Impact Project

Impact Centre
The University of Melbourne
153 Barry Street, Carlton
Vic. 3053 Australia
Phones: (03) 344 7417
Telex: AA 35185 UNIMEL
Telegrams: UNIMELB, Parkville

IMPACT is an economic and demographic research project conducted by Commonwealth Government agencies in association with the Faculty of Economics and Commerce at The University of Melbourne, the Faculty of Economics and Commerce and the Departments of Economics in the Research Schools at the Australian National University and the School of Economics at La Trobe University.

INFLUENCES OF THE APPLICATION ENVIRONMENT ON THE DEVELOPMENT OF SOFTWARE FOR LARGE ECONOMIC MODELS

by

George Codsi
IMPACT Research Centre

and

Ken Pearson
Mathematics Department
La Trobe University

Preliminary Working Paper No. IP-33

Melbourne July 1987

ISSN 0813 - 7986
ISBN 0 642 10168 X

Revised version of a paper presented at the Australian
Software Engineering Conference, Canberra, A.C.T.,
13th - 15th May 1987

REFERENCES

ANSI (1978), *American National Standard Programming Language FORTRAN X3.9-1978*, New York, American National Standards Institute.

Bisschop, Johannes and Alexander Meeraus (1982), 'On the Development of General Algebraic Modeling System in a Strategic Planning Environment', *Mathematical Programming Study*, Vol.20, pp.1-19.

Codsi, G. and Pearson, K.R. (1986), GEMPACK Documents numbered GED-2, GED-3, GED-5 up to GED-8, and GED-11 up to GED-18, Impact Project, approximately 700 pages in total.

Dixon, P.B., Parmenter, B.R., Sutton, J. and Vincent, D.P. (1982), *ORANI: A Multisectoral Model of the Australian Economy*, North-Holland, Amsterdam.

Drud, Arne, David Kendrick and Alexander Meeraus (1986), 'HERCULES: A System for Development of Multisectoral Economywide Models', *World Bank Discussion Paper* No. DRD169, pp.12, April.

Pearson, K.R. (1986), 'Automating the Computation of Solutions of Large Economic Models', Impact Project *Preliminary Working Paper* No. IP-27, University of Melbourne (March), pp.28.

Rutherford, Thomas Fox (1985), 'MPS/GE User's Guide', Department of Operations Research, Stanford University, pp.72.

CONTENTS

ABSTRACT

This paper identifies several important aspects of the environment in which the GEMPACK software system for economic modelling is evolving, and describes how each of these aspects has affected the development and use of the software.

4 CONCLUSION

In order to cope with a wide variety of users (of varied experience) in a complex and changing user environment, the GEMPACK system has been designed to be modified or expanded easily and reliably, while remaining always portable and forming a total, integrated software package.

Although we were motivated by the grass-roots needs of our application environment and not directly by abstract software engineering principles, the development of GEMPACK has nevertheless resulted in a software package that has many of the properties that 'good' software engineering techniques aim for: modularity, the integration of all the separate parts, a hierarchical structure, well-defined interfaces betweeen the different functional parts and, of primary importance, the satisfaction of the users.

subroutines or in the FORTRAN PARAMETER declarations of <u>main</u> programs (which are clearly identifiable and of which there are a limited number). When GEMPACK is installed at a new site, the site is provided with detailed instructions about where source code changes may be required and how to determine if a modification is required. Special programs to test the potentially machine-dependent routines are provided.

The use of a standardised programming language, that is, ANSI standard FORTRAN, has helped in overcoming those portability problems associated with having different versions of a language on different machines. Nevertheless, there have been problems where, despite the manufacturer's claims, a particular FORTRAN compiler does not conform to the ANSI FORTRAN standards.

One disadvantage of needing to provide portable software is that we are unable to provide GEMPACK users with sophisticated terminal interfaces. Currently, we can only give them what is available through standard FORTRAN input and output. If we did not have to port our software to a wide variety of computing environments and if we had greater resources, we could provide selected sites and selected terminals with simple and easy-to-use screen-based input interfaces. In the future, this problem can be overcome by incorporating a well-defined user interface into GEMPACK software. A standard, portable input/output module (looking much like the current user interface) can be provided, but additionally different sites will be able to write their own customised input/output modules to replace the standard one.

INFLUENCES OF THE APPLICATION ENVIRONMENT ON THE DEVELOPMENT OF SOFTWARE FOR LARGE ECONOMIC MODELS

by

George Codsi and Ken Pearson

1 INTRODUCTION

Economists have been using large economic models for policy evaluation for more than two decades. One field of modelling, known as computable general equilibrium (CGE) analysis, is the fastest growing area in applied economics. The "computable" in the name of this research area makes it clear that these models could not be used without computers to do the calculations. Until recently, each different model was implemented by means of specially written code. This meant that

- o there was a huge investment made to implement each model;
- o changes (even small ones) were difficult to make;
- o the continuing support of each model was often critically dependent on the one (or few) programmer(s) who wrote the code;

and

- reliability was questionable since, because the code was often specific to one machine and not fully documented, the results of the model could not be verified by independent research groups.

Recently some general-purpose software systems have been developed. GEMPACK, described in Codsi and Pearson (1986) and Pearson (1986), is one; others are GAMS and HERCULES (which were developed at the World Bank -- see Bisschop and Meeraus (1982) and Drud, Kendrick and Meeraus (1986)), and Rutherford's MPS/GE (Rutherford (1985)). Once the economic basis of a model is fully worked out, these systems can be used relatively quickly by economists to solve and/or to modify the model without the need to write much, if any, one-off software. These software packages hence enable economists to concentrate on the economic (rather than the computing) aspects of their models.

GEMPACK (<u>G</u>eneral <u>E</u>quilibrium <u>M</u>odel <u>PACK</u>age) is a growing suite of software being developed under the auspices of the Impact Project, which is a research endeavor supported by six Australian Federal Government agencies and three Australian universities. It is installed at several different locations in Australia and is regularly used to solve one of the world's largest economic models, ORANI (Dixon, Parmenter, Sutton and Vincent (1982)). At present GEMPACK consists of 3 main parts (one for <u>data handling</u>, one for <u>model implementation</u> and one for carrying out <u>simulations</u> with models), each of which consists of a number of programs and support modules.

In section 2 we describe the environment in which GEMPACK is used and developed, identifying those aspects which have affected GEMPACK. We detail these influences in section 3 and briefly summarize our experience in section 4.

needed by a subroutine, then the PARAMETER value is passed to that subroutine in its calling sequence. (The PARAMETER name is normally also passed to the subroutine, so that if the PARAMETER value is found to be too small for the model, the error system can report the PARAMETER name to the user.)

2. Passing data to subroutines in FORTRAN

   There is almost no FORTRAN COMMON used in GEMPACK software. Rather, variable values are passed to subroutines by explicit inclusion in the calling sequence. Economists who want to use GEMPACK subroutines for their own applications need not fear inadvertently failing to initialise a variable in COMMON. The non-use of COMMON has the important advantage that users know exactly what variables need to be defined in order to call a subroutine.

Since the economic models handled by the GEMPACK software system can be enormous, they have associated with them extremely large data bases. In order to reduce the amount of memory needed, these data bases are stored routinely in a compressed, binary form. This means that GEMPACK software must access these data bases via special subroutines, and must also provide a variety of utility programs that allow the data bases to be examined and modified by economists.

3.5 Portability

As noted above, resource limitations preclude our attempting to customise GEMPACK for different machines or computing environments. Rather GEMPACK has been designed and implemented so that <u>all</u> site and machine dependent aspects occur either in a small number of low-level

Despite its restricted set of available data types, FORTRAN has nevertheless ably served as the high-level programming language for GEMPACK. We believe that successful software development is more influenced by good design than by the choice of a particular high-level language. While other languages may be more conducive to the production of 'good' source code, FORTRAN presents no great difficulties so long as care is taken with the design of new software, and certain sensible programming techniques are employed. Two examples will serve to illustrate how potential problems in the use of FORTRAN can be overcome.

1. Lack of dynamic memory allocation in FORTRAN.

    This is a very important aspect of GEMPACK implementation, because of the very large economic models that GEMPACK must be able to handle.

    Most parts or subpackages within GEMPACK consist of a single main program and specialised subroutines, together with other general GEMPACK subroutines from the GEMPACK library. The main program defines (via FORTRAN 'PARAMETER' statements) a number of parameters each of which usually determines the size of one or more working FORTRAN arrays, thereby effectively limiting the size of the economic model that can be treated. The values of these parameters can be increased (the program must then be re-compiled and re-loaded) to handle larger models. When a model is too large, the GEMPACK error system is triggered. It reports the name of the FORTRAN PARAMETER that needs to be increased and the minimum size it needs to be set to. The user need only change this parameter value <u>once</u> in the source code for the main program. If the PARAMETER value is

2 THE GEMPACK ENVIRONMENT

2.1 Economic Background

Computable general equilibrium models are used for policy evaluation ("How much <u>difference</u> would it make if the government were to increase tariffs by 20 per cent?") and for forecasting ("What <u>will</u> happen to the economy if the government increases tariffs by 20 per cent?"). Each model has a detailed theoretical basis (equations modelling different parts of the economy and their interrelationships) and a body of data (often very large). Some economists (model developers) build models which are in turn used by other economists (model users) to carry out policy simulations or make forecasts. Until recently the hands-on use of these models was restricted to economists at the forefront of research; but now their use is spreading, even to well trained undergraduates in the later years of their courses. (The availability of general-purpose software is one reason for this trend.)

2.2 The Tasks Involved in Developing and Using a CGE Model

The life of a computable economic model can be conveniently split into two stages:

o the building of the model; that is, the development of the theory, equations and data of the model and the implementation of the model on a computer;

- o the use of the model by economists for simulations, policy analysis and forecasting.

Below are listed various tasks in the design, implementation and the solution, via Johansen's linearisation method, of any large model. The building of a model commences at task 1, while the use of the model begins at task 6.

1. Developing the theory underlying the model and formulating the equations of the model in non-linear form.
2. Selecting a suitable linear approximation for the set of non-linear equations.
3. Reducing the size of the model, both by eliminating some variables (by substitution) and by absorbing others (into condensed variables). This task is not always required.
4. Setting up a structural definition of the (possibly reduced) model.
5. Providing a suitable data base for the model.
6. Solving the model.
7. Printing the results.

New models can be built either from scratch or by making major or minor changes to an existing model. When basing a new model on an existing one, much of the output of tasks 1 to 5 that was generated when building the original model can be re-used, thus reducing significantly the amount of work needed to implement the new model. Because of this

One special area of error handling involves notifying the user when a model is larger than an executing GEMPACK program is dimensioned to handle. The error-handling system stops the program, but not before giving detailed instructions to the user about how to fix the error by making minor and well-defined changes to the source code, followed by re-compilation and re-loading of the program. Details of this facility are given in the next section.

3.4 Implementation of GEMPACK Software

GEMPACK is written almost entirely in ANSI standard FORTRAN 77 (ANSI (1978)). We use FORTRAN for largely historical reasons, it being the high-level programming language mostly used by economists before GEMPACK came onto the scene. It is not expected that we will change to a different high-level language for a number of reasons :

- o FORTRAN has a defined standard which aids portability of GEMPACK software;
- o a substantial number of FORTRAN programs already exist that will continue to need support;

and

- o probably most economists will continue using FORTRAN when they still need to write their programs.

For these reasons, GEMPACK developers are obliged to provide FORTRAN support, both generally and in the form of a library of specialised FORTRAN subroutines. As there are not enough resources to provide general support for another high-level language, FORTRAN is also used for most of the GEMPACK software.

Also stressed in GEMPACK development is a robust error-handling system which is designed to trap most (hopefully, all) conceivable errors and to allow recovery from them whenever possible. Errors that occur when running GEMPACK software fall into two groups:

(i) Syntax errors.

By far the most common source of errors is invalid input caused by such factors as user inexperience with terminals, user unfamiliarity with the input conventions, invalid file names, incorrectly sequenced input and so on.

(ii) Substantive errors.

A smaller number of errors result from input which is economically invalid, which may lead to insolubility of the model. This may be caused, for example, by not selecting enough variables to take user-defined fixed values (in which case the number of linearly independent equations is less than the number of variables -- underdetermination), or by setting up an economic environment which leads to inconsistencies between related variables (overdetermination).

Since the users have different levels of experience in both the particular economic approach used and in the use of computers generally, no assumptions can be made about how 'sensible' a user's input will be. As well, many economists using GEMPACK are unable to consult anyone with either detailed GEMPACK knowledge or even some general software experience. Because of the design of the GEMPACK error system, users are informed why errors occur and are given detailed advice on how to recover from them.

possible relationship between development tasks for different models (especially when working with large models), the development and use of a suite of related models can be rather complex from both economic and computational points of view.

2.3 The Users of GEMPACK

GEMPACK users have widely differing backgrounds and experience in economic modelling techniques and in computer usage. Some are engaged in research in tertiary institutions, others are in government agencies providing policy advice. Some users prefer to write their own programs while many (perhaps most) do so only if there is no alternative. As GEMPACK continues to provide more aids in both the computational and the managerial aspects of modelling, the number and variety of users can be expected to increase. As well, the users are likely to want to harness GEMPACK for increasingly sophisticated applications.

2.4 The Applications

Economic model builders have differing views about what is the best type of model. Some develop smaller, special-purpose models, while others prefer larger, multi-purpose models. The size of the model affects how it can be solved. GEMPACK was developed with a view to handling very large models (tens and hundreds of thousands of equations) as well as smaller ones, solving them by a closed-form linear approximation technique. GAMS, HERCULES and MPS/GE are aimed at smaller, more special-purpose models, solving them by iterative techniques.

2.5 The Amount of General-Purpose Software Available

Originally all parts of the modelling process had to be done with one-off software. With each addition of general-purpose software to GEMPACK, the required amount of such one-off code decreases. To help in the production of any one-off code that is still written, and to ensure that such code can interface with the existing general-purpose code, GEMPACK provides a publicly available and documented library of software modules (Codsi and Pearson (1986)).

The first parts of GEMPACK to be developed automated the <u>use</u> of existing models for policy analysis (tasks 6 and 7), but not their development. Currently some model development tasks (for example, defining the structure of the model in task 4) can be handled by GEMPACK and we have prototype software for the major remaining one (task 3).

2.6 The Computing Environments

The computers which are accessible to the various economists wishing to use GEMPACK inevitably are different. For this reason a major requirement in its development has been its portability to as many computing environments as possible. Currently GEMPACK runs under the following configurations:

VAX/VMS, Prime/Primos, Pyramid/Unix, NAS/VM, Amdahl/Unix, IBM PC/DOS.

2.7 Software Development and Support Personnel

Groups using GEMPACK vary in size from one or two economists (having only general system support) to a government agency with a full-scale ADP department. Often users with little previous software experience have to install GEMPACK themselves, starting from scratch with a tape containing source code and data files.

new ways of accomplishing these tasks. Hence, the statement and analysis of the detailed requirements for a new GEMPACK addition typically require minimal effort since the general-purpose approach normally will mimic the way in which that task would be done via one-off software.

3.3 Design of GEMPACK

An extensive collection of pre-GEMPACK software (of varying quality) and of large data files (in various storage formats) exists. This influences how new parts of GEMPACK are designed. In particular, GEMPACK needs to interwork with some of this software and these data files until such time as they are either replaced or no longer needed.

An important aspect in GEMPACK design is the attractiveness of the user interface. Some economists are reluctant to change their ways when switching from one-off special-purpose code to new general-purpose software. A user interface is more likely to be attractive to them if it is simple (in design and appearance) and familiar (in terms of how it is set out, the order in which it requests information and the form in which the information is entered). Newer parts of GEMPACK incorporate switches to provide appropriate communication with users dependent on both their experience in using GEMPACK and the mode of use. Since a given modelling task often requires user input throughout it (and not just at the beginning during some well-defined 'input' phase), the design of user interfaces to GEMPACK software requires a certain degree of foresight and, given the inevitability of frequent modifications, must concentrate on the structure rather than the details of the user interface.

user must be protected);

- o demands for modifications of models are more common than for the construction from scratch of new ones;
- o building a model from scratch can require large amounts of (one-off and non-reusable) code;

and

- o from an existing model, many other models can be generated easily and quickly by minimal (one-off and non-reusable) coding.

A reasonable allocation of resources in the light of these points is initially to restrict the general-purpose code to handle the building of new models from scratch.

Another common trade-off is the continued partial use of existing, inferior code, versus waiting to make a complete switch-over. The wait might be occasioned either because the new general-purpose code is not yet completed, or because the user is able to adapt only slowly. Whenever we make available new general-purpose software, we also provide and document additional interfaces to that software. These allow different possible transition points from one-off code to the corresponding general-purpose code, and permit economists to use, gradually, more parts of the new software as they become confident with it.

While the use of general-purpose software in the development of economic models certainly streamlines and integrates the tasks that must be performed, and provides standard interfaces between them, most of our experience to date suggests that it does not open up any substantially

GEMPACK has been developed in a university environment with at most two people (one an academic with teaching duties) giving it their full attention at any one time. Within these constraints, there are no resources to customise GEMPACK for different configurations of hardware and/or operating systems.

3 THE INFLUENCE OF THE ENVIRONMENT ON THE DEVELOPMENT AND USE OF GEMPACK

The different parts of the GEMPACK environment have each influenced its development. In discussing these influences in this section, we focus on the following topics:

1. The overall evolution of the GEMPACK software system.
2. The requirements statement and review of additions to GEMPACK.
3. The design of GEMPACK additions.
4. The implementation of GEMPACK additions.
5. The portability of the entire GEMPACK software system to different computing environments.

3.1 General Influences on the Evolution of GEMPACK

The GEMPACK software system has evolved and will continue to evolve under two main influences: (A) the priorities assigned to automating each of the tasks listed in Section 2; and (B) feedback from users.

Specifically:

A. Decisions as to which tasks should be brought within the scope of GEMPACK, and the priority associated with each one, determine the order in which new parts of GEMPACK are developed.

In the period since general-purpose GEMPACK software first became available to model users, subsequent additions have been directed to a broader set of longer-term objectives, including:

- o the identification of design and management principles which will allow GEMPACK to evolve flexibly (for example, one should not cut off options for future developemnt in directions which can only be perceived vaguely at present);
- o the needs of individual potential users of different types;
- o the best means of introducing each successive addition to GEMPACK to the user community, for example, by providing a complete first version or by involving users in testing initial prototypes;
- o maintaining the portability of the software;

and

- o upward and downward compatibility between different parts of GEMPACK.

B. Feedback received from users about the various parts and sub-packages of GEMPACK often results in the need for new versions of parts of GEMPACK to extend their scope, to fine-tune them, or to improve their user interfaces.

3.2 Choosing Directions for the Evolution of GEMPACK

Frequent periodic reviews of GEMPACK's evolution result in decisions about what new facilities will be added. These reviews consider a wide variety of issues associated with CGE modelling, including the economic uses of the prospective addition, the feasibility of implementing it and the expected uses to be made of it by GEMPACK users.

One way of dealing with the demand for a new facility is to consider initially only a part of it. From the software developers' point of view, the selection of a sub-task means that there is no need to cope at the outset with the task's full complexity. The sub-task identified forms a suitable focus for an initial software <u>prototype</u>. Once this developmental version has been completed and used by economists, the provisional decision to add a production version to GEMPACK can be re-examined.

Consider the alternative tasks that may face a model developer, namely:

o building a model from scratch,

or

o modifying an already existing model.

Suppose we want to provide <u>general-purpose</u> software to aid the model developer. Some relevant considerations are:

- o it is nearly always much harder to build a model by modifying the software of an existing model, than by building it from scratch (this is because modifications to an already complex model often result in unforseen repercussions, against which the