My file

IN CONFIDENCE

SCIENCE AND ENGINEERING RESEARCH COUNCIL
RUTHERFORD APPLETON LABORATORY

INFORMATICS DIVISION

SOFTWARE ENGINEERING GROUP NOTE 60

issued by
Dr R W Witty

ACARD WG on SE
The Danger to Public Safety
- a discussion paper (draft 5)

10 July 1985

DISTRIBUTION: Prof C A R Hoare, Oxford
J Coplin, Rolls Royce Ltd
Miss S Bond, RSRE
J Whalley, British Aerospace plc
M Smith, ISTEL Ltd
M B Kelly, Cabinet Office
Mrs J Connell
C Whitby-Strevens, INMOS Ltd
P Ost, DMCP
D E Talbot, Alvey
K Crinean, NCC/STARTS
G Holmes, SDL (Camberley)
R W Witty
A Tolan, Cabinet Office
EL/ACARD file

KEYWORDS: SEGN 60 ACARD Standards Public Safety

PREFACE

This is the fourth draft of a discussion paper. Please let me have your comment, additions and deletions etc. Feel free to be critical!

1. THE PROBLEM: POLITICAL

No computer system failure has yet killed or injured a large number of people. It is only a matter of time before such a disaster occurs. What steps should the UK Government take to:

(a) prevent such a disaster,
(b) cope with it when it does occur,
(c) ensure such a disaster, having happened once, cannot recur?

2. THE PROBLEM: TECHNICAL

Stored-program digital computers must be amongst the most reliable mechanisms ever built by man. Millions of computers throughout the world, and thousands in space, are executing millions of instructions per second for millions of seconds without a single error in any of the millions of bits from which each computer is made. In spite of this, nobody trusts a computer; and this lack of faith is amply justified.

The fault lies not so much in the computer hardware as in the programs which control them, programs full of the errors, oversights, inadequacies and misunderstandings of the programmers who compose them. There are some large and widely used programs in which hundreds of new errors are discovered each month; and even when these errors are corrected, the error rate remains constant over several decades. Indeed, it is suspected that each correction introduces on average more than one new error. Other estimates offer the dubious comfort that only a negligible proportion of all the errors in these programs will ever be discovered.

New computers are beginning to be used in increasingly life-critical applications, where the correction of errors on discovery is not an acceptable option - for example industrial process control, nuclear reactors, weapons systems, station-keeping of ships at sea, aero engines and railway signalling. The engineers in charge of these projects are naturally worried about the correctness of the programs performing these tasks, and they have suggested a number of expedients for tackling the problem. Many of these methods are of limited effectiveness, because they are based on false analogies rather than a true appreciation of the nature of computer programs and the activity of programming.

The steps which ACARD is invited to recommend to the UK Government, in answer to the introductory question, are discussed under the following headings:

(a) Disaster Prevention
(b) Disaster Management
(c) Disaster Analysis.

3. DISASTER PREVENTION

The initiative for disaster prevention must come from the UK Government and system customers. Current software is built, operated and maintained using methods, tools and staff which, everyone agrees, cannot achieve the desired levels of a priori safety and reliability. The necessary improvements in software engineering require investment in research, development, production techniques, education training and legislation. The supply side will not make this investment of its own accord.

3.1 Registration

A Register must be established of those (software) systems which, if they fail, will endanger lives and/or Public Safety. The Register should be held by <X> where X is some UK Government controlled body.

3.2 Operation (Demand Side)

Before any organisation can operate a life-critical computer system it must first obtain a Licence To Operate (LTO) from <X>. A LTO will only be issued when the operator can demonstrate that certain conditions (detailed below) have been met.

Each life critical system must be operated by a Certified Software Engineer who is named as being personally responsible for the system. This Certified Software Engineer must have received the appropriate mathematical training.

A life critical system must be adequately maintained; this is one of the conditions of the LTO. Maintenance, ie Rectification and Development, must be the responsibility of a named Certified Software Engineer. It must be carried out by an organisation possessing the appropriate Licence to Construct (LTC).

(This to initially apply to public purchasing and operating organisations, then to all UK).

3.3 Construction (Supply Side)

Only approved suppliers may be allowed to build life-critical computer systems; thus suppliers must gain a Licence To Construct from <X>. A LTC will only be issued to suppliers who can show that they build systems to certain approved standards using methods which are mathematically sound, and certified tools and staff.

(Initially this will only apply to suppliers of public purchased systems but later will be extended to all UK).

3.4 Certification

A LTO may be granted when a Safety Certificate has been issued. Certificates will be issued for limited periods, eg 5 years. Operational systems will thus need to be recertificated (relicenced) periodically. (Analogous to Certificate of Airworthyness).

3.5 Reliability Data Collection

To aid research into system reliability and to assist Boards of Enquiry all Registered life-critical software systems must supply data to the (Alvey) Software Data Library.

4. DISASTER MANAGEMENT

Disasters can happen very quickly, eg plane crash takes a few minutes, or moderately slowly, eg Bhopal took several hours from start to finish. What should the LTO require of the operator in the event of a disaster?

4.1 Procedures

The LTO should require disaster management procedures to be laid down in advance of operation and practiced regularly during operation (eg 'fire drill practice').

Disaster management procedures are most crucial in systems for which loss of life cannot be prevented by switching off the controlling computer and/or the controlled process eg aeroplane flight control.

There may well be opportunities for sharing technology between military and civil disciplines to cope with disaster management.

4.2 Data Logging

The Disaster Management Procedures should include the logging of data so that any subsequent enquiry can ascertain the cause and progress of the disaster (eg 'black box recorder' in aeroplane).

4.3 Expert Advice on Standby

Teams of experts should be nominated, trained and kept on standby so that the UK Government has someone to call on at short notice for advice in a crisis which lasts hours or days (eg Bhopal).

5. DISASTER ANALYSIS

5.1 Collection of Reliability Data

During the normal (safe) operation of any life critical system, data on its performance and reliability must, as a condition of the LTO, be supplied to the (Alvey) Software Data Library. This data will be made available to any Enquiry. (This is additional to the data logging required in 4.2).

5.2 Board of Enquiry

Any disaster should be the subject of an official Board of Enquiry (similar to Rail, Air disaster enquiries). A Board of Enquiry may have the power to make changes to the system under investigation and/or the methods, tools, products and people associated with the Certification procedure.

5.3 Near Miss

Any serious 'near miss' must be reported to the Licencing Authority (and the Data Library). A Board of Enquiry should be held if the Licencing Authority is concerned at the incident's implications.

5.4 Any Error Triggers Board of Enquiry

Any error, no matter how 'small', in a software system which has been certified as being 'safe' must be the subject of an Enquiry. This is the only way of discovering weaknesses in the Certification process itself, or misuse or misunderstanding of its application. Enquiries concerning non fatal errors should not have disciplinary implications to encourage operators to always give notification of minor faults.

6. CERTIFICATION

The UK must develop the ability to certify aspects of software system construction and operation. These include:

(a) certification of the mathematical soundness of the methods of construction.

(b) certification that certified methods are properly applied during construction and subsequent maintenance (rectification & development).

(c) certification of the tools used during construction and maintenance.

(d) certification of the software engineers who build and maintain the systems.

(e) certification of the end product, ie the software itself, that it possesses certain properties. Modification of installed/operating software invalidates the certificate and so maintenance work implies re-certification before a LTO can be re-issued.

Methods should not be certified which are merely 'good practice'. Safety and reliability require more rigorous theoretical bases than existing good practice, so that system behaviour can be accurately and consistently <u>predicted</u>, hence the need for mathematical soundness to enable prediction to be based on mathematical proof.

Certification of a tool will only be given when it is shown that the tool preserves the mathematical soundness of the method it supports.

Certification of software engineers will only be given when they have completed an approved level of formal mathematical and methodological training together with an approved track record of experience. Certification should be of limited duration: recertification should require additional formal training both of the refresher type and new developments. Recertification should occur at 2-3 yearly intervals.

Certification of end products (and their components) implies proof obligations not just 'testing'. Proofs must be performed and checked by competent mathematicians (human or mechanical - both must be Certified).

7. TECHNICAL STANDARDS

The technical standards in common use in today's software industry are often built on mathematically unsound foundations.

Formal standards, in the ISO/BSI sense, are usually the result of national and commercial pressures and compromises. Due to the rapid pace of technical development in the IT industry, technical standards are usually not thoroughly tested before addition to the 'statute book'. Indeed many standards have never been implemented at all, let alone tested, when they become 'official'.

Informal standards, such things as popular methods and tools, are also usually built on unsound foundations. They tend to represent some encapsulation of 'best practice', but this is based on 'craft practice' rather than 'good engineering principle' let alone 'scientific foundation'. Much more research and development needs to go into establishing the sound basis of software engineering.

(a) Disaster Level

Failure could involve more than ten deaths. The whole of the software must be checked by formal mathematical proof, which is itself checked by a comptent mathematician. Further precautions required if damage limitation by switch-off is not feasible.

(b) Safety Level

Where failure could cause one death, and danger can be averted by switch-off. The whole of the software must be constrructed by proof-oriented methods, checked by a competent mathematician. On occurrence of a fatality, the enquiry will name the programmer and mathematician responsible, who might be liable for criminal negligence. Perhaps one error per 100,000 lines of code would be a realistic expectation, so that most shorter programs will contain no errors.

(c) Quality Level

Appropriate for software sold commercially, where error could bring financial loss to the customer. By law, such losses must be reimbursed. All programmers involved should be certified competent in mathematical methods of software design and construction. Their use of the methods is checked by sampling. An acceptable error rate would be one error per ten thousand lines of code delivered. Each error corrected requires recertification at safety level. If the target error rate is exceeded, certification is withdrawn. Eventually, all software used to construct other certified software should be certified to this level; and the construction of 'disaster level' software should include independent checks on the correct working of all support software used (eg check of binary code against higher level source codes).

(d) Normal Quality

Corresponds roughly to the best of current practice. (One error per thousand lines of code).

7.1 Improvement of UK Technical Standards

Major steps need to be taken now to improve the technical soundness of the standards used in the UK software industry.

The UK should develop, independently of international standards bodies in the short term, a new set of standards relating to:

(a) methods
(b) tools
(c) staff
(d) products

which have a much improved engineering and scientific foundation than current standards. Ideally, these standards should be issued under the auspices of BSI/IEE/BCS.

These improved standards will be a UK specific, medium term development. Priority should be given to this medium term activity, as opposed to enshrining 'current best practice', because it will lead to a significant UK commercial advantage (see section 7.6).

7.2 Software Industry Research Association

The development of these new UK engineering standards should be undertaken by creating a Software Industry Research Association. This will consist of both customer and supplier organisations. They will donate staff and money to support a small Directorate (a la Alvey) which will supervise the R&D programme which is actually carried out by industrial, academic and government establishments.

7.3 Demand Side Leadership

The drive to improve standards can only come from the demand side of the industry, of which the UK Government and other public purchasers are a dominant force in setting standards, eg MOD procurement policy.

It is recommended that the NEDO/NCC STARTS PPI (Public Purchasers Initiative) be given significantly increased power to persue the recommendations in this paper, being a natural focus for the demand side.

MOD and DTI must do much more to demand improved methods, tools, skills, products and standards from their suppliers.

7.4 Legal Implications

Work should be undertaken to investigate the legal implications of software system failure. What obligations do suppliers and operators have regarding public safety and employee safety? Does the Sale of Goods Act, Consumer Protection etc come into play?

7.5 Guarantees, Warranties

Demand side leadership should extend to include a call for increased use of guarantees and warranties from suppliers. The US/DOD has already instigated 2 year, zero cost warranties from some software suppliers.

The nature of long term maintenance contracts (eg MOD 20 year product life cycles) should be investigated by the STARTS PPI to see if demand side pressure can improve the situation.

7.6 Commercial Benefits

The consequences of demand side lead improvements in UK reliability, general quality and technical standards will create an international respect for UK work which will enable us to gain an increased fraction of the world market.

The Japanese view of quality, and the world's view of Japan's commitment to quality, will make the Japanese software industry a major threat to UK market share within 10 years unless action is taken now.

As opposed to technical innovation these new methods require managerial drive, engineering discipline, Government/industry cooperation, technical integration and commitment to quality; skills at which the Japanese exceed the UK.

8. STAFF

8.2 Professional Certification of Software Engineering Staff

Personnel of high quality are essential to the IT industry. One of the persistent difficulties is ensuring that personnel are of good quality; more difficult still is ensuring that quality is maintained. It is suggested that continuous professional certification would help maintain quality. Certification also would be a means of promulgating desirable techniques and for ensuring a common base of understanding and experience.

(1) certification should be set to maintain a minimum useful level and not be intended as a deterrent to entering the profession; the curriculum should be driven mainly by industry since it is a professional qualification.

(2) re-certification should be required at a rate consistent with technological progress; eg every two to three years.

(3) the certification examination would take into account the diverse and specialist nature of the industry by having specialist sections, as well as a general component; it is suggested that the Graduate Record Examination (US) is a suitable model. This is based on Multiple Choice questions, these can be marked easily and cheaply.

(4) the distance learning techniques of the Open University programme are an excellent means of disseminating course material; it is possible that universities, presently without a national curriculum, might adopt this as a core curriculum.

(5) there are suitable bodies for the promulgation and administration of a certification scheme; eg the BCS and the IEE who are currently considering Certified Chartered status for software engineers.

(6) the UK Government should initiate such a scheme by initial funding, further it by allowing tax incentives and ensure success by requiring 100% certification of software companies producing software for the UK Government and Public Purchasing bodies.

8.2 Education via Self Assessment CACM-style

Managers and engineers (managers especially!) can easily fall behind in their technical knowledge and not realise it due to the rapid rate of change in software engineering. How can they find out if their knowledge is up to date?

The CACM (monthly journal of the USA computer profession equivalent to UK BCS) regularly publishes sets of questions (and later, the answers) like newspaper quizzes at Christmas time, so that an individual can privately test his own knowledge against some professional norm.

Such self assessment can provide

(a) greater self awareness of an individual's knowledge level

(b) surrogate training (because reading the answers educates!)

(c) motivation to undertake training

(d) prepare individuals for certification and recertification

Self assessment is

(a) cheap

(b) easy to spread to wide audience

(c) quick and easy for individual to complete

(d) private ie no embarrassment to out of date senior staff.

8.3 Queens Award for Education & Training

The Professional Certification and self assessment ideas both require industry to increase the education and training of staff on a continuous basis. Those companies who make significant efforts should be praised publicly. It is therefore suggested that a Queen's Award for Education and Training be given to such companies.

9. BENEFITS

Improved software engineering will mean:

(a) safer public

(b) better UK sales of better products

(c) increased accountability when failure occurs

(d) creation of new industry: Certification which means more jobs in high-tech industry with considerable export potential for consultancy and small business.

10. FINANCING RECOMMENDATIONS

10.1 Registry/Enquiries

Suppliers and operators must be registered. Registered organisations pay for Registry by fees charged for registration, licencing and re-licencing.

Registered organisations pay for Boards of Enquiry via Insurance mechanism.

10.2 Certification

Certification agencies vetted by NATLAS and certification paid for as part of purchase contract (supply) and maintenance bill (re-certification), eg customer (MOD) pay supplier (SDL) to have independent certification agent (NCC) check out methods, tools, staff, standards and delivered product.

10.3 Research

The research and development needed to implement these recommendations should be financed by

(a) Alvey II

(b) MOD, DTI

(c) Levy on registered organisations to fund R&D activities for new standards in Software Industry Research Association, eg develop certification technology.

11. SHORT TERM ACTIONS

1.	Methods & Tools:	Alvey II should concentrate on standards for reliability.
2.	People:	increased need for education and 'life time' training and retraining. BCS/IEE Certified/ Chartered Engineer scheme for software engineers.
3.	Products:	STARTS PPI to lead demand side drive for improved standards.
4.	Standards:	BCS/IEE/BSI/Alvey to develop new, technically sound set of standards on which to base certification.
6.	Register:	Register of life-critical systems to be set up.
7.	Enquiries:	Boards of Enquiry to investigate software failures.
8.	Queens Award:	new awards to be set up for Reliability or Zero Fault Technology; for Education & Training; for Technology Transfer.