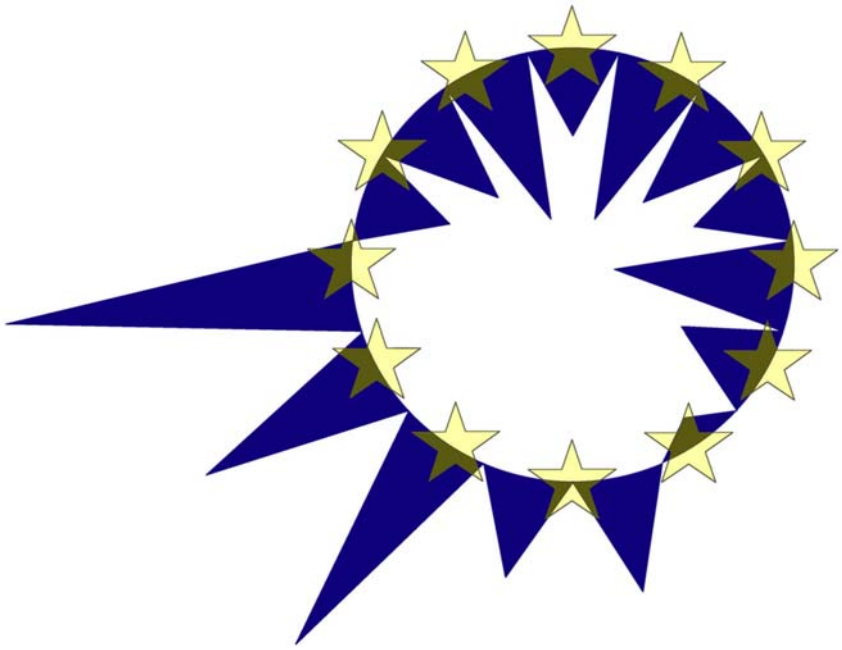
EUROMOD

WORKING PAPER SERIES



EUROMOD Working Paper No. EM2/01

TOWARDS A MULTI-PURPOSE FRAMEWORK FOR TAX-BENEFIT MICROSIMULATION

Herwig Immervoll and Cathal O'Donoghue

December 2001

Towards a Multi-Purpose Framework for Tax-Benefit Microsimulation: a discussion by reference to MMEANS, a software system used for constructing EUROMOD, a tax-benefit model for the European Union[1]

Herwig Immervoll and Cathal O'Donoghue

EUROMOD Working Paper No. EM2/01
December 2001

Abstract
This paper introduces a generalised model building platform (MMEANS) for implementing and using tax-benefit microsimulation models. It is designed to aid in the construction of single- and multi-country tax-benefit models by providing all essential components and a system by which these can be parameterised and combined into a full model. We explain the conceptual and computational issues arising in the design and development of MMEANS. One application of the software has been to construct EUROMOD, a 15 country European tax-benefit model (Immervoll *et al.*, 1999; Sutherland, 2001). However, we argue that, apart from its direct usefulness for this model, MMEANS can be used as a general software tool for microsimulation model building.

JEL Classification: C81; C88

Keywords: Microsimulation; Tax-Benefit Model; European Union

[1] Immervoll is Research Associate at the Microsimulation Unit in the Department of Applied Economics at the University of Cambridge and Research Fellow at the European Centre for Social Welfare Policy and Research, Vienna. O'Donoghue is Research Associate at the Microsimulation Unit in the Department of Applied Economics at the University of Cambridge. This paper was written as part of the EUROMOD project, financed by the Targeted Socio-Economic Research programme of the European Commission (CT97-3060). We are grateful for helpful comments from Holly Sutherland, Heikki Viitamäki and participants at the EUROMOD advisory group meeting (Copenhagen), the Nordic Microsimulation Workshop (Copenhagen) as well as from conference and seminar participants in Brussels (EcoMod, 2001), Cambridge, Rio de Janeiro, at the United Nations Social Policy Division and the World Bank. The authors alone are responsible for any errors, as well as the views presented in this paper.

Towards a Multi-Purpose Framework for Tax-Benefit Microsimulation. A discussion by reference to MMEANS, a software system used for constructing EUROMOD, a tax-benefit model for the European Union.

1 Introduction

Tax-benefit microsimulation models (MSMs) have been widely used in many countries for a number of years. Recently there has been interest in carrying out cross-country comparative exercises, examining the performance of policy instruments in different countries. Because of the limitations of national specific models in this regard (See Callan and Sutherland, 1997), an integrated multi-country model, EUROMOD, has been developed by a consortium of teams in all 15 EU countries (Immervoll *et al.*, 1999; Sutherland, 2001). This paper describes MMEANS[2], a generalised software framework for building and using MSMs. MMEANS consists of a set of tools and components which have been used to develop EUROMOD and its 15 sub-models and can aid in the implementation of MSMs for any country's (or group of countries') tax-benefit system(s). The aim of the present paper is to explain the conceptual and computational issues arising in the development of MMEANS. Although some reference is made to EUROMOD as the first application of MMEANS, the paper is not intended to serve as a guide to EUROMOD. More user-oriented treatments of the various steps and aspects of using EUROMOD, including illustrations of various components discussed here, are provided in Immervoll (2002), Immervoll and O'Donoghue (2001c) and Mantovani (2002).

Microsimulation and tax-benefit models

Simulation is a method by which a 'model' of some sort is used to generate specific output given a specific input. The *specific* nature is what sets simulation methods apart from mathematical approaches that seek to derive, in a deductive manner, *general* representations of relationships between variables of interest. Simulation techniques, on the other hand, take the inductive route, providing specific results of a model given the particular initial variable values and model parameters (Orcutt, 1986). They are especially useful in situations where the general mathematical specification of the mechanisms of interest is unfeasible or considered too laborious.

Microsimulation models do not explicitly specify relationships between variables of the system as a whole (e.g., the overall transfers between the public and the private sector, or the changes of the demographic composition over time). Instead, they are built around models of relevant relationships at the micro-level, which seek to explain endogenous variables Y (such as disposable income, labour market participation, survival) as a function of exogenous variables X (e.g., socio-economic characteristics of the micro entity) and system parameters P (such as tax rules, the shape of relevant utility functions, or life-expectancy).

[2] *M*icrosimulation *M*odelling *E*nvironment and *AN*alysis *S*oftware

By applying these models to a large number of *specific* micro-entities (e.g., household members as observed in a representative sample of the household population) the behaviour of the overall system can be explored along with effects at the micro level. Each simulation run can be considered as an 'experiment' where, given the model parameters *P* and the population characteristics *X*, certain results are obtained. Each new specification of *P* corresponds to a different 'treatment' of the subjects of the 'experiment'. It is therefore possible to not only compute totals and averages of the endogenous variables (as, for example, macroeconomic models do) but also their distribution across the different micro units. Since both *P* and *X* may be varied, microsimulation models can be used to investigate consequences at the micro level of both changes in system parameters (e.g., macroeconomic variables) and changes in the underlying population (demographic changes, changes in the distribution of market income, etc.). It is therefore possible to capture the interaction between various influences. Of similar importance, however, is the ability of microsimulation models to hold constant many variables in order to unpick the isolated effects of the phenomenon of interest. For example, one can establish the distributional consequences of changing social and fiscal policy legislation under the assumption of an unchanged population or vice versa. By separating the various forces at work, microsimulation models contribute to a better understanding of complex systems (Orcutt, 1957; Orcutt *et al*., 1976).

Tax-benefit microsimulation models are computer programs that calculate tax liabilities and benefit entitlements for individuals, families or households in a nationally representative micro-data sample of the population and are used by both governments and academics to study existing social and fiscal policies as well as policy reforms. As micro models, they take as the basis of their analytical framework the micro-level, typically individuals, families and households.[3] As simulation models, they simulate the detail of institutional rules and thus are in a position to evaluate existing tax-benefit legislation and aid in the design of new individual schemes or entire systems. They calculate applicable amounts of each element of the tax-benefit system in the legal order so that interactions between different elements of the system are fully taken into account. The resulting taxes, benefits and income measures for each individual, family or household are weighted to provide results at the population level. MSMs have been developed and are in use in many OECD countries (see Sutherland, 1995 and the references cited therein).

By incorporating the interactions of different elements of the tax-benefit system and by taking full account of the diversity of characteristics in the population, this approach allows a very detailed analysis of the revenue, distributional and incentive effects[4] of individual policy instruments and the system as a whole. In particular, it provides a powerful means of performing "what if" analyses by allowing the analyst to manipulate all relevant parameters of the system such as tax rates, thresholds, amounts, income concepts, in an intuitive and user-friendly environment (see Redmond *et al.*, 1998). It is thus possible to determine the marginal

[3] Although it can, for some purposes, be useful to analyse the effects of policy on hypothetical populations (see, e.g., Immervoll *et al*, 2001b), these models typically build on micro databases drawn from either household surveys or administrative register information. Using data on actual populations, MSMs can be used as tools to analyse 'real-world' effects of social and fiscal policy. Because they compute variables (such as taxes and benefits) that may not be observed in the underlying data, MSMs are also routinely used as instruments to "enrich" existing databases by imputing missing variables (Weinberg, 1999; Immervoll and O'Donoghue, 2001b).

[4] They allow us, for instance, to analyse replacement rates or marginal effective tax rates. See Immervoll (2000) and Immervoll and O'Donoghue (2001a).

effects of policy configurations, both in terms of changing certain policy options and by varying underlying assumptions of the model (e.g., Lewis and Michel, 1990). As a result, MSMs can be an important aid in improving policy design.

The focus on a large number of micro-entities avoids problems of aggregation bias encountered in models operating at higher levels of aggregation. In addition, the model representation of detailed processes at the micro-level contributes to an improved understanding, by both model-builders and users, of the mechanisms underlying the effects of policy measures. However, the required size and detail of micro-data and the complexity of relationships to be modelled (in the case of tax-benefit models, the institutional rules governing taxes and transfers) pose a serious challenge for model builders. This challenge may also carry over to model users who may be faced with models that are too complex to be useful in practice. Tax-benefit models need to be flexible enough to allow the simulation of far-reaching and *ex ante* unknown policy alternatives while keeping the specification of relatively minor policy changes reasonably simple.

As a result of the complexity of MSMs and the conflicting demands of usability and scope of analyses that are possible, "the development of micromodels frequently needs too much time and its costs are accordingly high."[5] Designing re-usable model components that can be used in many different models is one logical step towards making model-development less resource intensive. In the case of EUROMOD, the goal of implementing an integrated model covering all 15 EU countries provided an additional motivation for developing MMEANS, since it was to be expected that implementing 15 tax-benefit systems would present potential synergies that should be exploited.

Multi-country tax-benefit models

Recent years have seen an increasing demand for tools to perform international studies, particularly in Europe. This has been driven by stronger socio-economic links between countries, a more comparative focus in policy analysis and a desire to verify theories in different national settings. While building country specific MSMs is a complex and resource intensive task, designing a multi-country model brings up entirely new issues. Previous research using cross-country microsimulation provides a guide to some of the approaches, opportunities and pitfalls. Research can essentially be divided into three types:

(a) comparisons using a single country MSM;

(b) comparisons using different national models; and

(c) models embedded in a consistent and comparative design.

Type (a) models apply different national systems on the population of one given country: the tax-benefit rules represented in a given MSM are adjusted to resemble the tax-benefit rules of a second country. Examples include Atkinson *et al*. (1988), who compared the impact of replacing the French tax-benefit system with the UK system and O'Donoghue and Sutherland (1999) who studied different European family tax instruments using UK data. Abstracting

[5] Hoschka, 1986, p. 46.

from differences in population structures, they can examine the direct impact of different national systems as applied to one specific population. However, because policy instruments are designed with a particular national policy, or social context in mind, care must be taken in interpreting results that ignore these differences. Type (b) models incorporate the differences in national populations and income distributions by using different existing models. For example, Callan and Sutherland (1997) compared the Irish and UK tax-benefit systems using separate models for Ireland and the UK. Comparing two very similar tax-benefit systems in this way is a complex exercise. Yet, due to large conceptual differences between national models in terms of their structure, definitions, scope and output, extending the analysis to cover additional countries while maintaining comparability proved to be insurmountable.

Type (c) models which have recently started to be developed try to address these difficulties. As a step towards an EU-wide model, a prototype six country model, Eur6 (Bourguignon *et al*, 1997) was constructed and has avoided many of the pitfalls associated with using different national models. As an integrated methodology, designed from the outset for comparative purposes, such models allow for flexibility in specifying the optimal data and modelling definitions. Using this type of model it is possible not only to compare national model results but to also pool them across countries, e.g., allowing for the position of individuals from different countries to be analysed in the context of a multi-country (e.g., a 'European') income distribution. Atkinson *et al*. (2001) use a prototype European model in a case study of a European minimum pension.

MMEANS draws on the experiences from the development of the Eur6 prototype. MMEANS was first designed with the aim to run, on one integrated platform, the 15 national sub-models that make up EUROMOD. Doing so presents several problems. Each national tax-benefit system has a different structural logic and accommodating this structural diversity while keeping the model logically correct, robust and transparent to users is a major task. This is complicated further by the aim to be able to transfer policy instruments *between* countries to see, for example, what effects a public transfer X of country A would have if implemented in country B. Also, being a model that is to operate at the European level, EUROMOD needs to be able to evaluate the differential effects that a common policy instrument would have in the different member countries. A consequence of these requirements is, for instance, that each instance of a policy instrument needs to have a common interface so that it can be taken out of its original context and "plugged" into another system.

In what follows, we will consider some of these design issues and illustrate possible approaches by reference to the MMEANS microsimulation modelling platform. The structure of the paper is as follows. Section 2 discusses the general objectives and desirable features of MSMs. Using a single platform for a multitude of different tax-benefit systems requires the essential 'building blocks' of tax-benefit systems to be conceptualised independently of any specific country's tax-benefit system. In other words, it is necessary to identify a *general* structure that is appropriate for modelling any conceivable system of tax and transfer legislation. Section 3 outlines the generalised structure of tax-benefit systems on which MMEANS is based. Section 4 describes the principal design of the microsimulation framework and the computing environment into which it is embedded. We also explain the modelling components provided by MMEANS and how they can be combined to construct a complete MSM . In the following part of the paper, we discuss how the framework has been

"generalised" in order to be usable for building a broad range of different MSMs. Section 6 goes into more detail by relating the various design issues raised earlier to key concepts of tax-benefit model building and explaining how these key concepts were implemented in MMEANS. The final section concludes.

2 Objectives and Desirable Features of MSMs

Previous studies which have focused on the overall design of tax-benefit models have mainly focused on data- and broader design issues (Hoschka, 1986; Merz, 1991; Citro and Hanushek, 1991a, 1991b; Sutherland, 1995). Our paper aims to build on previous work by operationalising accepted concepts in terms of the detailed computational design of MSMs. In addition, it will also discuss entirely new aspects which only arise in building an integrated multi-country model.

To set the scene as to how a microsimulation development environment should be created for tax-benefit modelling, it is useful to start by discussing the potential demands placed on the model. Broadly the objectives can be classified under the following headings:

- Flexibility;
- Ease of use;
- Robustness;
- Transparency and consistency of structure and concepts;
- Maintainability; and
- Cost effectiveness.

It is the role of tax-benefit models to assist in the analysis of existing and alternative policy scenarios. Depending on the purpose of the analysis, scenarios to be analysed will often need to satisfy a number of requirements such as revenue neutrality, improving work incentives, reducing poverty, etc. Because tax-benefit systems are highly non-linear with a large number of parameters, the list of possible constraints is literally endless (see, e.g., Sutherland, 1991). It follows that the design of tax-benefit models should be *flexible*, enabling users to specify a wide range of different policy scenarios and making it easy to switch between scenarios. Hancock (1997) argues in a first analysis of the computing requirements for EUROMOD that flexibility is probably the most important corner stone of the computing strategy.

The development of MSMs involves the construction of a software environment to handle large amounts of data, the simulation algorithms themselves as well as input/output routines and user interfaces. It also involves the transformation and matching of existing micro-datasets and the translation of tax-benefit laws into a computational framework (Klösgen, 1986; Mot 1992; Merz, 1995; McCrae, 1999). An important expense is updating the model which involves repeating many of these steps in more or less regular intervals. Doing so separately for each country multiplies costs, while using one single microsimulation

framework for different countries or purposes can be a very *cost-effective* method of building new models and maintaining them.

Although MSMs may in part be constructed by computer programmers, typical users will include economists, statisticians and social policy analysts in both academia and government. *Ease of use* is to ensure that all relevant features of the model are accessible to a wide range of users rather than just model developers. At the same time, the complexity of the model should be organised hierarchically. In other words, it should be possible to use 'basic' features of the model without having to know all the details about more 'complex' model components. This ensures that the model is powerful while at the same time being useful for users with different backgrounds or different analyses in mind.

To enable users to make changes to tax-benefit algorithms in a relatively safe environment, one would ideally have a set of pre-fabricated building blocks that could be re-configured in order to implement algorithms for every possible tax-benefit instrument without any need for major reprogramming. Each element should be a derivative of a basic template and should have the same type of input and output data structures. Only the core algorithm which determines the behaviour of the element would need to be element specific. Once users have become familiar with this structure, they can then adapt any tax-benefit algorithm without having to 'dig' through program code. However, in general there exists a trade-off between flexibility and *robustness*. It is technically possible to develop highly flexible elements that, through parameterisation, can be used for many different purposes. For example, a generalised tax allowance that could in principle be used to construct all types of tax-allowance used in a set of different countries. However, a very large number of parameters which attempt to provide for any potential use of an instrument may result in a model that is both difficult to use and is more prone to produce errors through mis-specification of parameters.

Once accustomed to the operation of one country sub-model, users of a multi-country model need to be able to access the parameters of other countries' tax-benefit systems in a similar way. The multitude of necessary definitions and concepts (e.g., income concepts, fiscal units, sharing rules, as discussed below) mean that a *consistent* specification of relevant concepts across countries is essential. As highlighted in the previous section, simply lining up national models next to each other is not suitable as the design of national models will tend to reflect national priorities. Such conceptual differences will make it hard or impossible to compare the results of different models and prevent a consistent specification of policy reforms. A generalised modelling framework should therefore allow consistency in the specification of different systems. On a related point, given the fact that a typical tax-benefit system of any one single country encompasses thousands of parameters, all of these parameters need to be readily accessible and organised in a *transparent* manner.

In order to be able to contribute to the debate of contemporary policy issues, tax-benefit models will need *maintenance* on a regular basis. In addition to frequent revisions of institutional tax-benefit rules along with any behavioural model parameters, underlying micro-data will need to be updated regularly so that model results continue to be based on representative data. The model framework should therefore be able to access and organise

different data sets with ease and provide tools for updating 'older' micro-data to a current (or future) year.

Another aspect of *maintenance* relates to model testing and validation. The complexity of tax-benefit systems make validation of the logical correctness an essential component of any model building project. Finding and analysing discrepancies between model results and reference figures should be aided by a model structure which allows the relevant model algorithms to be 'traced' in order to find any modelling errors. Similarly, it is important to break complex algorithms down into manageable pieces that can be analysed separately. Apart from logical errors in the model algorithms, the micro data, which provide values of the exogenous variables used in these algorithms, are the other main source of error. There are various techniques that can be applied to 'improve' the quality of micro data *prior* to using them as input for microsimulation (matching of information distributed across different data sources, weighting schemes, etc.). Indeed, the transformation of existing micro-data into a format usable for microsimulation purposes is one of the most demanding tasks of any model building project, particularly if more than one country is involved (Sutherland, 2001). Yet, microsimulation models essentially have to take as given the quality of micro-data resulting from these exercises. However, the model should inform users about the potential seriousness of data related errors by providing statistics that enable them to judge the degree of sampling errors. In addition, by providing quick access to model parameters and assumptions, the model structure should enable users to repeat simulations under various scenarios in order to facilitate sensitivity analyses of the results obtained.

3 The Structure of Tax-Benefit Systems

A general tax-benefit modelling environment will ideally be able to accommodate any existing or hypothetical tax-benefit system. In designing such an environment it is therefore essential to identify the essential elements of tax-benefit systems. In other words, it is necessary to find a suitable 'common denominator' of all (reasonably) possible structures. However, in general, there exists a trade-off between structure and flexibility: The modelling framework needs to provide the structure necessary for setting up simulation models without limiting the range of tax-benefit systems that *can* be simulated.

In 'real world' tax-benefit systems, elementary policy rules are grouped together to form identifiable blocks such as 'instruments' (e.g., a tax credit) and 'policies' (e.g., income tax). In modelling a country's system, it is desirable to match the real system's hierarchy as closely as possible so that the logical representation in the model provides a good intuitive equivalent of the original. From a model construction point of view it is desirable to try to generalise this representation as much as possible, so that most national systems can be described utilising the same structure.

Figure 1 shows the hierarchical structure that we use for a general tax-benefit modelling framework. The figure also introduces the terminology used in the remainder of this paper. Each tax-benefit **System** is made up of individual **Policies**. These are collections of tax-benefit instruments. Examples for a Policy are income tax (IT), social insurance contributions (SIC) or social assistance benefits (SAB). The **Policy Spine** is a list of Policies indicating the sequence by which they are applied in the tax-benefit System. The Spine thus controls the

flow of the computational operations taking place within the model. For example, if SICs are tax deductible, then the SIC Policy would have to appear *before* the IT Policy because the model requires the amount of SICs as a prerequisite for calculating IT; similarly, if SAB depend on after tax income, then the SAB Policy would have to appear *after* the IT Policy since IT (and, as a result, SICs) would be a necessary input for calculating the SAB amounts applicable for each family. At the lowest level of the hierarchy is the tax-benefit **Module**, which performs the calculation of a certain part of the tax or benefit (e.g., a deduction, or applying a rate schedule to a tax base). The Modules represent the elementary building blocks of the tax-benefit system: Only they contain actual tax-benefit algorithms. All other levels are merely necessary to structure and organise these rules and for applying them in the appropriate order.

4 Main components of MMEANS and computing environment

Figure 2 shows the principal components of MMEANS and how they relate to the structure of tax-benefit systems described above. The main functional elements that MMEANS provides for microsimulation modelling are:

- the micro data access and management system;
- the input routines for handling the parameters specified by the model user;
- elements for producing model output and for analysing this output;
- and, at the core of the model, a system for managing the correct sequence of the tax-benefit algorithms. This is built around the generalised structure of tax-benefit systems as described in section 3.

In addition, there are explicit links, which permit optional model elements to be added to models based on MMEANS. In figure 2, these optional elements are indicated by dashed lines:

- The simulation of behavioural responses to policy changes normally involves the evaluation of opportunity sets ('budget constraints') showing the effect on disposable income of a alternative behaviours (e.g., different numbers of working hours). MMEANS supports the computation of such opportunity sets. Essentially, this simply involves simulating tax and transfer amounts for all behavioural alternatives of interest. Given the shape of indifference curves estimated from an econometric model, it is then possible to find the welfare maximising point on the opportunity locus.[6,7]
- 'Ageing' of micro-entities includes techniques to account for changes in the population on which the simulations are to be performed. This is, for example, necessary if the structure of a population has changed between the period when the

[6] See, e.g., Pylkkänen (2001).
[7] In EUROMOD, this potential link to behavioural modules has not been exploited to date. The limited resources of the EUROMOD model construction project were targeted towards implementing a detailed and comparable model representation of the tax-benefit rules existing in the 15 countries (but see Klevmarken, 1997 for a discussion of issues related to including behavioural dimensions in EUROMOD).

data were collected and the period one is interested in. Static ageing leaves the characteristics of each observation unchanged. It seeks to approximate the structure of a given period's population by merely altering the weights assigned to each observation included in the original data (see section 6 below). Dynamic ageing methods, on the other hand, simulate the relevant processes at the micro level that underlie the structural changes of a population from period to period. This can be desirable simply in order to approximate a dataset for a certain period using existing data for an earlier period. However, one may also be interested in the dynamic processes *per se* and how they interact with a country's tax-benefit system in order to analyse, say, longer-term effects of a policy measure or issues of life-time redistribution (Falkingham and Lessof, 1992). In a household dataset, the most trivial of these changes concern people's ages. Other processes and transitions that are typically taken into account are deaths and births, household formation and labour market participation.[8] Although MMEANS itself does not provide tools for building dynamic models, tax-benefit models based on MMEANS can be linked into dynamic models.[9] A successful implementation of such a link is described in O'Donoghue (2001b).

- MMEANS provides a set of parameter files to store and alter parameters relevant for specifying all aspects of a simulation run (see section 5 and 6 below). Based on these parameter files, a user interface (UI) can be implemented in order to simplify user interaction with the model (see Schofield, 1995). For example, one may wish to design a graphical UI that presents the large number of parameters in an hierarchical manner so that regular users do not see all parameters relating to more 'advanced' uses of a model. Models based on MMEANS can work alongside any type of UI as long as the UI component is able to read and store all simulation parameters entered by the user in the standard format parameter files which MMEANS requires.

4.1 Steps of the simulation process

The individual components of MMEANS shown in figure 2 are related as follows. The data access element separately reads information on each micro-entity from the micro-database and updates monetary information (see section 6). All observations (e.g., individuals) which are part of this micro-entity (e.g., household) are then grouped together to form the assessment units to which the tax-benefit rules relate (e.g., "married couple" for joint income tax, see section 6 below). The core of the model consists of the tax-benefit algorithms. For each instrument, these are performed separately for each assessment unit. The algorithms are organised according to the generalised structure of tax-benefit systems discussed in section 3 and are started in the order specified in the relevant parameter files. All outputs of these calculations (monetary amount of the simulated instruments, measures of disposable income, etc.) are then written to a micro-output file containing all variables of interest for each unit of

[8] If state transitions in the dynamic model are not purely probabilistic but instead based on behavioural models then some processes will themselves depend on simulated tax and benefit amounts. For reasons of simplicity, this link has been omitted in figure 2. To allow for such feedbacks, the possibility sets computed by the 'Behavioural Response' module would be need to form an input into the 'Ageing' module.

[9] A detailed survey of existing dynamic microsimulation models is provided by O'Donoghue (2001a).

analysis (e.g., individual or household).[10] This main loop, indicated in figure 2 by bold boxes and connectors, is repeated over all (*N*) micro-entities stored in the micro-data.

4.2 Computing environment

An effort has been made to ensure the longevity of MMEANS by not irrevocably attaching it to one specific computing environment. Currently, MMEANS is implemented on a standard Windows platform. However, care has been taken to avoid a rigidity, which would prevent future adaptations to other platforms such as UNIX. The programming language used is C/C++. This facilitates efficiency in programming. However, the ability of C/C++ to write very streamlined and "direct" algorithms sometimes reduces the readability and transparency for less experienced users. As a rule, where trade-offs existed between transparency and speed, we accepted lower speed in return for improved readability and usability.

By default, MMEANS does not provide a custom-made user-interface. Instead, users communicate with their models by means of parameter files (see figure 2) stored as well-formatted spreadsheet tables and can be read and manipulated with any spreadsheet software (e.g. Microsoft Excel). Many functions of these existing and widely available software packages can be utilised by users in order to view, change and archive model parameters. Using the functionality of existing software such as Microsoft Excel, it is thus possible to use models based on MMEANS without going through the time intensive process of developing graphical user interfaces. However, as discussed above, MMEANS permits such user interfaces to be linked into models where this is desirable.

Communication with the micro-data has been implemented using a standard interface for database access (ODBC) which is supported by popular operating systems and is available for all major relational database management systems and statistical packages. As a result, MMEANS is independent of the database system used. Almost any database system can be chosen for storing and managing the micro-data. Both the input micro-data (i.e., data that feed into the model algorithms) and the model's micro-output (i.e., simulation results for each micro-entity) can be stored in one of the widely used relational database systems (Oracle, Microsoft Access, Microsoft SQL, etc.). Although less efficient in terms of storage space, it is, by virtue of the standard ODBC interface, also possible to access data stored in, say, SPSS, spreadsheet or text formats. Input- and output data can be stored in separate databases. In this way, the input micro-data can remain "read-only" and, if required by data access restrictions, altogether hidden. General advantages of using relational database systems for data management in microsimulation models are efficiency in data storage and, via the Standard Query Language (SQL), access to well tested and powerful data manipulation and analysis tools. For example, the relational data structure makes it possible to combine the physically separate input and output data into one logical table in order to analyse the results of specific policy scenarios in relation to all sorts of characteristics (age, household size, etc.) stored in the input database. However, it is also possible to produce 'flat' format text output files (see discussion of output tools below).

[10] The unit of analysis will often be different from the various units of assessment required for computing tax and benefit amounts.

5 Generalisation and parameterisation

In order to satisfy the numerous requirements and accommodate the different uses outlined in section 2 as well as to permit the individual boxes of the hierarchy described in section 3 to be 'filled' with any type of tax-benefit algorithm, the modelling framework needs to be quite generalised. The degree of generalisation relates to the degree to which a model is 'parameterised' so that model code can be re-used for different purposes without having to alter the underlying computer code. For instance, a 'tax-schedule' Module that is programmed in a way that works with any number of tax bands and any set of tax-rates can be re-used for modelling tax schedules of many different countries. While generalising as much as possible makes a model more flexible, it also has the effect of making it more difficult to develop, potentially less transparent and conceptually and computationally more complex (and, hence, slower) than models which is built for narrow and *a priori* clearly defined sets of applications.

More specifically, there exists a trade-off between demands placed on end-users and the scope and flexibility of a model in terms of the scenarios and policy reforms that can be simulated. MMEANS does not impose any *a priori* position in this simplicity vs. flexibility spectrum: it is possible to construct very simple models that are limited in scope and more complex models offering a very large range of parameters and, hence, policy options. The quality of any model in this respect depends on the number and content of the individual 'boxes' outlines in section 2. In other words, it depends on the specification of the tax-benefit algorithms.

For example, splitting up an income tax policy into many individual Modules (e.g., separate Modules for each deduction, each tax-free allowance, the tax schedule and each tax credit) and reading all relevant parameters (e.g., amounts, limits, rates, income concepts such as taxable income, definition of the tax unit) from the external parameter files enables to user to specify many policy reforms without having to re-program any of the algorithms. Instead, users would simply change the value of appropriate parameters. The actual tax-benefit algorithms would be coded as functions of these externally defined parameters and this code will not normally have to be accessed by the model user.

On the other hand, if the entire income tax algorithm is "packed" into one single Module with many model parameters "hard-wired" into the algorithm rather than accessible via external parameters lists, these lists would be shorter and (maybe) easier to deal with from the users' perspective. However, almost any policy reform would require re-programming of the tax algorithm contained in this Module.

Since tax-benefit models are to a large extent about analysing policy *reforms*, we are clearly in favour of flexibility even if this sometimes results in very large numbers of policy parameters. The number of parameters merely reflect the actual complexity of tax-benefit systems. If these systems are to be well represented by the tax-benefit model then a reduction in the complexity is, for most analysis purposes, not desirable.[11] Indeed, explicitly showing all relevant model parameters (rather than hiding them in the program code) has the potential of

[11] However, the ability of microsimulation models to provide a platform where users can experiment with different policies also makes them a promising tool for teaching purposes (see, e.g., Merz, 1995). In these cases, there may, for pedagogic reasons, frequently be a case for reducing complexity.

considerably increasing the transparency of the computations within the model. Parameterisation also facilitates experiments with different parameter values – an important aspect since tax-benefit model are well suited for fine-tuning policy proposals.

In constructing EUROMOD, we have, for example, made an effort to break tax-benefit algorithms down into manageable Modules and Policies that have an intuitive equivalence to real-world tax-benefit instruments. We have also tried to parameterise all relevant features of these algorithms.[12] What was regarded as 'relevant' in this context depended on the likelihood of a given feature (e.g., tax unit) requiring adjustment for specifying a policy reform or for re-use in another context/country. Given the larger scope of re-using Modules (not only across different policy instruments but also across countries), parameterisation is even more relevant for the construction of a multi-country model.

Although more costly to build initially, an extensively parameterised model is less costly in the long run if it can be used for a multitude of purposes. In addition, the robustness and reliability of a modelling framework such as MMEANS will be positively related to the number of users and uses. As a result, a generalised modelling platform that is used for many different purposes can 'mature' more quickly than purpose-built models with a more narrowly defined scope. A generalised multi-purpose framework will also facilitate communication and co-operation between researchers and reduce training costs as many people will share similar experiences and problems.

Despite these points in favour of model flexibility, the trade-off against usability remains. There are several design strategies that can increase usability even if the full complexity of real-world tax-benefit systems is represented by the model. Important issues in this respect are

- a hierarchy of parameters (figure 1);
- consistency of concepts across all parts of the model;
- sensible default values for all parameters but particularly those which will only be changed infrequently;
- automatic consistency checks of parameter specification and useful error/warning messages in case of logical misspecification;
- and, more generally and related to all previous points, predictability of model behaviour.

6 Implementation of key concepts for tax-benefit modelling

Efforts were made to improve the usability of models based on MMEANS by considering the design issues mentioned above in the implementation of the modelling components. The realisation of these components will be discussed in turn.

[12] The EUROMOD Country Reports provide a description of what parts of national tax-benefit systems have been simulated in EUROMOD and how. They are available through www.econ.cam.ac.uk/dae/mu/emod.htm.

6.1 Modules

Modules, the primary building blocks of the model containing the actual tax-benefit algorithms. Components of Modules to be parameterised include the definition of parameters directly related to the respective tax-benefit algorithm (e.g., rates, bands thresholds, type of income concepts, fiscal units). The concept of modules as distinct building-blocks of a MSM has special advantages. By using the same building blocks for different instruments or tax-benefit systems, one can build up a large "library" of algorithms and can use these as a resource for specifying policy reforms or for exercises involving the construction of entirely new models. Examples are the introduction of a new benefit for which a generalised "eligibility" Module can be used instead of having to be programmed from scratch. Or, the use of existing Modules such as "schedules" or "means tests" or "deductions" as ready-made building blocks in constructing a new model for a specific country. The flexible order of modules and the high degree of parameterisation ensure that the same modules can be used for a multitude of different purposes.

In order for Modules to be re-usable in other contexts, it is necessary for them to have a common well-defined interface by which they can be linked into different places of a model. A standard interface is particularly important for a multi-country model as it becomes possible to take certain components or entire tax-benefit instruments from country A and link them into a model of country B. As shown by the EUROMOD project, this architecture opens up entirely new possibilities in comparative social and fiscal policy analysis. By creating all Modules necessary for the implementation of 15 European tax-benefit systems, the EUROMOD project has built up a large library of often very different instrument. As a result, there is already a wide range of "building blocks" which any analyst or model builder can, in principle, chose from for the purpose of specifying policy reforms or building entirely new tax-benefit models. As more models get built using the MMEANS framework (or existing ones updated to reflect evolving social and fiscal policy measures), this library will keep growing.

In terms of the actual implementation as computer code, there a number of desirable features. The structure of the Module should be a function with clearly defined inputs and outputs. The structure (or skeleton) of this function should look similar for each tax-benefit instrument. Every section should be clearly labelled and documented so that users wishing to adapt an existing instrument would readily see where changes have to be made while those wishing to implement a new instrument would only have to fill in the blank spaces. It should be possible to freely define intuitive variable names used in the algorithm to make interpretation as straightforward as possible. MMEANS provides a Module "template" which can be used for implementing model algorithms in a consistent manner.

Figure 3 describes the general structure of all Modules as implemented in MMEANS. The inward arrows define the set of inputs into the Module. Several steps are performed before the tax-benefit algorithm (as coded into the Module) is initiated. The first step involves determination of the relevant fiscal unit (unit of assessment) such as "individual", "household", "couple" or "family", that the instrument applies to. The next step involves storing all relevant parameters such as rates, bands thresholds, age limits etc. in variables that can then be used by the algorithm. One specific set of parameters relate to aggregate income

concepts. At this stage, the Module reads the definition of all aggregate income concepts used within the module such as "earnings", "total benefits", or "tax base" (see the discussion of aggregate income concepts below). The actual value of each required income concept is then later (and separately for each unit of assessment) computed within the tax-benefit algorithm (step 4b in figure 3). A last input is concerns the variables that should be accessible by the tax-benefit algorithm. Tax-benefit instruments are mostly a function of some characteristic(s) of the unit they apply to. In the model, these characteristics can be either recorded in the input micro-data (e.g., age, occupation or number of children in a static model) or they have themselves been simulated by the model (e.g., social insurance contributions, after-tax income or, in a dynamic model which simulates births endogenously, the number of children). To prevent accidental modification of recorded or model-generated variables, all variables which are required for the tax-benefit algorithm must be declared.

To minimise the scope for errors and interference with other parts of the model, each Module should only contain those parts of the program code, which are absolutely necessary for specifying the algorithm. Everything else should be "hidden" and not be accessible from within the Module.[13] Applying this design philosophy ensures that individual Modules can be developed independently (e.g., by different members of a model building team, as in the case of the EUROMOD model building project) since the operation of one Module does not interact with the operation of other parts of the model in unforeseen ways.[14] Once a Module has been thoroughly tested and is found to work, it can be added to the system as a whole. This basic technique is also employed, to varying extents, by other modern microsimulation models.[15]

In order to support safe and efficient implementation of new instruments, MMEANS provides a large number of frequently used standard functions. These are routines that perform operations or determine characteristics which are relevant for the simulation of many policy instruments (e.g., `NumberOfChildrenInUnit()`, `IsMarried()`, `IsLoneParent()`) so that Modules do not have to directly access certain variables in the micro data. One advantage of the use of such functions is that one can ensure a consistent interpretation of variable values across all Modules (especially categorical variables, such as marital status). As a result, those programming the tax-benefit algorithm do not have to worry about how certain variables are coded. On a related point, this also guarantees a consistency of relevant assumptions and definitions across different parts of a model. For example, a function `GetHeadOfUnit()` may use primary income, age and other characteristics to determine who is to be considered the "Head" of a family, a household, etc.. Since applicable conditions can be quite complex, using the same function to find the "Head" across all tax-benefit algorithms has clear advantages. Given that actual variables of the micro-data are accessed in only one place, this also greatly simplifies the maintenance of the model. For example, if the coding of a variable in the micro data or the assumptions as to who is, say, considered the "Head" of a unit should need to be changed, one only needs to adapt the functions referring to this variable/assumption rather than a large number of individual Modules.

[13] These concepts of "modularisation" and "encapsulation" are well known programming principles.
[14] Since there are often considerable interdependencies between different parts of a tax-benefit system, it will, of course, frequently be the case that Modules do interact in ways that are *intended*. However, such interactions will only take place via well-defined interfaces, viz. certain output variables of one Module (e.g., SICs) that are used as an input into another Module (e.g., deduction of SICs from the income tax base before applying the tax schedule).
[15] TRIM2 in the US (Mot, 1992), or TAXBEN in the UK (Giles and McCrae, 1995) among others.

In addition to modules that have been designed for a specific purpose in a specific country MMEANS also provides a large number of "common" Modules, which were designed without any single country or use in mind. Instead they can be used for many different purposes. An example already mentioned is a "schedule" Module where the number of rates, etc. is flexible and where the income base to which the schedule is to be applied can be freely defined. All these parameters can be specified in parameter sheets, which means that in many cases, even very complicated instruments can be implemented without any need for programming. Apart from the considerable amount of time and effort that can be saved by re-using already existing building blocks, there is, again, the added advantage that these "common" Modules have already been thoroughly tested. One can therefore be confident that the risk of programming errors is minimal.

For many countries, the sub-components of social benefits can be classified in a similar manner. It is therefore possible to classify benefits into a number of "common" Modules: Eligibility, Disregard, Means and Benefit Amount. Figure 3 describes these common elements of social benefits as provided by MMEANS. Eligibility is determined first. If a unit is found to be eligible and if the benefit is "means-tested" then their "means" m (i.e., the income that is set against a benefit) are calculated. In computing m, proportions of certain income sources such as earnings may be disregarded (e.g., to avoid undesired incentive effects). We denote the disregarded part of m as d. As a last step, we determine the amount of the benefit. Most benefits can be conceptualised as a base amount b times some scaling factor e that depends on certain characteristics of the benefit unit (e.g., the number of children).[16] The scaling factor can be seen as the equivalence scale implicit in this benefit, i.e., a factor, which is supposed to ensure that an "equivalent" amount is received by different types of benefit unit:

$$e = e\,(\mathbf{C})$$

where $\mathbf{C}$ is a vector of the characteristics of the benefit unit. The final benefit amount is then computed as

$$Benefit = b \cdot e \cdot (\,m - d\,)$$

By using a common structure for different benefits, this method improves the comparability of seemingly very different instruments. For example, permits e (i.e., the relative generosity of benefits for different family types) to be easily compared across countries without having to adjust for currency differences.

Both eligibility and equivalence scale modules contain large numbers of different types of parameters to permit modelling of many different types of benefits and reforms. For example, all eligibility conditions can be combined using a combination of logical AND, OR and NOT operators. In total, one can chose among several hundred possible parameters. In constructing EUROMOD, we were able to implement almost all benefit instruments that existed in the EU

[16] For example, we would specify a base amount 100 and an equivalence scale 1 for the head of unit, 0.7 for their spouse and 0.5 for each child. This method has the advantage (over specifying absolute benefit amounts such as 100, 70 and 50), of simplifying the specification of many policy changes. In specifying policy reforms, one will often want to scale benefit amounts up or down (e.g., up by 5%). Using the method with one base amount and equivalence scales, this can be done by changing only one parameter (setting the base amount to 105) rather than 3 (i.e., 105, 73.5, 52.5).

using the "common" Modules described here. A more detailed description of these Modules, including a list of all parameters, is provided in Immervoll and O'Donoghue (2001c).

6.2 Policies and Policy Spine

Policies are collections of Modules. As seen in figure 2, The Spine contains all Policies that are, in turn, run *separately for each micro-entity* stored in the input micro-data (e.g., households). Similar to Modules, Policies can only communicate with each in terms of well-defined output variables as specified by the model user. For example, if the only output of the SIC Policy is a variable called `sim_sic` then the only way this policy can influence the calculations of other Policies (e.g., income tax) is via this variable (e.g., `sim_sic` may be subtracted from the tax base in the IT Policy because SICs are tax-deductible).

A significant feature of MMEANS is that the order in which Policies within the Spine (and Modules within Policies) are simulated can be altered without any reprogramming. No specific order is "hard-wired" into the program. Instead, the individual model components are dynamically linked during run-time, based on the order specified in the model parameters. For example if one decides to make child benefits taxable, one would need to move child benefits ahead of income tax in the Spine (and include them in the definition of the tax base), while if child benefits were to be means tested on after-tax income, one would put child benefits after income taxes in the Spine. Similarly, the sequence of the Modules contained in a Policy determines the sequence of Module calculations. For instance, in the income tax Policy, "deduction" Modules would be computed before the income tax schedule is applied to the tax base.

Most Policies in the Spine are merely containers for Modules directly related to tax-benefit computations. However, MMEANS also provides two types of Policy that serve somewhat "special" purposes. The reason why they have also been conceptualised as Policies is that it is useful to be able to use them at different stages during the model simulations. As for any other Policy, this can be achieved by linking them into appropriate slots along the Spine.

One "special" type of Policy are the so-called "branches". The tax-benefit structure that we have described so far (figure 1) is linearly sequential. However in certain cases when a decision must be made between a different alternative instruments, "branches" are required in this structure. For example a benefit unit may be entitled to a range of different benefits and must choose one; or, as in the case of optional joint taxation, individuals may have to choose between being taxed individually or pooling income to form a joint tax base. Similarly, behavioural reactions to policy change are best modelled as an optimisation over a range of alternative decisions. MMEANS allows for such typologies by providing decision-rule elements. If required, these special types of Policy can be inserted anywhere along the Spine to construct the "branches" necessary for a decision.

The second "special" type is related to the generation of micro-output from the model. MSMs simulate taxes and benefits for each micro-entity contained in the input micro-data. The most direct output from such models is therefore a table with all simulated variables (taxes and benefits) for all micro-entities. This micro-output can then be further analysed using any statistical package or the "Summary Output" component provided by MMEANS (see figure

2). It is useful to be able to generate micro-output at different stages of the simulation. For example, one may want to output disposable income (or any other variable that changes as a result of the simulations) before and after certain instruments (e.g., before and after income tax). This may be desirable for analytical reasons (e.g., in order to find the isolated effect of some instrument on some variable) or as a technique for validating model results. By placing an output Policy in the desired slot in the Spine, it is possible to generate any number of micro-outputs, and, thus, to "trace" the value of variables through the process of the simulation. Of course, the typical position of the output element will be at the very end of the policy spine, writing simulation results for the tax-benefit system as a whole.

There can be any number of instances of the same output "policy" in the spine to ensure that exactly the same kind of output is produced at different stages of the tax-benefit calculations. In addition, there can be any number of different output "policies" in the spine to produce, for example, micro-output files with different sets of variables or for different units of analysis. Since a Policy is only a container for Modules, the output Policy, as any other Policy, needs to be "filled" with content. MMEANS provides special output Modules to be embedded in output Policies. These Modules provide parameters for specifying the type of micro-output one wishes to generate.

6.3 Definition of fiscal units

Tax-benefit rules relate to certain fiscal units, i.e., the person(s) on which the tax-benefit rules are to be performed; for example the persons over whom taxable incomes (means) are to be aggregated in order to determine total taxable income (total means) for a joint tax system (means tested benefit).

There are two ways in which conventional MSMs identify the units relevant for a given instrument. One possibility is to exploit explicit unit identifiers contained in the micro-data. For example, tax record data structure all information by tax unit. On one hand, this simplifies simulations since all the information in the data is already available at the required aggregation level. On the other hand, however, these models are severely limited in terms of the policy reforms they are able to simulate. Given the existing data structure, it will often not be possible to simulate a reform that entails changing the tax unit (e.g., moving from joint to individual taxation). In addition, it is difficult to use these data sources for simulating a range of instruments that use different fiscal units.[17] The other possibility is to determine endogenously within the model who belongs to which unit. This requires two things. First, sufficiently disaggregated and detailed micro-data (i.e., if each individual can be identified separately and if some basic relationships, such as parent/child, partner or living in same household, between these individuals can be observed). The second requirement is an algorithm within the model that can assign each data record to the relevant fiscal units. Given the country-specific nature of most MSMs, these algorithms tend to be optimised for a specific set of unit definitions, namely those which are relevant for a given tax-benefit system. As a result, there is little flexibility to change these definitions without major reprogramming efforts.

[17] Given data access restrictions, administrative data are, in any case, often only accessible within government departments. Since these departments tend to be mainly interested in the policy instruments which they are responsible for, not being able to simulate other instruments may not be regarded a major problem in practice.

As shown in figure 2, MMEANS also provides a routine to endogenously form the appropriate units. Given that the immediate motivation for developing MMEANS was to construct EUROMOD, it was necessary to find a flexible way to deal with the large number of different unit definitions across Europe. In a model based on MMEANS, each Module must contain the name of the *type* of fiscal unit on which the tax-benefit algorithm is to be performed (e.g., `SAB_Family`). Who belongs to which fiscal unit type can be specified using dedicated fiscal unit parameter lists. No re-programming is necessary for changing unit definitions. Since units across all instruments and countries are defined using the same format, it is simple to transfer unit definitions between instruments and countries. For example, if one wants to reform the Dutch child benefit by making it available to all families with children according to the definition used by UK child benefit rules, this can be done by simply copying the relevant parameters of the UK child benefit unit to that of the Dutch equivalent.[18]

In the simplest cases, the fiscal unit type is either the largest identifiable unit in the micro-data (usually the household) or the smallest (the individual). If it is neither then one has to define exactly which members of the largest unit (household) belong to the same unit as the "Head" of the fiscal unit. Possible choices are Cohabiting Partner, Married Partner, Child and Dependent Parent. For the latter two, a set of conditions is available for defining what constitutes a Child or a Dependent Parent (e.g., age limits, income limits, conditions relating to marital-, labour market-, or education status). All these conditions can again be combined using logical AND, OR and NOT operators. A pseudo-code of the routine used to assign people to fiscal units is described in appendix 1.

6.4 Sharing of benefits and tax burdens within the fiscal unit

By default, the results of all tax-benefit instruments in models based on MMEANS are assigned to the head of fiscal unit. However frequently it is desirable to be able to use other incidence assumptions. In order to do so, it is necessary to provide information about assumed sharing arrangements. MMEANS supports a number of different assumptions. In the current version, it is possible to select sub-groups of a fiscal units based on a number of characteristics and combinations thereof.[19] The benefit/tax amount is then shared among members of the selected sub-group.

Sharing in the selected group can be either equal among all members of this group or in proportion to the level of a particular income amount held by each individual. As an example, one could specify to share a jointly paid income tax between the adults of a family where the tax should be assigned in proportion to the taxable income of each adult. While the problems of empirically establishing appropriate sharing assumptions remain, by allowing explicit definitions of intra-unit assignments of simulated taxes and benefits, it becomes possible in principle to analyse simulation results at any level of analysis (e.g., gender specific rather than just at the household level).

[18] This has, in fact, been done in a study comparing the poverty reducing properties of EU family benefits. See Immervoll *et al*. (2001a).
[19] In the current version, these are adult/child; economically active/inactive; working part-time/full-time; and male/female.

6.5 Income concepts

Income concepts used in the tax-benefit algorithms (e.g., taxable income, "means", etc.) or as output of the model (e.g., disposable income) can be defined in terms of all monetary variables (whether contained in the micro-data or simulated by the tax-benefit model) available in the model. Each income concept is defined in terms of a vector of numbers between –1 and +1. The size of the vector is equal to the number of monetary variables in the model. For each of the variables, the number in the vector indicates what fraction of this monetary variable is part of the income concept. For example, if mortgage interest payments are deductible from taxable income then the "taxable income" vector would contain a "-1" entry for the variable mortgage interest payments. As with all other parameters in MMEANS, income concepts can also be exchanged between different countries (as long as the monetary variables available in each of these countries are the same).

6.6 Debugging aids and automatic checks for logical consistency

One aspect of usability mentioned above is an "automatic consistency checks of parameter specification and useful error/warning messages in case of logical misspecification". This is particularly important given the multitude of model parameters that can be specified. MMEANS incorporates several mechanisms by which errors in logic can be detected:

- Progress indicators: During run-time, models based on MMEANS report the micro-entity they are currently evaluating. This simplifies the process of finding problems/errors by enabling users to go back to, say, the household where the problem/error occurred.

- Missing/out-of-range parameters: Error messages are displayed showing the relevant model parameter and where it can be found.

- Plausibility of simulation results: Warning messages are displayed in case "very large" values result from an internal computation. This is to help avoid "divisions by zero and similar problems. Users can specify the relevant "very large" limits.

- Logical consistency of model structure: Given the flexible order of model components, it is essential to test whether a specified order is feasible. If a specific component requires the result of another component as an input then an error message should be displayed in cases where this input has not been computed yet. This is necessary to avoid using wrong or undefined values for such variables. MMEANS automatically checks the value of all variables every time they are accessed anywhere in the model. This is true for individual variables as well as sub-components of aggregates, such as individual monetary variables making up an aggregate income concept. If a variable that is accessed has not been computed yet, an error message is generated, indicating the variable concerned and the algorithm trying to access it.

- Changes of endogenous variables during the course of a model run: This tracing of variables is supported by permitting users to produce similar types of output at different stages during the simulation process (see sections 6.2 and 6.9).

6.7 Data management and manipulation

In addition to parameters related to the tax-benefit algorithms *per se*, a number of parameters relate to the micro-data on which the tax-benefit system is to be simulated. One of the desirable features of a microsimulation modelling framework is that it should be possible to add new variables with ease, to remove variables that are no longer needed, or to change their characteristics. The computer program itself should not need any alterations as a result of these changes, i.e., it should be data independent.

In MSMs based on MMEANS, all variables used in the model are specified in a list containing the variable names and additional information such as whether the variable is an individual variable or relates to the household as a whole, whether or not it is a monetary variable, or whether the value of the variable is simulated by the model or read from the input micro-data. During run-time, the model then automatically carries out all the procedures (declaration, initialisation, allocation of space in the relevant data structures, etc.) necessary to make the variable useable by the model as input or output of a tax-benefit calculation.

For cases where certain variables are not available in the micro-data underlying a simulation, default values can be specified. This is especially important in a multi-country context, where one may want to simulate the effects of introducing a tax-benefit instrument from country A in country B. If the tax-benefit rules of this instrument are specified as a function of a certain variable which is not available in country B's micro-data, then one can specify appropriate default-values for this variable. Default values can be specified either directly (i.e., by specifying an actual value) or by referring to a variable that *is* available in country B's micro-data and is considered a good approximation of the missing variable. For example, in a situation where the tax-benefit rules of country A require information on whether someone is a civil servant and where there is no civil servant variable in country B micro-data, one can specify that a variable "public sector" available in country B data should be used as a proxy for "civil servant".

6.8 Data updating; 'Ageing' the population

Frequently the data available for microsimulation are not from the same year as the year to which the policy scenario relates. This is because tax-benefit policy changes most years, while data is often only collected infrequently and usually involving a time-lag of at least one year. To still be as representative of the population as possible, adjustments will often be necessary.[20] The first type of adjustment relates to the value of monetary variables. They will generally have changed due to general and relative price changes during the period between data collection and the reference year relevant for the policy simulation.

Uprating monetary variables only implicitly assumes that the structure of the underlying population has remained unchanged. In many cases, this will not be realistic (Harding, 1996; Merz, 1991). The data may, for example, have been collected during a period of low unemployment, while the policy year we are interested in may be characterised by a much

[20] Adjustments may also be desirable for forecasting purposes. In this case, the underlying data would be adjusted to match *expected* changes in the population and in incomes. Data thus adjusted can then be used as input into the tax-benefit model to explore projected aggregate revenue/costs and or distributional features.

higher unemployment rate. Given sufficiently long gaps between data collection and simulation reference year, many demographic dimensions may have changed as well (age-profiles, fertility, female labour force participation, etc.).

As discussed above, there are basically two techniques for taking these changes into account. The static ageing approach seeks to adjust the weights in the data without actually altering any observations. By specifying conditions that have to be met by the re-weighted data-set ("restrictions") such as a set of aggregates and/or distributions that the data should reproduce, it is possible to find new weights that approximate the conditions that have been specified. However, no firm statements can be made about how representative the "new" data are about dimensions which have *not* been controlled for (particularly if there is little correlation with the control variables). To partly address this problem, techniques have been developed to re-weight in a "conservative" manner. Essentially, they amount to solving an optimisation problem where the specified restrictions have to be met while changing original weights in the data as little as possible (see, e.g., Merz, 1994).

Dynamic ageing methods, on the other hand, directly alter characteristics of existing observations as a function of past experience and/or time. Along with the age of individuals, other status variables, such as family status, labour force participation, will change as well (see Harding, 1990). Given changing ages of existing observations, it will generally also be necessary to simulate births and survival.[21] It is obvious, that the informational requirements for realistic simulations of this type are considerable.[22] In terms of policy analysis, it may often be preferable to use static ageing methods or even exclusively rely on indexation of monetary variables. While the resulting data will not be strictly representative of the policy year, one is at least aware of the source of this error. On a theoretical level, dynamic ageing is, of course, more appealing and can be very useful in exploring the consequences of various assumptions on the longer-term effects of policies.

In models based on MMEANS, the uprating of monetary variables is implemented via separate parameters for each monetary variable. Different incomes may increase at different rates and since these rates may themselves differ for different groups (for example employment income may increase at a different rate for males/females, civil servants etc.), we allow for differential uprating. External packages can be used to adjust weights (see, e.g., Merz, 1993). As discussed in section 4, tax-benefit models based on MMEANS can be linked into dynamic models.

6.9 Output

The principal output of a simulation run is a micro output file that can output any variable of the model. The micro output-file can then be used with statistical packages for performing more elaborate analyses. Conceptually, the micro-output component has been designed in exactly the same way as the Policies used in tax-benefit calculations. In particular, this means that the micro-output Policies need to be entered in the Spine just as any other Policy (see section 6.2).

[21] An exception are exercises that seek to simulate the life-time histories of a given cohort.
[22] Also, given its dynamic nature, any systematic errors will be compounded as the process is repeated over several periods.

Even though any statistical package can be used to analyse these micro-level outputs it is, for a number of reasons, desirable to have most of this analytical capability available within the model framework. Keeping track of numerous large micro-output files for many different simulation runs can be difficult and a source for errors. This is the case for any tax-benefit model since it is often necessary to formulate a large number of policy scenarios in order to explore the research question at hand. For a multi-country model, the number of output files is potentially much larger. In addition, the total sample size of the micro-data underlying a multi-country MSM can be very large. In the case of EUROMOD, these data represent more than 100,000 households containing more than a quarter of a million people. Any multi-country micro-output will therefore be of a similar size which may exceed the relevant limits of some commercially available software tools. Most importantly, analysing micro-output can be very time consuming. Many different analyses are possible and each of them entails a set of assumptions and definition which needs to be decided upon. As a result, it is convenient to have a standard 'summary output' which can reliably perform most of the desired analyses while keeping all the related choices and assumptions as transparent and accessible to the user as possible. In this way, it is possible to ensure consistent output across uses, users and countries. Users can rely on the 'summary output' routine to have been tested and to produce correct calculations that are robust and consistent across different applications.

The summary output has been embedded into the MMEANS framework in order to work effectively on model generated micro-output files. However, it can also be run separately and can thus be used as an analytical tool for analysing any micro-data file, whether generated by a tax-benefit model or not. The standard output is designed to:

- provide statistics and summary indicators that are accepted standards among researchers and policy analysts and that can be used in a consistent way across different countries and uses of the model;

- permit users to analyse the sensitivity of the various indicators by allowing them to vary underlying concepts and definitions such as exchange rates, poverty lines, equivalence scales, etc.;

- mirror the flexibility of models based on MMEANS by not imposing any *a priori* definition of concepts such as disposable income, a 'child', etc.;

- be able to handle the very large amounts of data resulting from the simulation of policy instruments for all households contained in micro-datasets of several countries;

- use parameter files that have a similar structure and layout as other parameter files that are part of MMEANS;

- attach a comprehensive description to the numerical output which clearly shows the kinds of choices made by the user of the output program. Given the multitude of possible definitions and concepts such 'labelling' is essential to ensure that the numbers produced by the output program are interpreted in an appropriate way;

- be computationally reliable and robust.

Currently, the Summary Output program supports the computation of quantile groups; inequality indicators; poverty indicators; summary statistics for any variable as well as gainers/losers and summary statistics of *changes* between two different scenarios. There is a separate module for defining various equivalence scales for comparing the income situations of households of different sizes and/or composition and a general purpose module for filtering cases in order to analyse certain sub-groups of the population.

Each of these computational algorithms is implemented as a separate component. Similar to the specification of the structure of tax-benefit systems in figure 1 above, it is possible to arrange the various components in different orders. This is useful because components can be combined by using the output of one computation as an input into later ones. This allows for the analysis of indicators and summary statistics for different sub-groups of the population which depend on variables that are endogenously computed within the Summary Output program. One can, for example, produce[23]

- average tax payments or benefit receipts for the poor/non-poor population;
- socio-economic characteristics of different groups of "gainers" and "losers" of a policy reform;
- inequality indices separately for the poor and non-poor population;
- redistribution measures for different income quantile groups.

Each specification of summary output can be archived and re-used for later analyses (similar to, say, SPSS or Stata syntax files). Since the specification of the summary output is *independent* of the underlying tax-benefit simulations, this system of archiving is particularly useful to ensure consistency in the underlying definition of summary indicators for different policy scenarios within or across countries. By running the same summary output specification on different micro output files, one can, for example, be sure to use the exact same set of equivalence scales, definition of poverty or inequality concepts or units of analysis.

7 Conclusion

We have described the rationale for microsimulation tax-benefit modelling and the demands placed on these models. We note the high cost of developing microsimulation models in different countries and argue for the need to control these costs. Using a generalised microsimulation modelling platform can result in substantial economies of scale. We introduce a new modelling platform (MMEANS) developed to aid in the construction of single- and multi-country tax-benefit models by providing all essential components and a system by which these can be parameterised and combined into a full model. Although the time taken to construct this general modelling framework has been considerable, these economies of scale have already become evident as the framework and its components have successfully been used to implement an integrated European tax-benefit model comprising the tax-benefit systems of all 15 EU countries.

[23] Details are provided in Immervoll (2002).

The principal design feature of MMEANS is the extent to which routines and operations have been generalised and parameterised and are thus re-usable for different purposes. Examples not typically found in national specific tax-benefit microsimulation models include the consistent parameterisation of

- the fiscal unit of analysis;
- the order in which instruments are simulated;
- income concepts; and
- the input database and related operations such as data updating.

The use of encapsulated model components makes resulting tax-benefit models flexible and robust by preventing unintended interactions between them. It also allows users to focus on those parts of the tax-benefit system which are of interest for the research question at hand while not having to worry about computational details of the modelling framework as a whole.

While a great number of conceptual and theoretical issues have to be addressed in any model construction project, the implementation of tax-benefit microsimulation models is characterised by large overheads in terms of the practicalities of setting up the complex computer environment on which the model can be based. These very large fixed costs may sometimes prevent the construction of models altogether or lead to situations where the potential of these models is inhibited by the degree to which model developers are occupied with the technical tasks of maintaining a model or getting it to run in the first place. Using a multi-purpose model building platform should enable researchers to spend more time addressing the many methodological and conceptual issues related to the specification and application of microsimulation models rather than having to "reinvent the wheel" by reproducing the essential building blocks of these models over and over again for each model that is being developed.

8 References

Atkinson, A.B., F. Bourguignon and P.A. Chiappori, 1988. "What do we learn about tax reform from international comparisons? France and Britain", *European Economic Review*, v32, pp 343-52.

Atkinson, A.B., F. Bourguignon, C. O'Donoghue, H. Sutherland and F. Utili, 2001. "Microsimulation of Social Policy in the European Union: Case Study of a European Minimum Pension", forthcoming *Economica*.

Bourguignon, F., C. O'Donoghue, J. Sastre-Descals, A. Spadaro and F. Utili, 1997. "Eur3: a Prototype European Tax-Benefit Model", Cambridge: *Microsimulation Unit Discussion Paper* No. 9703.

Callan, T. and H. Sutherland 1997. "The Impact of Comparable Policies in European Countries: Microsimulation Approaches", *European Economic Review Papers and Proceedings*.

Citro, C.F. and E.A. Hanushek (eds.), 1991a. *The uses of Microsimulation Modelling, Volume 1, Review and Recommendations*. Washington: National Academy Press.

Citro, C.F. and E.A. Hanushek (eds.), 1991b. *The uses of Microsimulation Modelling, Volume 2, Technical Papers*. Washington: National Academy Press.

Falkingham, J. and C. Lessof, 1992. "Playing God or LIFEMOD – The construction of a dynamic microsimulation model", in: H. Sutherland and R. Hancock (eds.) *Microsimulation models for public policy analysis*, STICERD, London School of Economics.

Giles, C. and J. McCrae, 1995. *The IFS micro-simulation tax and benefit model*, IFS Working Paper W95/19.

Harding, A., 1990. "Dynamic Microsimulation Models: Problems and Prospects", Welfare State Programme Discussion Paper WSP/48, STICERD, London School of Economics.

Harding, A., 1996. "Introduction and Overview", in: A. Harding (ed.) *Microsimulation and Public Policy*, Amsterdam: North Holland.

Hancock, R., 1997. "Computing strategy for a European tax-benefit model", Microsimulation Unit Discussion Paper MU9704, Department of Applied Economics, University of Cambridge.

Hoshka, P., 1986. 'Requisite research on methods and tools for microanalytic simulation models', in: G. Orcutt, J. Merz and H. Quinke (eds.) *Microanalytic Simulation Models to Support Social and Financial Policy*. Amsterdam: North-Holland.

Immervoll, H., C. O'Donoghue and H. Sutherland, 1999. "An Introduction to EUROMOD", *EUROMOD Working Paper* no. 0/99, Department of Applied Economics, University of Cambridge. Available through http://www.econ.cam.ac.uk/dae/mu/emod.htm.

Immervoll, H., 2000. "Fiscal Drag - An Automatic Stabiliser?", Paper presented at the 56th Congress of the International Institute of Public Finance, Seville, August 28-31.

Immervoll H., and C. O'Donoghue, 2001a. "Welfare Benefits and Work Incentives: The Distribution of Net Replacement Rates in Europe ", *EUROMOD Working Paper* no. 4/01, Department of Applied Economics, University of Cambridge. Available through http://www.econ.cam.ac.uk/dae/mu/emod.htm.

Immervoll, H., and C. O'Donoghue, 2001b. "Imputation of Gross Amounts from Net Incomes in Household Surveys: An Application using EUROMOD ", *EUROMOD Working Paper* no. 1/01, Department of Applied Economics, University of Cambridge. Available through http://www.econ.cam.ac.uk/dae/mu/emod.htm.

Immervoll, H. and C. O'Donoghue, 2001c. "EUROMOD, A European Tax-Benefit Model. Operational Guide", Microsimulation Unit, Department of Applied Economics, University of Cambridge.

Immervoll, H., H. Sutherland and K. de Vos, 2001a. "Reducing Child Poverty in the European Union: the Role of Child Benefits", in Vleminckx, K., Smeeding, T.M. (eds.) Child Well-Being, Child Poverty and Child Policy in Modern Nations, The Policy Press, Bristol.

Immervoll, H., F. Berger, M. Borsenberger, J. Lumen, B. Scholtus and K. de Vos, 2001b. "The impact of tax-benefit systems on low-income households in the Benelux countries. A simulation approach using synthetic datasets", *Schmollers Jahrbuch (Journal of Applied Social Science Studies)*, v121 n3, pp 313-52.

Immervoll, H., 2002. "Producing Output in EUROMOD", Microsimulation Unit, Department of Applied Economics, University of Cambridge.

Klevmarken, A., 1997. "Modelling Behavioural Response in EUROMOD", Working Paper No. 9720, Microsimulation Unit, Department of Applied Economics, University of Cambridge.

Klösgen, W., 1986. "Software implementation of microanalytic simulation models – state of the art and outlook", in: G. Orcutt, J. Merz and H. Quinke (eds.) *Microanalytic Simulation Models to Support Social and Financial Policy*. Amsterdam: North-Holland.

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

Mantovani, D., 2002, "Simulating policy reforms with EUROMOD – A case study", Microsimulation Unit, Department of Applied Economics, University of Cambridge.

McCrae, J., 1999. "The Development and Uses of Tax and Benefit Simulation Models", *Brazilian Electronic Journal of Economics*, v2 n1.

Merz, J., 1991. "Microsimulation- a survey of principles, developments and applications", *International Journal of Forecasting*, v7 n1, pp 77-104.

Merz, J., 1993. "ADJUST – A program package for the adjustment of micro data by the minimum information loss principle: Program Manual", FFB-Discussion Paper No. 1e, Forschungsinstitut Freie Berufe (FFB)), Department of Economics and Social Sciences, University of Lüneburg, Lünerburg, Germany.

Merz, J., 1994. "Micro data adjustment using the minimum information loss principle", FFB-Discussion Paper No. 10, Forschungsinstitut Freie Berufe (FFB)), Department of Economics and Social Sciences, University of Lüneburg, Lünerburg, Germany.

Merz, J., 1995. "MICSIM – Concept, Developments and Applications of a PC-Microsimulation Model for Research and Teaching", FFB-Discussion Paper No. 14, Forschungsinstitut Freie Berufe (FFB)), Department of Economics and Social Sciences, University of Lüneburg, Lünerburg, Germany.

Mot, E. S. , 1992. *Survey of Microsimulation Models*, Social Security Research Committee, The Hague: VUGA.

O'Donoghue, C. and H. Sutherland, 1999. "Accounting for the family in European income tax systems", *Cambridge Journal of Economics*, v23 n5.

O'Donoghue, C., 2001a, "Dynamic Microsimulation: A Methodological Survey", *Brazilian Journal of Economics*, v4 n2. Available through www.beje.decon.ufpe.br.

O'Donoghue, C., 2001b, *Redistribution in the Irish Tax-Benefit System*, unpublished PhD thesis, London School of Economics.

Orcutt, G., 1957. “A new type of socio-economic system”, *Review of Economics and Statistics*, v58, pp 773-97.

Orcutt, G., S. Caldwell and R. Wertheimer, 1976. *Policy Exploration through microanalytic simulation*, Washington DC: The Urban Institute.

Orcutt, G., 1986. ”Views on microanalytic simulation modelling”, in: G. Orcutt, J. Merz and H. Quinke (eds.) *Microanalytic Simulation Models to Support Social and Financial Policy*. Amsterdam: North-Holland.

Pylkkänen, E., 2001. ”Modelling wages and hours of work”, *Brazilian Journal of Economics*, v4 n2. Available through www.beje.decon.ufpe.br.

Piachaud, D. and H. Sutherland, 2000. “How Effective is the British Government's Attempt to Reduce Child Poverty?” CASE paper 38, CASE, London School of Economics.

Redmond, G., H. Sutherland and M. Wilson, 1998. *The arithmetic of tax and social security reform: a user’s guide to microsimulation methods and analysis*, Cambridge: CUP

Schofield, D., 1995. “Designing a user interface for a microsimulation model”, *STINMOD technical paper* no 7. National Centre for Social and Economic Modelling, Faculty of Management, University of Canberra.

Sutherland, H., 1991. “Constructing a tax-benefit model: what advice can one give?”, *Review of Income and Wealth*, v37 n2, pp. 199-220.

Sutherland, H., 1995. “Static Microsimulation Models in Europe: A Survey”, Cambridge: Microsimulation Unit Discussion Paper, MU9503.

Sutherland, H. (ed.), 2001. “Final Report. EUROMOD: An integrated European benefit-tax model”, EUROMOD Working Paper no. 9/01, Department of Applied Economics, University of Cambridge. Available through http://www.econ.cam.ac.uk/dae /mu/emod.htm.

Weinberg, D.H., 1999, "Income Data Collection in International Household Surveys", Paper presented at the 3rd Meeting of the Canberra Group in Ottawa, Canada, June 7-9.

Figure 1. General Structure of Tax-Benefit Systems.

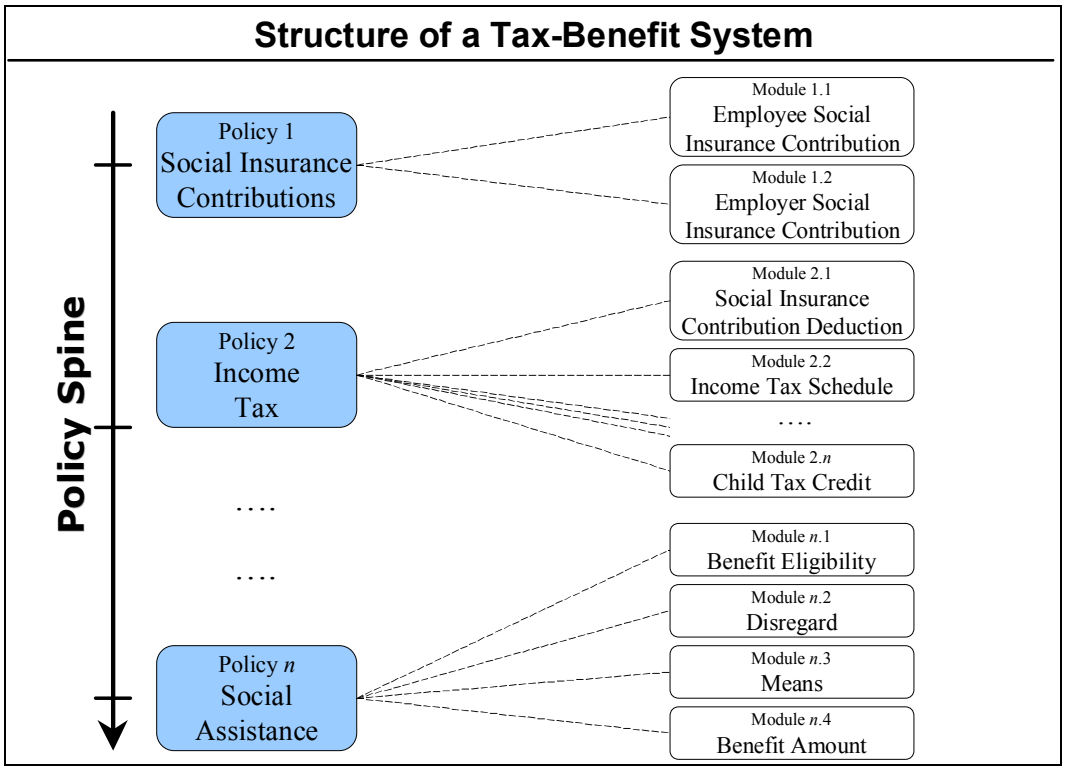


Figure 2. Basic Components of MMEANS.

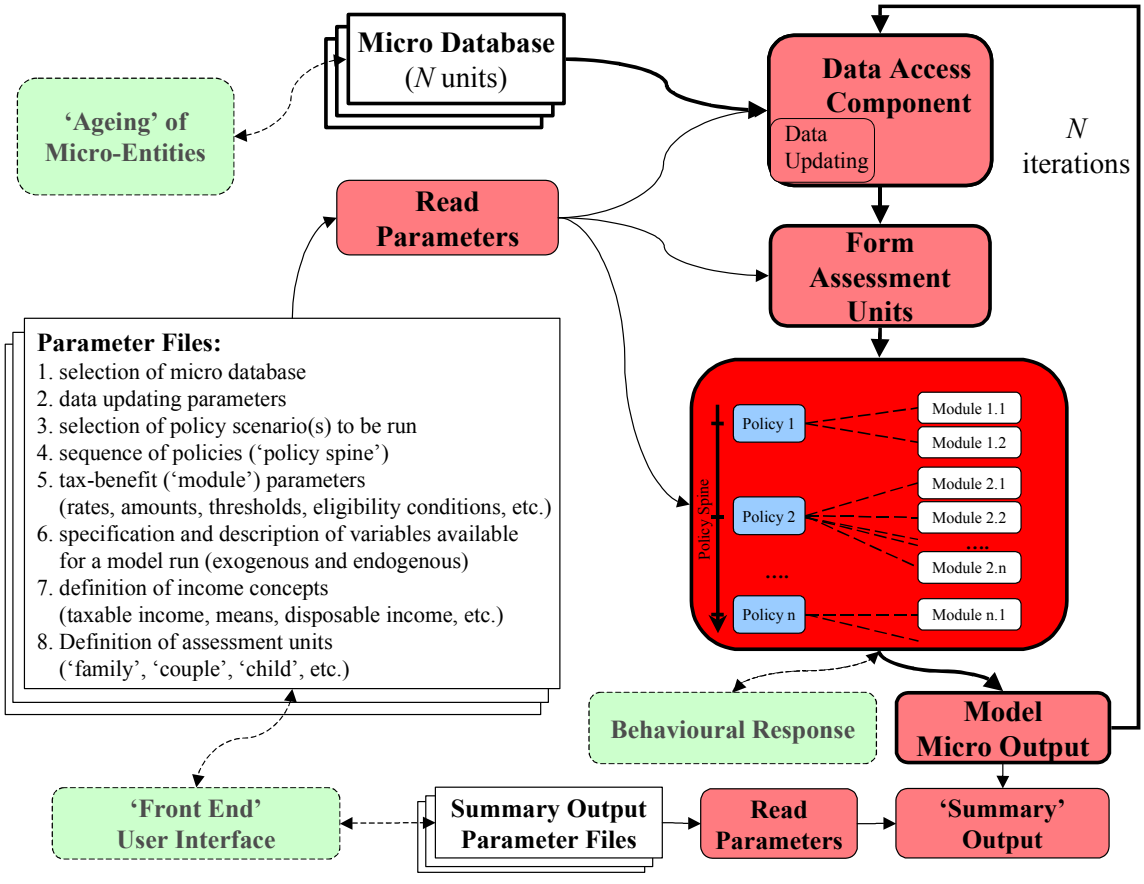
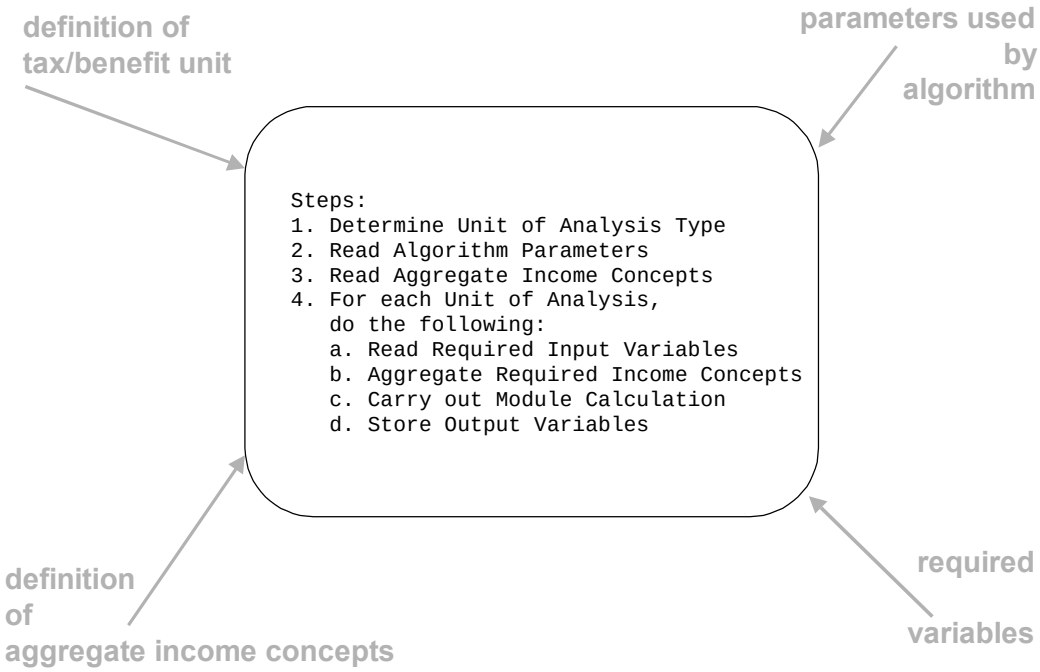


Figure 3. Architecture of Modules.



Appendix. Fiscal Unit Calculations.

In each household, there may be one or more instances of a fiscal unit *type*. Fiscal units are instances of a certain type (a household may, for example, contain two units of type "Married Couple"). For each fiscal unit type, each person in the household receives a number indicating the fiscal unit (of this type) they belong to. Using the conditions mentioned above, it is possible to decide for each person, whether or not they are member of a specific fiscal unit.[24] A fiscal unit can be fully or partly occupied so that if the fiscal unit type is "Married Couple" then in a one-person household, this one person is a fiscal unit of this type even though there is no spouse. Persons who are not assigned to a fiscal unit together with other persons form their own fiscal unit.

Pseudo Code of Fiscal Unit Calculations for a household HH:

```
For all fiscal unit types do the following:
{
        IF fiscal unit type="household" THEN put all persons in HH into same fiscal unit number
        IF fiscal unit type="individual" THEN put all person in HH into different fiscal unit number
        IF fiscal unit type=something else THEN do CALC_FAMILY_TU
}
```

Routine to calculate specific fiscal units (CALC_FAMILY_TU):

```
For all fiscal units (maximum is number of persons in household) do the following:
{
    For all persons in HH do the following:
    {
        //assign first person to fiscal unit:
        IF ( person is adult OR if household has no adults ) AND
           person has not been assigned to a fiscal unit of the current fiscal unit type AND
           no other person has been assigned to the current fiscal unit THEN
                assign current person to current fiscal unit

        //assign spouse of first person:
        IF spouses are part of fiscal unit type AND
           person is married to first person of fiscal unit THEN
                assign person to fiscal unit

        //assign cohabiting partner of first person:
         IF cohabiting partners are part of fiscal unit type AND
            person is cohabiting partner of first person of fiscal unit THEN
                assign person to fiscal unit

        //assign elderly dependants of first person:
         IF elderly dependants are part of fiscal unit type AND
             person is elderly dependent of first person of fiscal unit THEN
                assign person to fiscal unit

        //assign children of first person:
```

[24] A person can be member of more than one fiscal unit simultaneously (e.g., 'individual', 'married couple' and 'household') but he/she can only be member of one fiscal unit of a given fiscal unit type (e.g., if two married couples live in the same household, each person will of course only be member of one 'married couple' unit).

```
        IF children are part of fiscal unit type AND
            person is child AND
            person is child of first person of fiscal unit THEN
                assign person to fiscal unit
    }
}
```