APPROVING SOFTWARE PRODUCTS

Proceedings of the IFIP WG 5.4 Working Conference on
Approving Software Products (ASP-90)
Garmisch-Partenkirchen, F.R.G., 17–19 September, 1990

Edited by

WOLFGANG EHRENBERGER
Gesellschaft für Reaktorsicherheit Forschungsgelände
Garching, F.R.G.

1990

NORTH-HOLLAND
AMSTERDAM · NEW YORK · OXFORD · TOKYO

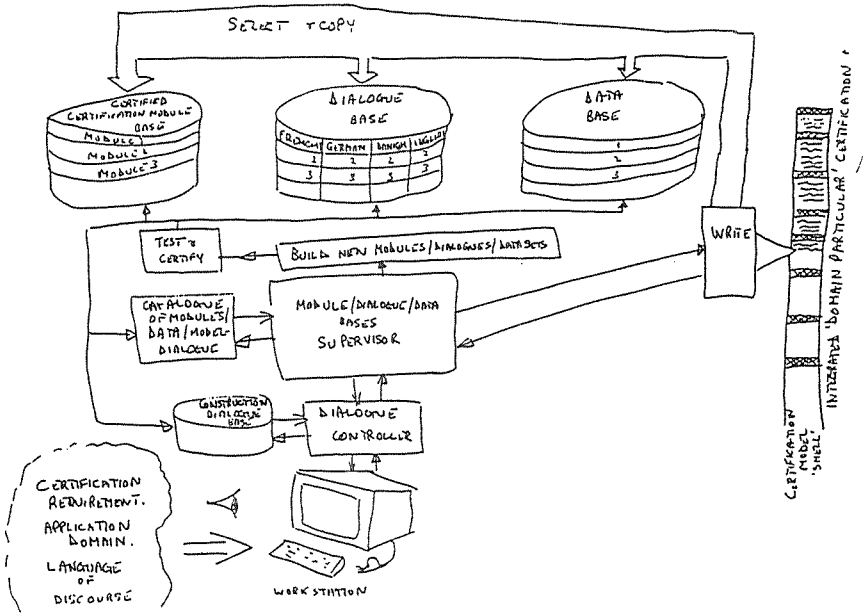

Fig. 3

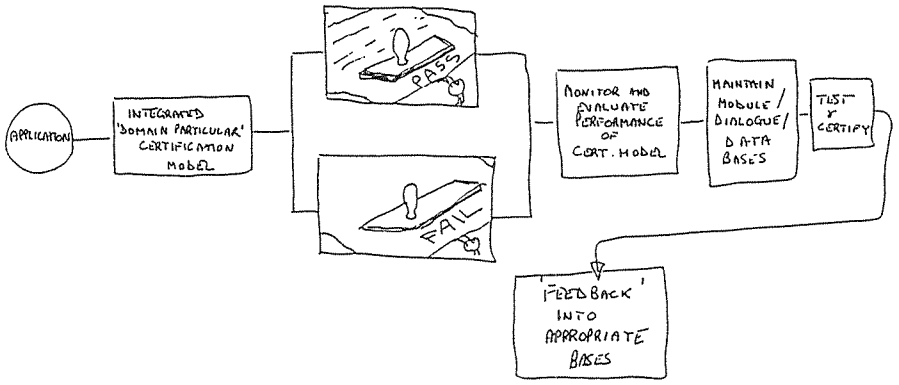

Fig. 4

Approving Software Products
W. Ehrenberger (Editor)
Elsevier Science Publishers B.V. (North-Holland)
IFIP, 1990

SOFTWARE CERTIFICATION BY DEMAND: AN EXPERIENCE FROM A THIRD-PARTY LABORATORY

A. BERTOLINO, C. CARLESI, M. FUSANI, V. LAMI

Istituto di Elaborazione dell'Informazione, CNR, Pisa, Italy

ABSTRACT

There is still a sort of ambiguity surrounding the expression "software certification". The dated, common-sense IEEE definition [IEEE/729] "a written guarantee that a system or computer program complies with its specified requirements" does not provide useful guidance for organizing an effective certification service. Besides, a certificate issued according to this interpretation is inevitably mistaken for a "guarantee" of product adequacy, that which can definitely not be risked on at the state of the art.

As an independent laboratory which has been offering this kind of service for some years, on request from Public Administrations, users and producers, we have been becoming more and more aware of this. Thus, we have conceived and applied a process in which a defined set of characteristics of both software product and developing project undergo a defined Validation Suite. This process ends with a formal declaration illustrating the actions performed and their results.

This is what we mean by software certification.

The Validation Suite must be designed for a class of product criticality and may be adjusted for each application field.

In practice, our laboratory had to perform both the tasks of putting together Validation Suites for a number of cases and of carrying out the certification processes. If we would refer our work to some proposed certification schemes (like the European Community's), this would not be exactly the role of a *certification body*, which is not supposed to work directly in definition of standards. However, lack of established, commonly agreed rules, along with pressure from the demand side, convinced the laboratory to take on both aspects of the job.

The paper describes some critical aspects faced by the laboratory during the preparation and application of the Validation Suite to actual software products certification.

In particular, the concept of certifiability (that determines the feasibility of applying a Validation Suite leading to a positive certificate) is added to the yet conspicuous set of -ilities qualifying a product.

Techniques used in the Validation Suite are picked up from defined standards and recommendations for designing "good" software, and provide a way to deal with imperfect user specifications. Such techniques include inspection, testing and tracing specifications to code and vice-versa.

Problems rising from the peculiar experience of independent certification, such as dealing with manufacturer's privacy and interfacing software products to external inspectors, are also presented and a trace for their solution is proposed.

1. BASIC CONCEPTS

1.1. Foreword

In this paper we refer to the problem of approving general purpose software products, i. e., those for which it is not possible to devise functional standards.

When functional standards can be produced (compilers, communication protocols, graphic kernels), then it becomes possible to derive standard test suites which enable an external organization to approve or not approve the products, independent of the development process.

When general purpose applications are to be examined, we claim instead that the development process must be considered in conjunction with the final product in order to be able to express any evaluation (see [Copigneaux88] for a slightly different view). Thus, throughout the paper, relevant distinction between software products and their development process is seldom made.

1.2. Definitions

Nowadays, terms such as "Verification and Validation" (V&V) and "Certification" are largely used within the software industry, in spite of scarce understanding (or rather, wide misunderstanding) about their meaning.

In this paper, by touching on some of the related concepts, we do not pretend to enlighten people about them (basically, there is little to understand: it is a matter of achieving common agreement on the concepts); we only think it is useful to discuss some of our experiences.

According to an authoritative source such as ANSI/IEEE standard [IEEE/729], software certification is "a written guarantee that a system or computer program complies with its specified requirements".

Let us comment shortly this definition. First of all, a statement like "... complies with ..." looks somewhat drastic, a sort of all-or-nothing declaration, useful for, perhaps, some well defined technologies that depend on Nature's immutable laws. So far, software does not seem to have reached such a level of universality. Thus, we suggest replacing the expression with an ambiguous (why not? it fits the idea) "how well". Or, if we are not too afraid to be committed to a search for quantifiable

concepts, we could better say "how much" and leave other details to further work in metrics.

If we continue to explore the above definition, we find another critical point: validation must be conducted against "specified requirements". What one would naturally argue is that a system is always created in order to satisfy a set of "user's needs". Thus, validating the user's needs would sound more reasonable and understandable. Only, this would not help in establishing a validation policy. In fact, the user's needs are hardly known by the user himself when a project starts, and will not be much better known when the product is shipped, (granting that it will be). So, the quoted expression "specified requirements" fits better from the operating point of view. Unfortunately, the open question of whether the specified requirements reflect the user's needs remains.

When the process of V&V is conducted by [IEEE/729] "an organization that is both technically and managerially separate from the organization responsible for developing the product", then we speak of IV&V. Thus, the remaining part of the paper will chiefly concern IV&V.

1.3. The demand for IV&V

In spite of the fact that people do not exactly agree on what software validation is, there is a considerable demand for it.
The following comments do not pretend to be an analytical survey of this demand, but directly result from the experience of our IV&V Unit (this will be described in the next section).

Our experience tells us that demand derives chiefly from the user's world, where certification is mistaken for a kind of guarantee. Thus, what is requested is just the end product of the validation, that is a **certificate** declaring (hopefully) that "the product complies with the user's needs". If this opinion is shared by others, as we assume, it is small wonder that some organizations refuse to issue certificates regarding software products.

Manufacturers may also be willing to have their software validated, at least for a couple of reasons:

1) To add value to the product. In fact, products accompanied by a validation statement (here, certificate) would make better sellers than those with no certificate at all.

2) To add value to the process. Obviously, manufacturers could take great advantage by submitting their production apparatus to analyses and tests performed by an independent organization.

However, we expect manufacturers are aware of the limits outlined above in validating user's needs, so they could ask for taking more realistically the "specified requirements" as the comparing object.

On the other hand, manufacturers are concerned with the extra-cost implied by an independent validation. Thus, many of them (unfortunately, the bigger companies, which mostly influence the market) fiercely resist any external attempt of performing validation or suggesting methods for validation, because the proposed actions would not be in harmony with their internal development processes, in which money and efforts are invested. Apart from this, some claim that formal documentation requested by external organizations unavoidably introduces undesired, awkward bureaucracy.

Some requests for IV&V comes from Public Administration. It is a known fact, for example, that the U.S. Department Of Defense (DOD) asks for formal IV&V declarations. Such declarations may in fact be conceived and executed according to the definition given in Section 6, where the validation procedure is requested to follow a defined standard that may be, for instance, the [DOD/2167].

We are also aware of a request for IV&V from the Italian Administration, as we have been involved for some years in the job: that is the history of the so-called "Fiscal Meters" certification [Bertolino87]. The initial request from the Ministry of Finances in 1984 was just for validating reliability and integrity of the fiscal software functions and issuing a certificate for them. The job could not be done, at that time (nor now, in a sense), but we took the challenge to come up with an answer. We are still working at it. Sections 4, 5, 6 are intended to illustrate how this answer is being formed (while evolving, the approach has been (had to be) applied to many real cases).

Another, more vague, source of demand is simply the expressed will to "know more about" or "to be very interested in", that we could detect in almost any environment where technical, administrative or commercial activities are performed.

Finally, we have observed that a good deal of demand is not explicitly expressed, but results directly from the fact that software is embedded in a vast range of products of various technologies, for which certification is requested (this was exactly the case of the Fiscal Meters).

In front of this wide spectrum of requests, devising a feasible, practical solution for software IV&V can absolutely not be postponed to when the art is settled. In the immediate future, certificates stating something about properties of products (medical equipment, for instance) that also depend on the behaviour of a program will have to be able to circulate through Europe. Among many questions that may arise, let us choose three:

1) Will such certificates explicitly include the software component?

2) How is this component to be approved?

3) Who is going to give the approval?

Two quick answers. Point 1): Yes, of course. Point 3): The European Certification Scheme, with its (to be) accredited Certification Bodies and Testing Laboratories [M-IT 03] would well provide the "approvers".

Point 2) takes a little longer. Our proposal will be presented in Sections 4, 5 and 6.

2. AN INDEPENDENT V&V UNIT

2.1. Profile of our IV&V Unit

Our IV&V Unit comes from a joint venture between a *Public* research & service Institute working in IT, and an Independent *private* Testing Laboratory (Figure 2.1).

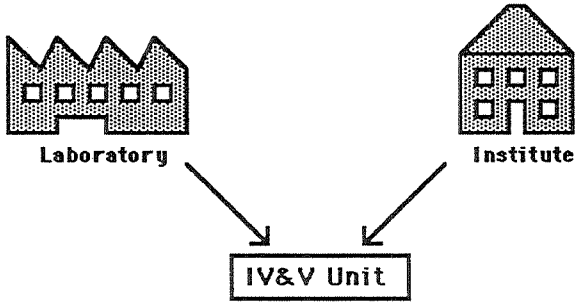

Figure 2.1 - IV&V Unit

Including a public-financed entity seems to have some advantages:

ADV:
- financial stability (the Institute can also choose other private laboratories as partners);
- larger inventory of tools;
- easier access to confidential project information;
- technical-scientific updating

and some disadvantages:

DISADV:
- complex organization;
- lack of flexibility;
- lack of efficiency due to a non business-oriented organization.

Our aim is to manage to get benefits from both types of organization. From the laboratory, we have:

ADV:
- flexible organization;
- sensibility to time schedule;
- business oriented staff.

2.2. The job of an IV&V Unit

In [Deutch88] the activity of an IV&V organization is naturally presented as not just monitoring ("reading", we would say) the developing process, but also interacting with it and influencing it in some way. Actually, at first we were rather concerned about the fact that an independent organization might, in a sense, approve or not what is also a consequence of its own activity. But the processes of reading and modifying are so intricated that it is impossible to play just one role. Therefore, we tolerate the fact that we are driven by demand, and articulate our job in the following actions:

- Assisting the user to prepare the Software Requirement Specifications.
- Monitoring software development activity during the entire life-cycle;
- Assisting the software developer during the entire software life-cycle;
- Preparing system test plans and integration test plans;
- Performing system tests;
- Reporting to the customer.

Each time we are called for a validation, we tailor these specific actions to the particular request.

As far as regards the criteria to be adopted for each point, our efforts are aimed to select and integrate from various sources:

1) State-of-the-art in software engineering.
2) Current standards, recommendations and guidelines, especially those likely to be referred to in the EEC environment.
3) Self-developed strategies, when sources 1) and 2) are not sufficient.

One of the methods of our Unit is to use the standard Forms for inspection, described in Section 4.

3. PROBLEMS WITH IV&V

Problems particular to IV&V are not easy to find in literature regarding the subject. Are there intrinsic differences between IV&V and Manufacturer's V&V (MV&V)? In our experience some critical, mutually related aspects can be pointed out:

- cost;
- lack of documentation;
- excess of documentation;

- difficulty in using automated testing tools;
- privacy of process and product information;
- manufacturer's resistance;
- certification.

Cost is probably the reason why IV&V is only recommended for critical software [Deutsch88]. However, we have observed that any software product, brand new from the developing phase, always contains defects. Therefore, it should be considered as a critical product. Perhaps it is not always risky, but it is worth spending some extra money in inspections and testing: why not by an Independent Unit? Of course, we agree that getting acquainted with someone else's project is more expensive.

Documentation is the main way we have for examining the process. Unfortunately enough, all the documents we were given during our activity interfaced very badly to external reviewers. Either the documentation is scarce, or, being richer, it is not easily accessible: in the latter case, one has to first review all of it to define the way of establishing an easier search for definite subjects.This enormously increases the cost of external validation.

It is for this reason that we have been investigating a low-cost method of extracting information about a process and are now experimenting standard Forms.

The use of automated test tools seems the practical solution to take the burden of effectively testing software products. Yet, in the case of IV&V, the independent organization must be prepared to accept products implemented in a variety of languages and dialects for a variety of hosts and operating systems. Therefore, each time, a new testing environment must be decided. Again, IV&V costs raise, in comparison with MV&V, in which testing must be reasonably equipped once and for all.

This paper cannot deal with all of the problems outlined above. However, employing standard Forms (Section 4) and a tailored Validation Suite (Section 5) can help, in our opinion, with more than one of these problems at a time.

4. STANDARD FORMS

The Forms are formatted documents designed by the IV&V Unit and filled up by the manufacturer's staff. They are an evolution of a previous approach of using ad-hoc questionnaires [Bertolino87] to allow an external organization to get acquainted with a project: they collect information about aspects of both the phases of the developing process and the final product, as well. This information is represented in a standard way, in order to help the independent reviewers to perform inspections.

Basically, a Form contains one or more questions about a specific function, part, structure, developing strategy, along with alternatives, where possible, or places for the answers. Besides, this contains spaces in which to put references to manufacturer (standard) documents. Figure 4.1 shows the general layout of a Form.

The underlying idea is to let the manufacturer explain the process/product: this should reduce the cost of external inspection, in that the manufacturer is obviously better better acquainted with what he has done. The references for the documentation are necessary in order to check whether the answers are correct and the Form is not a fake. We rely on the belief that the cost of a fake is comparable to that of documenting the real thing, so the manufacturer is encouraged to fill the Forms truthfully.

We distinguish two classes of Forms:

Process Forms: conceived to look for usability, traceability, consistency of process documentation;

Product Forms: conceived to look for critical points in the application (they are application dependent).

In order to cope with the problems presented in Section 3, the Forms should have the following characteristics:

- standard;
- synoptical;
- easy to manage;
- traceable.

In particular, traceability implies both the ability to reach inner details of the process/product, and the existence of a structure in the Forms system, so that a function or part of a product may be "pointed to" by a hierarchy of Forms.

5. THE VALIDATION SUITE

In order to choose the actions to be performed to validate a software product, we have devised a general scheme which includes a sequence of possible actions, and we denoted it as Validation Suite (VS). Basically, we apply inspection on a required subset of documentation, possibly supported by adequately filled forms, and testing on the executable product, possibly supported by automated testing tools. This implies that:

1) adequate documentation is available;

2) the product is testable.

Among all the possible scenarios, there is the case of the manufacturer performing his own validation (MV&V) (this matches with our experience). In such case, the best thing an IV&V Unit can do is to simply perform a validation over the MV&V documented (possibly with Forms) activity. This would keep the cost even lower for IV&V, and would prevent (through not involving deeper traceability) private process/product information from being disclosed.

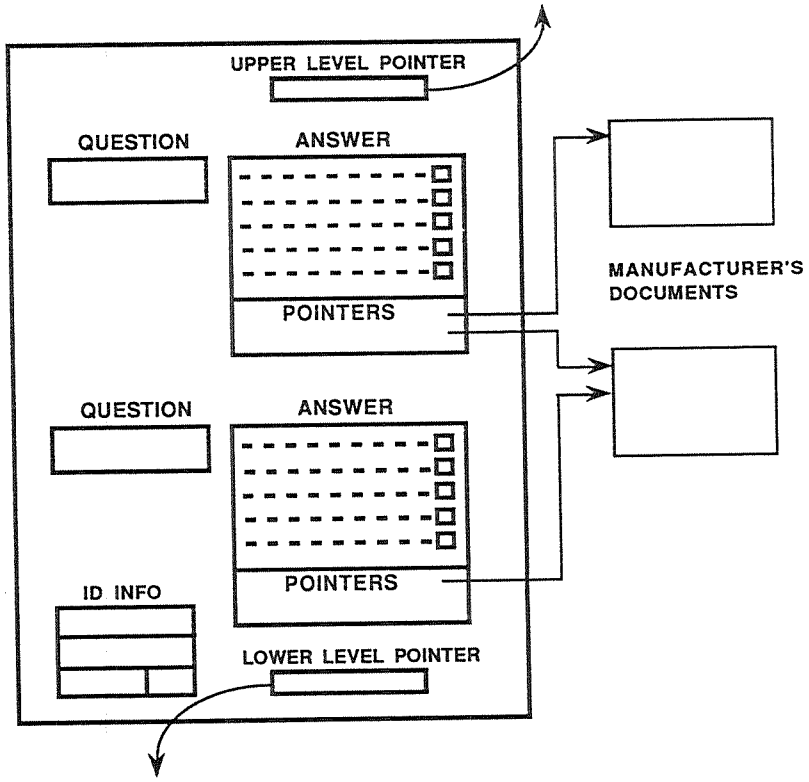

Fig. 4.1 - Form layout

The validation activity is divided into two phases: in the first one, to keep costs low, a preliminary short analysis of the documents submitted is performed, to check whether or not a more thorough validation can be carried out in the second phase. In case of insufficient documentation, the validation process is stopped and will be resumed when the manufacturer complies with the request for more information. We use the term "certifiability" to express the ability of a product to pass this first acceptance analysis.

Let us briefly comment on the concept of certifiability just introduced. Unfortunately, as it already happens for many others of the -ilities qualifying a product, also for certifiability it is difficult to give an objective definition, least of all a measurable one. Once again, the leading criterion is "cost": in a sense, we state that a product is "certifiable" if we judge that the cost for its IV&V, i.e. the time and effort we must invest, is "reasonably" adequate on the basis of our "good-sense" and of product criticity.

Once we accept starting a validation process, we also use this first phase to become acquainted with the product/process under examination and to tailor specific forms.

In the second phase of our validation activity, inspections, analyses and tests selected from the VS can be monitored or executed, depending on the degree of confidence we want to get with the process/product, and on the budget at disposal for the particular validation process.

Hereafter we report a synopsis of the VS:

1 Request for software product

- 1.1 Documents probably existing
  - 1.1.1 Functional specifications
  - 1.1.2 Design schemata and flow graphs
  - 1.1.3 Source code listings
  - 1.1.4 Documents describing the developing environment of the product to be validated: Hardware, Operating System, Applications, Approaches, Tools.
  - 1.1.5 Documents describing the actions performed during product development and testing
  - 1.1.6 Reference standards for 1.1.1, 1.1.2, 1.1.3, 1.1.4, 1.1.5 documents
- 1.2 Documents to be prepared
  - 1.2.1 Missing documents from point 1.1
  - 1.2.2 Filled Forms
- 1.3 Product instance
  - 1.3.1 On a support suitable for available host hardware
  - 1.3.2 With its target hardware

2 Acceptance analysis (Certifiability)

- 2.1 Adequacy criterion
- 2.2 Completeness criterion

3 Inspections

- 3.1 Document inspection, supported by Forms
  - 3.1.1 Check lists for documents (various levels)
  - 3.1.2 Forms (various levels)
- 3.2 Code inspection
  - 3.2.1 Check lists for code

4 Tests (planning/execution)

- 4.1 Functional tests
  - 4.1.1 Unit, Integration, Subsystem tests
  - 4.1.2 System tests
- 4.2 Structural tests
  - 4.2.1 Branch Coverage

The Validation Suite would involve a great deal of work if applied as a whole. In practice, it represents all of the possible actions, of which only some are reasonably worth performing in a single case of IV&V. The point is to devise the best subset to exercise each time, depending on product criticality, availability of documentation, cost constraints.

6. THE CERTIFICATE

Our validation process finishes with the issue of a "certificate". According to the position taken throughout the paper, in it we do not "guarantee" any property for the system validated. Instead, our certificate is a plain and exhaustive documentation of the validation process.

Thus, we certify that the product has passed the acceptance analysis, which is also described into the certificate. Then, we document the subset of the VS which has been performed, together with analysis results and evaluations.

The certificate presents a standard, formatted report, consisting of eight pages expressly designed to enhance self-descriptiveness and completeness of its contents.

In brief, the certificate statement can be summarized as follows:
"the product P and the accompanying documentation d1, ..., dn were submitted to the actions a1, ..., an of the VS. Results and evaluations are reported at the points r1, ..., rn of this certificate".

REFERENCES

[Bertolino87] Bertolino, A., Fusani, M., "Software Validation: A Government-imposed Challenge to the State-of-the-Art in Certification", *Computers & Standards*, Vol. 6, 1987, pp. 433-436.

[Copigneaux88] F. Copigneaux, "Software Process and Product Certification", in *CSR Software Certification Workshop*, Gatwick, 13th-16th September 1988.

[Deutsch88] Deutsch, M. S., Willis, R. R., *Software Quality Engineering: A Total Technical and Management Approach*, Prentice Hall, 1988.

[DOD/2167] DOD-STD-2167, Defense System Software Development, 1985.

[IEEE/729] ANSI/IEEE Std 729-1983, IEEE Standard Glossary of Software Engineering Terminology.

[M-IT-03/87] CEN, CENELEC, CEPT, Memorandum M-IT-03 on Certification of Information Technology Products, 1987.

Approving Software Products
W. Ehrenberger (Editor)
Elsevier Science Publishers B.V. (North-Holland)

SOFTWARE FOR FARMERS IN THE UNITED KINGDOM

John Nixon

School of Business
Royal Agricultural College
Cirencester, Gloucestershire GL7 6JS

ABSTRACT

This paper looks at the problem of developing appropriate software for agricultural applications. In this area, the gulf between the user and the writer of the software is potentially very great, with the practical farmer usually having no background in information technology and the systems designers and programmers typically unclear about what the users need and what level of IT background can be assumed. The main emphasis of this paper is the possibilities that exist for overcoming the gulf by participation of users in the decision making concerning what software needs to be developed, and the actual design process.

This theme will be developed both from the standpoint of the commercial software firm and the farmer. Firstly, the paper gives a brief background into the market for commercial packages for the farmer in the UK. Secondly, it turns to the supply side of this market looking at the life cycle of a software product. To make this life cycle more concrete we will look at the development process within the major firm producing agricultural software in the UK, Farmplan. This section will emphasise the efforts made by Farmplan to involve farmers in the process, especially during the specification and testing stages. Thirdly, it looks at the demand side. It looks at the experiences of farmers with microcomputers using various studies, mainly in the UK. Fourthly, it summarises our conclusions. It discusses the possibility of regulation of this market, but suggests that the arguments for regulation are hardly overwhelming and that the free market provision of software products as it exists may prove to be an acceptable solution.

1 INTRODUCTION

The use of computers "on" the farm began with the arrival of microcomputers, principally 8 bit machines with DOS operating systems, in the late 1970's. Previously, farmers had made use of mainframes and used the services of computer bureaux on an occasional basis, but this was a relatively expensive business, as it meant entrusting the farm's data to an outside agency and paying for the storage of the data on a permanent