with SPSD/M. The study concluded that:

1. TRIM2 should not be moved to a microcomputer environment. The investment would be sizeable without corresponding short term benefits. Medium term benefits might be negative in terms of preempting more efficient longer run strategies. Investment in TRIM2 should remain incremental and track current areas of policy evaluation interest.
2. Medium and long term investment in computing environments for microanalytic simulation activities should focus primarily on systems implemented on desktop hardware platforms.
3. An in-depth, medium term study should be initiated to define over the next 1-2 years a next generation computing environment for supporting microanalytic simulation modelling activities. The specification of such a system should be oriented to the objects used and actions taken by system users.
4. A part of the medium term next generation system specification study should be to assess how to move the capital embodied in the TRIM2 model program modules to the new environment in a verifiably functionally equivalent form.
5. The imprecise treatment of the notion of "aging" needs conceptual and definitional attention. A better conceptual framework would allow components of aging to be implemented in such a way that static, dynamic, and mixed models could be implemented within the same software system framework for performing microanalytic simulation experiments. The results should be part of the specification of the next generation software recommended above.

These recommendations put emphasis upon the future of TRIM2 because of the specific focus of the National Academy of Science's Panel's mandate. However, the recommendations have broader scope with respect to the future of microsimulation modelling activity and could be applied to much of the activity in the microsimulation community today.

3. The microsimulation community

Microanalytic simulation modelling activity has to date been relatively restricted to one general area of economic activity — tax and transfer policy — and has had a limited number of producers and direct consumers.

There are several reasons for this state. First, until now general microsimulation models have been relatively difficult to create and use. Second, access to data sets from which large initial populations could be extracted has required dealing with government agencies and investing substantial time in understanding the content and limitations of the data as well as extracting and reformatting the initial population — a potentially laborious task. Finally, until recently, microsimulation models have been implemented on large batch-oriented computers and have not been portable for the non-specialist.

The result is that the microsimulation community is limited largely to government agencies dealing with taxation and transfer payment systems, research organizations working with government agencies to analyze policy in these areas, and academic organizations and individuals having an interest in or connection to the methodology or the underlying substance. The barriers to entry into the microsimulation community have historically been sufficiently high to discourage more casual entry or exploration.

This is an unsatisfactory state of affairs. We believe that the limited number of users of such models is more a reflection of the degree of difficulty of dealing with most existing models than it is a reflection of underlying demand. If such modelling techniques and model uses are not sufficiently simple to be taught in the classroom during a one semester course, then the use of these techniques is likely to remain largely limited to present uses and users.

Several recent events and trends hold some promise for empowering non-traditional groups of users of this technology. They are:

- The development of microsimulation models on microcomputer platforms, which are commonly available and easier to work with than traditional mainframes; and

- The possibility of approaching such models based upon client-server architectures and linked via data networks, allowing implementation and use of remote models on specialized hardware platforms without incurring traditional historical penalties associated with remote computing.

Both of the above environments promise more rapid turnaround time for obtaining the results of simulation experiments. As turnaround time decreases, the importance of the ease of use of the interface increases, especially as new groups begin to use this methodology. As the speed of response increases, the user interface must become very intuitive and non-technical for a large subset of simulation experiments of interest if this methodology is to grow in use. We believe that suggestions made in the following sections regarding ease of use, portability, and rapidity of response are likely to satisfy considerably more demand than is currently observable.

It is intriguing to speculate what the effect of approaching near immediate response to microsimulation experiments would be on the behavior of policy makers, and what difference it would make to legislative and administrative functions. One could hope that with a friendly and intuitive interface such an environment could yield a fundamental and dramatic qualitative improvement in the process of exploring the policy space for those areas of social and economic policy that are currently being analyzed with the aid of microanalytic simulation models.[7]

4. Hardware issues

Microanalytic simulation models of non-trivial size or complexity have relied for solution upon the use of digital computers, and the ability to use these models in a practical manner has depended upon rapid technical progress in the computing industry. This progress has allowed the complexity of the microanalytic models to increase as well as the costs associated with a specific simulation experiment to decrease substantially over time.[8] The future of the implementation and use of such models depends upon the available computing environment evolving to support more complex models and new uses.

Technological progress in the computing industry has provided computing users with an apparent steady technical dividend on the order of 10% to 30% per year, depending upon the specific component and the stage of its evolution. Based upon current knowledge of the limitations imposed by physical laws, the size of the market, and knowledge of computer industry manufacturing techniques, there is no substantial evidence that this rate of technical progress will diminish in the next 10 years. An analysis of the literature indicates that the overall improvement in the price-performance ratio for computer technology has approached 30% over past years, with variations by component.[9]

This ratio has been remarkably stable over last 30 years. If undiminished, it implies that in less than 5 years there will exist products on the market having 100 MIPS (*M*illion *I*nstructions *P*er *S*econd) processing power, 64 MB of primary memory and 1 GB of disk storage available for about $5,000.[10]

This rate of price-performance improvement applies to computers built according to a von Neumann architecture, i.e. computers with one processor and one ambiguous memory storing a mixture of program instructions and data. Computers having this architecture are now often referred to as SISD (*S*ingle *I*nstruction stream, *S*ingle *D*ata stream) machines to distinguish them from other architectures that are now emerging as viable alternatives.

Genuine innovations or breakthroughs in computer architecture will render any rate of technical progress based upon classic von Neumann architecture too conservative. In particular, any substantial cost-performance breakthrough in the use of parallel processing for general purpose computing tasks will allow additional processing power to be added to a system at an incremental cost related to the cost of the processing chip and its interface rather than to the cost of the entire system, so that substantial gains in performance could then be achieved at more modest increases in price.

Gains in the productivity of computing equipment have been measured by the Department of Commerce starting in 1986, and are published as _______ in ____. The index is flawed in that it measures only productivity gains in IBM hardware(!!!). Nevertheless, in the past two years, the rate of productivity improvement has been XX% per year.

Advances in computing technology may be partitioned into three major categories: (1) processors, primary memory and secondary memory; (2) more general architectural considerations: and (3) software. Other categories, such as advances in computer architecture or in co-operative computer networking (itself a new form of architecture) are likely to have a major impact upon the productivity of both the general computing environment and the desktop environment and are addressed below.

Further evidence can be obtained from observation of the price performance characteristics of new products. In mid-1989, the cost per MIP[11] for a Unix workstation was over $1,000. In mid-1991, the cost per MIP is $400. In mid-1989, the cost per megabyte of random access memory on the margin was approaching $100; today it is about $40. In mid-1989, disks with 100 MB capacity cost about $1,000, while larger disks such as 600 MB models cost $3,500. In mid-1991 the prices of these disks were about $450 and $1,800 respectively.

This progress will continue in part because the production of both processors and primary memory is now an exacting photolithographic process. Technical advances in improving the density of memory elements, as well as increasing the yield of the production process, have steadily raised capacities per solid state chip as well as lowering the cost per unit.

Alternative architectures have long held promise for increasing processing throughput at a fraction of the cost that would be required by replicating the SISD architecture. Some of this promise is being realized; for example, vector processors can process multiple streams of data at a rate much faster than it would take to process a single stream multiple times.[12]. More recently, architectures containing multiple processors have become more common; such architectures generally support execution of truly simultaneous tasks in one computing system; see [Trew 1991] for details. Software operating systems such as Unix can mesh nicely with such architectures, since programs written in Unix can be structured to spawn subtasks across processors to accomplish their objective.

Static microanalytic simulation exercises, by virtue of there being no interaction between micro population units during a forward projection in time, are well suited to exploit a multiprocessor architecture, assuming that the overhead of disaggregating the task of model execution is not large and the system software allows disaggregation and aggregation of results to be performed efficiently. Processing of the micropopulation file can be decomposed into multiple independent threads, each of which process a subset of the original file. The cost of the decomposition is measured in terms of the creation of the independent processes and the aggregation of the results of each of the independent threads.

Computer networks for research and education are now becoming widespread and are now beginning to be organized to work cooperatively to solve specific tasks. Such cooperation may take several forms, such as specialization of function, in which two or more dissimilar computers are linked together so that each can perform that part of the processing task for which it enjoys a relative advantage.[13] Other forms of cooperation include separating resource usage among computers[14] as well as enlisting multiple computers to process parts of a task in parallel, much as the above scenario described multiple processors in the same system unit performing independent subtask threads into which a task had been decomposed.[15] One increasingly common form of exploiting networks is the implementation of tasks in a client-server mode, in which a client program, often running on a desktop system and having a graphical interface, is coupled with the major computational process executing on one or more independent server systems on the network.

Computer architectures other than SISD have been rare in the past, in part because of the difficulty in expressing algorithms in a form capable of exploiting effectively those other architectures. However, this situation is changing. As Gordon Bell noted in 1989:

> “The good news is that a vast array of new, highly parallel machines are becoming available and that autotasking compilers are evolving to take advantage of this in a transparent fashion to a limited degree. The bad news is that not all applications can be converted automatically. Users are not being trained to use such machines in an explicit fashion. No concerted effort is in place covering the spectrum from training to research. This will require a redirection of resources.”[16]

It seems evident that microanalytic simulation modelling activities are likely to be able to exploit, perhaps quite substantially, some of these new architectures. The extent to which their productivity will increase is difficult to determine because such an architectural shift implies a structural change in the way in which tasks are performed. However we believe that the beneficial effect of such new architectures will be quite substantial, even though we cannot now predict with any degree of certainty how they will be realized.

Important advances in performance and response time may be achievable by studying the use of different forms of computer architecture.

SISD architectures. Since the creation in 1945 of the ENIAC, the first programmable electronic digital computer, the vast majority of computing systems have been designed using a *von Neumann architecture*. This architecture is often referred to as SISD (*S*ingle *I*nstruction *S*ingle *D*ata stream) because it is characterized by one processor, executing a single stream of instructions sequentially and operating on a single stream of data.

While historically the most efficient SISD architectures have been large mainframes, this is not the case now. The most efficient SISD cost-performance available today and in the foreseeable future is provided by microcomputers and workstations. Very fast desktop workstations at reasonable prices are now a reality. Capable of very large primary memory at non-exorbitant costs, they promise very rapid execution of microanalytic simulation models completely in primary memory.[17]

In theory, if a computer system of this type were to run sufficiently quickly, calculation of the results of a microsimulation experiment would appear to be essentially instantaneous. Given the recent substantial improvement in processing speeds of SISD workstations. such architectures cannot be counted out for supporting microsimulation experiments.[18]

SIMD architectures. SIMD (*S*ingle *I*nstruction, *M*ultiple *D*ata stream) architectures are quite appealing for executing static microsimulation models. A SIMD system consists of a large number of individual processors, each with its own local memory, responding to a synchronized stream of instructions coming from a central point. For example, Thinking Machine's CM-2, (*C*onnection *M*achine) may contain up to 32,768 individual physical processors. Such highly parallel systems promise the possibility of parallel execution over all units in the simulation population very rapidly.

Connection Machines are now installed in NSF-sponsored supercomputer centers and in a variety of universities, as well as making some modest penetration into the private sector. CM systems can be programmed in Fortran and C, but the programmer must be aware of the peculiar structure of the system and adapt his or her programming methods accordingly.[19]

Such architectures are appealing because execution is truly simultaneous over all processors attached to the job. Further, the CM provides functions across all local data address spaces, so that operations such as aggregation of results are a fundamental and efficient part of the system structure.

The promise of SIMD machines for dynamic models is less certain. While individual processors can exchange messages, which could be individual population records, such operations are not easily parallelized and will add to the linearly executed part of the code and unduly lengthen execution time. SIMD architectures need to be tested to determine what their strong and weak points are in supporting microsimulation modelling.

MIMD architectures. MIMD (*M*ultiple *I*nstruction, *M*ultiple *D*ata stream) architectures are very appealing for microsimulation model implementation. For most model operations, simulation populations can be broken up into many small subpopulations and each subpopulation assigned to one of many processors. Results for the entire population would then be obtained by aggregating the local results obtained by each processor.

MIMD machines are beginning to emerge in the commercial world. A major difficulty for many MIMD architecture programmers is how to parallelize sequential code so that the efficiencies of multiple processors can be captured in increased throughput. In the general case, this is a very difficult problem upon which much effort is being spent. However for microsimulation model implementation the problem is considerably simpler, although probably at the cost of having to do manual processor allocation and management instead of it being done automatically by system software. Given an appropriate set of primitives for processor management, it should be possible to implement microsimulation models in a reasonably straightforward manner.

Components for MIMD computer systems have been available for some time. The INMOS transputer chip is built to run optimally in a multiprocessor configuration. Coprocessor boards have been announced for microcomputers that permit parts of a task to operate in a MIMD sub-environment manually. More recently the Santa Cruz Operation has announced an extension of SCO Unix for certain MIMD architectures. NCR has recently announced its first "mainframe computer," the NCR 3600, which is in actuality a MIMD system composed of 8 to 288 Intel 80486 processors.

The MIMD systems are a recent architectural event, and, like SIMD systems, do not yet have the mature software base that will ultimately be needed to support microsimulation modelling. However, they offer substantial promise for developments especially in the area of dynamic modelling where different micro units are likely to undergo different processing sequences and where the micropopulation is dynamic with new entrants, exits, and recombinations. The latter set of functions can be handled in a straightforward manner in a MIMD architecture, whereas it is likely to cause substantial complications in an SIMD environment.

With the exception of the SISD environment which is fairly well understood, time will be needed to understand how to program the other architectures to produce efficient code for executing microsimulation models. We believe that the state of hardware developments is sufficiently promising that initial investments to achieve such understanding should begin now.

4. Software Issues

In spite of the importance of microanalytic simulation as a tool in certain areas of public policy, software tools and implementations to support it have generally been specialized, parochial, not easy to to use, and lacking in software innovation. This situation can be partially explained by the project specific missions of organizations using the methodology and the lack of a commercial market for such systems. The result has been a splintering of scarce resources, production of incompatible systems and model implementations that will be difficult to move, and the lack of a common framework for development that could foster cumulative gains in this difficult area. The public interest does not appear to be well served by the direction of this development.

Specificity of model implementation. To our knowledge, the structure of the implementations of microanalytic simulation models to date have been specialized. While internally modular, there has been little in the way of object orientation or reusability of pieces outside of a specific modelling environment. As a result, code embedded in one implementation of a specific model is difficult to export to other environments. The general notion of subroutine, albeit in a more general complex context, has been missing. in part because of a lack of commonly accepted system software environment into which such modules could be introduced.

The construction kit paradigm has evolved in the microcomputing world as a powerful way in which to enlist non-technical users in the building and execution of models of many different kinds. Music Works, LabView, the Pinball Construction Set, Stella, and even the basic notion of the spreadsheet itself all typify the paradigm of programming by construction, using a fundamental set of objects abstracted directly from the discipline being addressed. It should be possible to construct a framework for microanalytic simulation that is also based upon programming by construction that supports the integration of model object modules by non-technical users and yet allows modules to be of sufficient complexity to represent real life operating characteristics.[20]

Lack of friendly, intuitive user interfaces. Almost all previous microanalytic simulation models have been implemented in a batch mode, with user interfaces that reflect batch mode processing practices. Within the last 10 years, the power and appeal of graphical user interfaces has become evident; they are more efficient, potentially considerably more intuitive, and have the capability of expanding the base of users of programs dramatically.

Knowledge, experience and tools are now available to produce effective interfaces for microsimulation models. User driven instrumentation that can be used both to define and steer such models can be implemented and, we believe, would increase both the ease of use and the overall applicability of microanalytic simulation as a relevant policy tool.

Windowing environments now exist both on a variety of individual machines and across network connections that can be used as a basis for constructing such interfaces. Sets of basic control devices, often called “widget sets,” are now becoming available for building instrumentation interfaces for specific programs with relative ease. Among these are Sun’s Open Look widget set, Ohio State’s ACE system, the Cornell Theory Center’s Scientist’s Workbench, and NeXT’s NeXTStep. The Department of Energy, NASA and the National Science Foundation now appear to be in the process of converging upon Stardent’s AVS package as appropriate software for defining high level interfaces for scientific visualization to be supported and used by their research communities.[21] Stardent’s AVS is now being ported to other platforms, including DEC systems, the IBM RS/6000 line, and Sun SparcStations. While these tool sets may not offer every feature desired, they represent a critical mass of tools that can and should be exploited.

Such a graphical user interface would provide more power if it contained mechanisms both for model implementation and execution. Using the construction kit approach discussed above, we believe that it is possible to design a system which would encompass both functions, and that could provide at least a partial unifying force that could allow the translation of model operating characteristics into computer modules that would be more cumulative, more chargeable, and more portable than currently exists today.

Static vs. Dynamic Model Schism. From the beginning of use of microsimulation models, there has been a divergence between static and dynamic model enthusiasts. For any given model, time horizon, and set of policy issues, there is some combination of static and dynamic features that will address the issue most adequately. As Devine and Wertheimer [1979] have argued well, almost all models constructed to that date contain both dynamic and static elements.

One of the issues contributing to the schism is the ease and cost effectiveness of implementation of dynamic elements in models. In the past, this has caused developers of some static models to adopt system structures that are highly resistant to the later introduction of any dynamic elements, even though such elements might be desirable for supporting revised policy directions or including new substantive knowledge. In this way, the schism hurts both groups.

We believe that a substantially new software environment for supporting micro model development and execution could be constructed in such a way that both static and significant dynamic elements could be incorporated within the one framework. If this can be accomplished, then model builders will be free to choose a mix of static and dynamic operating characteristics. They will be able to regard the methodologies as complementary rather than competitive, allowing for richer model development. The cost of achieving this gain is the cost of implementing a new software environment; such gains cannot be produced by investment on the margin in any existing system.

The key issue facing the microsimulation community with regard to new developments in the implementation of microanalytic models is one of paradigm orientation, not one of conventional software tools. Conventional tools, such as high level languages and development environments, exist in sufficient number and quantity to support model implementation on a sizeable number of hardware platforms, including increasingly newer SIMD and MIMD architectures.

5. Organizational Issues

The divergence of microanalytic simulation efforts is encouraged by the fact that such systems and their results are generally a technical intermediate product within a mission oriented agency. There is a tendency in such situations to treat investment in such systems as marginal and to continue to invest on the margin as requirements evolve. The absence of a commercial market for such systems means that there are no visible alternatives that can be purchased rather than constructed in-house.

No clear leader in the implementation of environments or standards for implementing such models has emerged. At present there are a number of *de facto* competitors in North America with different objectives and orientations. The absence of any unifying force that is perceived to add value to the work of current suppliers of such systems will encourage the perpetuation of the NIH (*N*ot *I*nvented *H*ere) syndrome, leading to continued multiple efforts in a field underendowed with resources to begin with.

We have no easy solution to this problem, except to note that agreement that the organizational dimension of current microanalytic modelling efforts is inefficient may provide a beginning for an eventual solution. If the current pattern of multiple developers investing only on the margin is perpetuated, such a solution will not be achieved.

6. Conclusion

Microsimulation modelling has proved itself to be an essential tool for addressing a variety of social and economic policy issues. Microeconomic models have been used for this purpose for almost 30 years now, with relatively localized development of such models in a variety of organizations for policy evaluation and research. Computer systems to support such simulation activities have been generally specific and parochial, and their commercialization has not been regarded as feasible.

The microsimulation community is to some extent at a crossroads in terms of implementation of new systems for microsimulation. Existing systems address localized needs moderately well, but are difficult to use. The opportunity cost of not making a coordinated investment in a new model support environment grows each year as technical progress in both hardware and software produce an increasingly rich set of interesting alternatives that existing systems cannot exploit.

We believe that microanalytic simulation models can soon be built that

- can increase significantly the effectiveness of developing and modifying models, as well as providing a framework for their possible cumulative use;
- have friendly, easy to use graphically oriented front ends for both model construction and development that could potentially increase the population of microsimulation model users very substantially;
- provide the possibility of integrated static-dynamic models with the two methodologies contributing cooperatively to model development in the same general model support environment; and
- execute very quickly, perhaps almost immediately, probably using new computer architectures not yet applied to socioeconomic microsimulation, with potential major improvements in the policy formulation and evaluation process.

We believe that this offers the microanalytic simulation community a great opportunity.

The opportunity costs of not exploring this space of potentially rich and useful alternatives are growing. We believe that the time to start the exploration is now. To the extent that there are impediments to starting, they are primarily organizational, partly financial, and to a very limited extent technical.

BIBLIOGRAPHY

Bell, Gordon, “The Future of High Performance Computers in Science and Engineering,” *Communications of the ACM*, Vol. 32, No. 9, pp. 1091-1101.

Betson, D., Greenberg, D. and Kasten, R., “A Microsimulation Model for Analyzing Alternative Welfare Reform Proposals: An Application to the Program for Better Jobs and Income.” *Microeconomic Simulation Models for Public Policy Analysis*, (ed by Havemann, R. and Hollenbeck, K.) New York: The Academic Press, 1980.

Bossons, J., *Studies of the Royal Commission on Taxation, Number 25: A General Income Tax Analyzer.* Ottawa, Ontario: Queen's Printer, 1967.

Cartwright, David W. and Smith, Scott D., “Deflators for Purchases of Computers in GNP: Revised and Extended Estimates, 1983-88,” *Survey of Current Business*, Bureau of Economic Analysis, U.S. Department of Commerce, November 1988, pp. 22-23.

Citro, Constance F., and Hanushek, Eric A., eds., *Improving Information for Social Policy Decisions: The Uses of Microsimulation Modeling. Volume 1: Review and Recommendations*. (Report of the Panel to Evaluate Microsimulation Models for Social Welfare Programs.) Washington, D.C.: National Academy Press, 1991.

Cole, R., Chen, Y.C., Barquin-Stolleman, J.A., Dulberger, E., Helvacian, N. and Hodge, J.H., “Quality-Adjusted Price Indexes for Computer Processors and Selected Peripheral Equipment,” *Survey of Current Business*, Bureau of Economic Analysis, U.S. Department of Commerce, January 1986, pp. 41-49.

Cotton, P. and Sadowsky, G., "Implications of Emerging Computer Technology for Microsimulation Modelling," National Academy of Sciences [Panel to Evaluate Microsimulation Models for Social Welfare Programs], to be published spring 1991.

Cotton, P., "Options for Creating a Portable Modelling System for Social Policy Simulation Database and Model," Internal Report, Ottawa, Ontario: Social and Economics Studies Division, Statistics Canada, October, 1986.

Devine, Judith and Wertheimer, Richard, *Aging Techniques Used by the Major Microsimulation Models.* Working Paper 1453-01, The Urban Institute, Washington, April 1981.

Flamm, Kenneth, *Targeting the Computer*, The Brookings Institution, Washington, D.C., 1987.

Guthrie, H. W., Orcutt, G. H., Caldwell, S., Peabody, G. E., and Sadowsky, G., "Microanalytic Simulation of Household Behavior," *Annals of Economic and Social Measurement*, Vol. 1, April 1972, pp. 141-169.

Johnson, J., Roberts, T.L., Verplank, W., Smith, D.C., Irby, C.H., and Beard, M., “The Xerox Star: A Retrospective.” *Computer*, Vol. 22, No. 9, pp. 11-29.

Lewis, G. H. and Michel, R. C., ed., *Microsimulation Techniques for Tax and Transfer Analysis*, Washington, D.C.: The Urban Institute Press, 1990.

Okner, B. A., *Income Distribution and the Federal Income Tax*, Ann Arbor: Institute of Public Administration, The University of Michigan, 1966.

Orcutt, G. H., Greenberger, M., Korbel, J., and Rivlin, A. M., *Microanalysis of Socioeconomic Systems: A Simulation Study*, New York: Harper and Brothers, 1961.

Orcutt, G. H., Caldwell, S., Wertheimer, R., Franklin, S., Hendricks, G., Peabody, G., Smith, J., and Zedlewski, S., *Policy Exploration Through Microanalytic Simulation*, Washington, D.C.: The Urban Institute, 1976.

Orcutt, G. H., Merz, J., and Quinke, H., *Microanalytic Simulation Models to Support Social and Financial Policy*, Amsterdam: North-Holland, 1986.

Pechman, J. A., "A New Tax Model for Revenue Estimating," *Government Finance and Economic Development* (ed. by A. T. Peacock and G. Hauser), Paris: Organization for Economic Cooperation and Development, 1965.

Pechman, J. A., Okner, B. A., and Munnell, A., "Simulation of the Carter Commission Tax Proposals for the United States," *National Tax Journal*, Vol. 22, March 1969, pp. 2-23.

Sadowsky, George, *MASH: A Computer System for Microanalytic Simulation for Policy Exploration*, Washington, D.C.: The Urban Institute, 1977, 300 pp.

Sadowsky, G., "MASH: An Online System for Socioeconomic Microsimulation of the U. S. Household Sector," *Online 72 Conference Proceedings.* Brunel University, England: Online Computer Systems, Ltd., 1972.

Schulz, J. H., *The Economic Status of the Retired Aged in 1980: Simulation Projections.* Social Security Administration Research Report No. 24.), Washington, D.C.: Government Printing Office, 1968.

Statistics Canada, *SPSD/M Product Overview*. Ottawa, Ontario: January, 1989.

Trew, Arthur, and Wilson, Greg, eds., *Past, Present, Parallel: A Survey of Available Parallel Computing Systems*. Berlin: Springer-Verlag, 1991.

Wilensky, Gail R., “An Income Transfer Computational Model,” The President’s Commission on Income Maintenance Programs: Technical Studies. Washington, D.C.: Government Printing Office, 1970.

Wyscarver, Roy A., The Treasury Personal Individual Income Tax Simulation Model. OTA Paper 32, Washington, D.C.: Office of Tax Analysis, U.S. Department of the Treasury, July 1978.

FOOTNOTES

---

[1] The authors are, respectively, Director of the Academic Computing Facility at New York University and Senior Technical Adviser at Fulcrum Technologies, Inc. in Ottawa, Canada. This paper represents the opinion of the authors and does not necessarily represent the official position of New York University or Fulcrum Technologies, Inc. The authors would like to express their gratitude to Faye Duchin and Michael Wolfson for helpful conversations and advice.

[2] Some of the material in this introductory section has been largely abstracted from the introductory section of [Cotton and Sadowsky, 1991].

[3] Aging a population can have several dimensions. Demographic aging may consist of applying rules for modifying population weights over time; economic aging implies modifying a set of economic variables over time. Both sets of rules apply to individual units in the initial micro population. The application of these rules during the progress of the simulation exercise is performed so that key aggregates produced will match control totals that have been defined using methods independent of the simulation. This process may be thought of as a complex normalization process.

[4] The class of microanalytic simulation models implemented in this field is a relatively specialized subset of the class of all such models. In particular, models implemented almost all are single sector, single population, and fixed

time step simulations, with only a minor geographic component if any.

[5] Using the Treasury's Tax Analysis Model in 1963, users could expect a delay of about 1 to 2 hours from the submission of a simulation run to delivery of output from it. This response time has not changed dramatically for such systems implemented on mainframe batch environments in the last 28 years. The performance of Statistics Canada's SPSD/M system, recently implemented on a microcomputer, has been somewhat better with a response time of about 20 minutes, with the possibility of prematurely terminating the run at any time and obtaining results from a randomized sample of the initial population for simulation.

[6] Both MASH [Sadowsky, 1977] and SPSD/M [Statistics Canada, 1989] support some user interaction during the course of the simulation. In practice, however, such interaction is quite useful for model execution setup and debugging, and is used sparingly during model execution itself.

[7] Analogies abound in the domain of current productivity tools. Real time response is the difference between doing one's own word processing compared to relying upon a secretary or a secretarial pool. Alternatively it is analogous to constructing one's own model using a spreadsheet compared to working with a programmer to write a special purpose program. The simpler and more intuitive the interface and the more immediate the response, the more people can and will use it, and technical intermediaries become less necessary.

[8] The costs of performing a simulation experiment are distributed over several areas. They include the formulation of the underlying microanalytic model or revision of its components, setting up the specific simulation exercise, executing it, and analyzing its results. Although the actual computer based simulation portion has decreased substantially in cost over the last 30 years, the cost and turnaround time of this step are central in that they determine the feasibility and scope of studies that can be attempted. Overall costs of microsimulation activities are now becoming dominated by the cost of research, programming, support, and operation related to microsimulation, not by raw computing costs.

[9] See [Cotton and Sadowsky, 1991], chapter 3 for a review of pertinent literature and analysis.

[10] This result is derived from the analysis in [Cotton and Sadowsky, 1991]. The analysis was performed in summer 1989. Two years later, the prediction appears to be on target, and if anything perhaps conservative.

[11] The MIP as a unit of power is not an exact standard, since it depends upon the power of the instruction set of the processor being measured. Other measures exist, such as Specmarks and results from various compute-intensive benchmarks. While there is some fuzziness in the MIP measure,we believe that approximate comparisons are valid as long as the processors being compared are RISC (*R*educed *I*nstruction *S*et *C*omputer) systems.

[12] Vector processors process multiple streams by using pipelining hardware which is analogous to a manufacturing assembly line; the appearance of processing multiple data streams is in reality caused by efficient overlapping of parts of instructions. The more than proportional gain in throughput is real nonetheless. Other computing systems, mostly ones with multiple processors with multiple local memories, can truly process multiple data streams with a high degree of parallelism. The Connection Machine, manufactured by Thinking Machines, Inc., is one such architecture.

[13] For example, Apollo Computer's NCS (*N*etwork *C*omputing *S*ystem) provides a mechanism whereby subroutine calls can be made between subprogram modules residing on different machines connected by a data communication link. Arguments, other linking information, and results are passed between machines over the network. Although the entire process incurs some overhead due to the intersystem communication, the gain accruing from the ability to tailor the resource to the subtask may yield a substantial improvement in overall productivity.

[14] An example is provided by Sun Microsystems' NFS (*N*etwork *F*ile *S*ystem), which allows a computer running Unix to graft all or part of another Unix computer's file system onto its own.

[15] Decomposition of programs into front and back ends appears to be an effective manner of exploiting comparative advantages of dissimilar but linked computer systems. For example, specific front end processors for *Mathematica* (Wolfram Research, Champaign, IL) have been built for a variety of computer systems, while a common computationally intensive back end is written in Unix and can run on any system, local or remote.

[16] [Bell 1989], page 1100. Bell discusses at length different machine architectures and their potential for radically improving computational speeds and productivity.

[17] For example, the Hewlett-Packard model 730 workstation recently announced is rated by its supplier at 67 MIPS and 28 MFLOPS (*M*illion *FL*oating point *O*perations *P*er *S*econd). A configuration having 64 MB of primary memory and 840 MB of disk memory is available at a standard education and research discounted price of about $33,000 in mid-1991.

[18] Part of the speed improvement in some current high speed workstations results in there being more than one processor operating simultaneously, perhaps on the same processor chip. At a finer level of granularity, such processors might be regarded as multiprocessors, even though the programmer and system user see the workstation as a SISD system. The translation between the two views is accomplished by high level language compilers, which map the single instruction stream into an equivalent set of machine instructions that obtain the greatest efficiency from the underlying machine architecture.

[19] As an example, each individual CM processor evaluates conditional statements to an arbitrary depth and executes or not depending upon the result of evaluating the conditional expression with data local to that processor. Each CM processor receives every instruction broadcast, and the decision to execute is local. In such an environment, a case statement with a large number of cases might be very inefficient, whereas table driven computations that substituted data array indexing might appear awkward in form but execute very efficiently.

[20] See [Cotton and Sadowsky, 1991], chapter 4, for a longer discussion of this subject and examples of what such modular decomposition might look like.

[21] DOE/NASA/NSF meeting on scientific visualization, Bodega Bay, California, May 1991.