Analysis of the COTS Debate[1]

Felix Redmill
22 Onslow Gardens
London N10 3JU
UK
Felix.Redmill@ncl.ac.uk

Abstract

Modern safety standards place considerable emphasis on development-process evidence in the assessment of safety-related systems. The move to use commercial off-the-shelf (COTS) components in such systems has stimulated a debate about their appropriateness, for a feature of COTS items is usually a lack of development-process evidence. After pointing out the attractions of COTS, this paper addresses the main components of the debate: evidence, risk, and the often unmentioned challenge posed by the use of COTS items to the rigorous requirements of the standards. Finally, the paper points to the need for a convention on the evidence that should be provided to support claims for the safety of COTS items.

Key words: COTS, evidence, risk, safety standards.

1 Introduction - COTS and Safety Management

Safety-critical system developers and operators must not only achieve safety but also demonstrate its achievement, and these are two different tasks. Safety is application-specific and needs to be achieved by matching a system to the safety requirements of its intended application. To be well understood, the safety requirements should be derived from thorough risk analysis, which addresses not only the system's functionality but also the ways in which it might fail, deviate from its design intent, and be misused. The resulting safety requirements must then be met by including appropriate safety attributes in the system's design and construction.

For the achievement of safety, modern safety standards demand the appropriate choice and management of processes throughout the development life cycle. Then, for the demonstration that adequate safety has been achieved, the standards require evidence that the appropriate processes were indeed used. Some standards, for example the international safety standard, IEC 61508 (IEC 2000; Redmill 1998), provide guidance on what processes are appropriate at each stage. However, since the use of some processes is considerably more difficult, lengthy and costly than that of others, a demand for all systems to be equally 'safe', or as safe as possible, would lead to many being prohibitively expensive. Thus, various methods of categorising safety requirements have been introduced. For example, IEC 61508 defines a

----
[1] Published in Safety Science, Vol. 42, 2004, 355-367

scheme of 'safety integrity levels' (SILs) in which the need for greater risk reduction is represented by a higher SIL, on a scale of 1 to 4. Then, the higher the SIL the higher must be the rigour of the techniques, tools and management processes employed during system development.

Yet, even as the standards are mandating development-process evidence, there is a move to use commercial off-the-shelf (COTS) products in safety-related systems - and a feature of COTS products is usually an absence of development-process evidence.

In response to this lack, COTS proponents argue that evidence of 'good' development process does not guarantee safety, that it is not the only type of relevant evidence, and that evidence about the product itself is just as valid.

Are the constraints imposed by the standards definitive, or do the COTS proponents have a point? Having to rely on development-process evidence can be expensive. Further, there is no empirically established link between development processes and product attributes. So the standards may not offer the whole answer. But can COTS products be justified in safety-related applications? If so, how? Central to this debate are two issues, evidence and risk, and both are discussed below.

If the standards are 'correct' in their rigour, the COTS proponents may be arguing that 'cost overrides threats to safety'. But if they have a case, then the requirements of the standards, which form the barrier against COTS items, may be too demanding at a time when COTS products are necessary in the safety-related systems industry. This challenge to the standards forms the 'other side' of the COTS debate and will also be discussed.

The COTS debate includes both software and electronic products. Regarding the former, COTS software is a sub-set of a larger class that might be labelled 're-used software', within which non-commercial reused items, or non-development items (NDI), of software are included. Software products not developed to bespoke standards, for whatever reason, are sometimes referred to as 'software of unknown pedigree' (SOUP). The factor common to all of these is a lack of development-process evidence, a lack often shared by 'legacy' software. In this paper, the term 'OTS' (off-the-shelf) is used to cover all these items, both electronic and software - though the problem of evidence is seen to be more acute for software products than electronic hardware, and the problems of change and new versions are more frequent for software.

This paper offers an explanation of the COTS debate. It does not offer a 'solution', because solutions need to emerge from the theoretical and practical research and trials that are spawned by the debate. The paper shows the arguments of both sides, considers the crucial issues of evidence and risk, and describes the case against the standards. It then proposes the need for a convention on the evidence that should be provided in support of OTS products, and ends with a discussion of the issues.

2 The Desire for COTS Products - And Some Potential Problems

Being developed for a wide market, OTS products are likely to be cheaper, at least at the time of sale, than bespoke items. They are also immediately available, and this carries considerable weight, not only with system purchasers who can install off-the-shelf systems rapidly, but also with designers who find it convenient to employ off-the-shelf components in their systems.

There are also technological reasons in support of the use of OTS products. For example, Dawkins and Riddle (2000) point out that the relatively small safety-related-systems market cannot sustain the rate of technological advancement stimulated by the huge commercial market, and that it would suffer technological retardation if it did not use OTS products.

Thus, there are not only tactical reasons why individual purchasers may wish to use OTS products, but also strategic reasons why the safety-related systems domain as a whole might choose to do so. However, there are also potential disadvantages, as exemplified in the following paragraphs.

At the point of sale, OTS products are likely to be cheaper than bespoke items. Yet, the savings may not apply over the entire system life cycle, for there are other considerations. For example, if the OTS item is a black box, without source code and design details, the supplier must often be relied on for maintenance. This incurs financial costs and introduces a dependence that carries further implications. For example, security may require higher levels of staff, additional management structures, vetting procedures, and local arrangements.

Maintenance and other support is often only guaranteed if the system user accepts the supplier's frequent software upgrades. Although a rapid upgrade path is often claimed as an advantage of OTS purchase, this can incur extra financial costs and have serious technical implications. One problem derives from the possible composition of the upgrades, over which the purchaser will in most cases have no control. Because software is often viewed as easy to change, upgrades include not only new features and corrections to faults in previous versions, but also changes that might better have been achieved by other means. Thus, for the purpose of testing and safety assurance, an upgrade can seldom be perceived as 'changed' software, and should, rather, be considered as new. In addition, many of the new features, included to attract a wide market, are likely to be unwanted functions in the safety-related application. They increase the volume and complexity of the total package and introduce the risk of undesirable and perhaps dangerous effects. Moreover, the functions in the OTS software that are to be used may require tailoring which, without the benefit of design and source-code documentation, could further compromise safety.

Thus, not only at the time of purchase, but also at numerous other times throughout its life, the safety-related system will be dependent on untried software. At each of these points there is a lack of evidence of safety - and, perhaps, also a lack of safety. Further, the cost of carrying out safety assessments and re-assessments at all these times - if they are carried out rigorously - can be considerable. It should be estimated and allowed for during the initial system planning.

Then the possible problem of 'asynchronous' upgrading (e.g. when two suppliers make mutually incompatible upgrades) can create complex problems in the management of integrated systems, particularly when it is accompanied by the withdrawal of support for older versions. The additional challenges, including new risks, which asynchronous upgrading introduces can be substantial. Safety assessment, which must be carried out before the 'new' system can be brought into service, is likely to be more lengthy, more complex, and more expensive.

All this emphasises the importance of resisting upgrades for as long as possible - at least until there is evidence that the upgrade has succeeded in non-critical use. But by the time this evidence and the resulting confidence have been acquired, the next upgrade may be produced so that the cycle recommences. The need is also apparent to keep OTS (as well as bespoke) software under strict configuration control. Yet, it is not uncommon for user organisations to use OTS upgrades without question or trial and to exclude them from configuration control. When safety is an issue, this is a dangerous practice.

A further consideration is that, if necessary evidence is absent, the process of negotiating the co-operation of the supplier and the assessors in the acceptance of substitute evidence could become protracted and costly - and this may occur not only at the initial purchase but also at the times of all subsequent upgrades. If the missing evidence is critical, there is also a risk that the safety case will not satisfy the assessors. Clearly, the risks of using OTS software should be assessed in detail at the safety-planning stage of a development project, at which time the assessors' requirements for evidence should also be elicited.

A further potential problem is that in the absence of source code and design documentation from the supplier, there is a need to verify OTS software through black-box testing. Yet there are severe technical limitations on the confidence that can be derived from this, some of which are reviewed by Armstrong (2001). It is therefore not surprising that a great deal of research is currently enquiring into ways in which OTS software may be justified (e.g. Bishop et al, 2001; Jones et al, 2001; Dawkins and Riddle, 2000).

Thus, while there are forces pressing the designers of safety-related systems to employ OTS software, there are also factors which might negate its advantages and, in some cases, cause it to be deemed unsuitable, even after a system has been developed.

This section has mostly concentrated on issues that affect safety, but non-safety issues, such as commercial and security risks, should also be taken into account in any cost-benefit analysis, over both the short and the long terms. The costs of extra risk management activities could negate the financial advantages of using OTS items. Furthermore, other system attributes such as reliability, availability, maintainability and security might be compromised by the need for additional safety measures and the application of rigorous safety-risk management. All these issues could add to the costs of the OTS product and increase the difficulties of employing it in a safety-related application.

3 The Issue of Evidence

The heart of the matter is not OTS *per se*, but evidence. If OTS software were delivered with its development-process evidence, as required by the standards, as well as product details such as its source code, design, and test results, there would not be an issue. With sufficient evidence, a safety argument could be constructed to support its use (if its use were supportable) and assessment of its appropriateness in this or that application could be carried out. But, typically, OTS products are not accompanied by such evidence.

Evidence may be lacking for any of a number of reasons. In the worst case, a supplier may omit systematic documentation from the development process simply because of a lack of good engineering practice. In other cases, suppliers may justify confidentiality on the grounds that the required information is closely related to the product's success and therefore commercially sensitive. Then, in some cases there may be national security reasons for not including certain information with the exportation or dissemination of a product. Or, for legacy systems, the required information may have been destroyed or mislaid, it may not have been kept up-to-date when changes were made to the system, or it may never have existed.

The lack of evidence is not a conclusive indicator that a product is inappropriate for an intended application. However, safety must be demonstrated as well as achieved, and plausible demonstration must be based on a structured presentation of evidence in a safety argument. Safety is a system issue and context-specific. However good an OTS product may be, or however rigorously developed, it is only 'safe' or 'unsafe' insofar as it does not contribute to unsafe failures of the system in which it is a component, in the system's operational environment. Even high-quality software can lead to disaster when carelessly reused. Not only must there be confidence that the OTS product is of the appropriate quality *per se*, the evidence must also exist to satisfy a safety assessor that it is suitable for the proposed application.

The fact that certain processes were used in the development of a product is not proof that the product is appropriate for use in any given safety-critical system. Only good design and construction, in accordance with correct requirements, will result in a system possessing the appropriate attributes for a particular application. Development processes do not ensure appropriateness. Yet bespoke systems, with substantial evidence of their development processes, are likely to be more convincing to a safety assessor than OTS products with none, and arguments for their safety are more easily made and assessed.

In the absence of development-process evidence, as called for by the standards, it may still be possible to acquire evidence by investigating the product itself. But here too there are difficulties. Proven-in-use evidence would be persuasive if the system to be used is unchanged in content and configuration from that claimed to be proven, and the environment for future use is the same or very similar to that in which monitoring was carried out. But that is seldom the case. One bug found and fixed renders the system a 'new' system, as does every added function. Thus, because small changes in a system or its operating environment can lead to large changes in the behaviour of software, caution is needed when evaluating arguments that an OTS product (or any software product) has been 'proven in use'.

If a safety-related system is to be proven by testing, this needs to include both black-box testing and static analysis. The former is inconclusive and the latter cannot be carried out if the supplier has not provided the system's source code - which is usually the case.

Thus, gaining sufficient evidence about an OTS product to have confidence in its use in a safety-related application is, at best, difficult. Frankis and Armstrong (2001) suggest that to assess OTS software adequately it must be validated against thirteen 'evidential requirements', and they list five classes of examination of the requirements: black-box methods, previous usage analysis, design-intention assessment, code assessment, and open-box assessment. Such extensive validation, and the analysis to facilitate it, are not usual aspects of implementation programmes. Although the proposals are extensive, they have not been shown in practice to be sufficient. But they represent an early attempt to address the evidential problem, for the

COTS debate is concerned with whether sufficient product evidence can be derived to allow safety assessment, and whether such evidence can replace the process evidence demanded by the standards. These issues recur in Section 5.

4 The Issue of Risk

The question of risk is as important as the question of evidence. If there were no risk attached to the use of OTS products, there would be no safety issue. At the same time, there can never be zero risk. Even bespoke systems, developed to the highest standards, can and do fail. Some risk must always be accepted, but in accepting it we need to be confident that it is indeed tolerable, and for this we need to understand it. But how can we assess the risk of an OTS product without evidence?

Without evidence from the supplier, the information required for a risk analysis of an OTS product depends on the part that the product is intended to play in the safety-related system. By the use of FMEA (failure modes and effects analysis) chains of cause and effect, leading from the OTS system or component to the relevant system-level hazardous events, may be derived so that rudimentary risk analysis can be carried out. In some cases, effects may depend on the particular failure modes, so it is desirable to know these in detail - though it may not be possible to do so.

If the OTS system is a black box, it is difficult to make a convincing argument for safety for a number of reasons. First, verification that all failure modes have been identified is not possible. Particularly in the case of software, in which faults are systematic rather than random, previous experience cannot be assumed to have revealed them all. Second, failures monitored at the interface between the black box and the rest of the system cannot easily be traced to their true causes within the OTS product and cannot be assumed to be representative of particular failure modes. Further, fixes made at the interface may only address symptoms, leaving unrecognised faults that could lead to dangerous system failures in the future.

Thus, in the absence of evidence to provide confidence in the reliability of the OTS product, it should be assumed that if it could cause a hazardous event it would - i.e. that its probability of dangerous failure is unity. To do otherwise, certainly if it is software, would be contrary to safe practice. In their report on the failure of Ariane 5 Flight 501, the Inquiry Board (1996) stated, 'software should be assumed to be faulty until applying the currently accepted best practice methods can demonstrate that it is correct.'

With the probability of dangerous failure assumed to be one, and any attempt to derive a more accurate probability being purely speculative, risk analysis and management would have to be based on consequence alone.

A first step towards assessing the acceptability of a critical OTS product would then be to examine the tolerability of the consequences *per se*. But tolerability is itself a subjective notion that involves a trade-off between safety and benefits. This gets to the nub of the COTS debate, for the proponents of the OTS product may argue that its benefits - such as cost, functionality, and immediate availability - outweigh the safety considerations. Just as it may not be possible to muster a convincing argument for safety, so it may not be possible to prove in advance that the OTS product is less than tolerably safe. Do we then apply the precautionary principle and adhere to safe practice? To do so may involve missing practical or commercial opportunities. Moral and ethical considerations enter the debate.

However, if a risk analysis is based on consequence, and if the consequences of failure are deemed intolerable, the next step should be to enquire into the use of a 'protection function' to reduce the probability of the consequence occurring. Such protection might be installed either in direct combination with the OTS product or at some point in a fault tree between the product and the hazardous event. In determining the required probability of failure of the protection function and, thus, its safety integrity level according to IEC 61508, no contribution to reliability could be assumed from the OTS component (its probability of failure is assumed to be unity).

A further point should be made here. The protection-function principle is based on the assumption that the protection function is independent of what is being protected. Care should be taken in assuming that there are no modes of failure common to the protection function and the OTS component.

In summary, the requirements for evidence in support of the OTS product should be related to the risks involved. Early decisions on what is needed for assessment should be made in conjunction with the safety assessors. If adequate evidence is not available, risk analysis must be based on consequence, with decisions being required about the tolerability of the consequences of the various possible hazardous events, and on whether and how the use of a protection function would justify the use of the COTS product.

The above discussion assumes that a risk analysis is carried out using whatever evidence is available. But suppose there is strong pressure to use the OTS product in the absence of necessary evidence, or without a protection function that analysis shows to be necessary (perhaps because it is expensive to implement), or, indeed, without analysis being carried out? How should decisions be made under such pressure?

From the safety perspective, the issue of employing an OTS product is one of deciding what risks to accept and when it is worthwhile to accept them. The more evidence there is, and the greater the confidence in it, the more consensus there is likely to be in decision-making. But when there is little or no evidence to support a claim for safety, the decision becomes a gamble. Further, in the absence of evidence, there is no knowledge of the odds that pertain. Then, any decision will depend on who is making it and how they perceive the risks and benefits of using the OTS product. Value judgements are involved, and decisions need to be made not by one party but by all stakeholders.

5 The Challenge to the Standards - The Other Side of the Debate

One side of the 'COTS debate' is the question of whether the use of OTS products in safety-related systems can be justified. The other side is a challenge to the standards' demand for development-process evidence in safety assessment. The issue is this: if OTS products are in fact found to be admissible, how can the standards' requirements for rigorous development processes be valid? After all, a good product can result from imperfect development processes. Moreover, although a relationship between product quality and the development processes may seem intuitively reasonable, it has no empirically proven foundation. Indeed, although there may be good correlation between bad process and bad product, there seems to be poor correlation between good process and good product.

Further, the relationships between safety targets and particular development techniques differ between standards and therefore cannot be said to represent a co-ordinated view. Whereas IEC 61508 provides two extensive annexes of tables that explicitly define the appropriateness of specific techniques to the various SILs, the Motor Industry Software Research Association guidelines (MISRA, 1994) provide a single table in which only outline guidance is offered. The Ministry of Defence standard, MOD 00-56 (MOD, 1996), calls for 'design rules and techniques appropriate to each safety integrity level' to be 'determined prior to implementation of the functions of the system'. Only IEC 61508 lays down strict relationships between techniques and SILs, and they are based on expert judgement. In MOD 00-56 the relationships between techniques and safety targets are defined by project personnel, may be different from those of IEC 61508, and may differ from project to project. If a system constructed according to MOD 00-56 turned out to be 'safer' than one built according to IEC 61508, it could be claimed that the latter's rigorous requirements are unnecessary. Further, if a safety-related system not developed according to any of the standards is found in practice to meet a given safety target, it might be used as a basis for refuting the standards' call for development-process evidence.

Most standards admit 'proven-in-use' evidence, though with constraints. For example, both the software and its operational environment must remain unchanged not only during the period in which the proof is gathered, but also in the new safety-related application. If the in-use behaviour of the software, and the conditions under which it operated, are shown to be relevant to the safety-related application, is the resulting evidence less valid than that of the development process? Many advocates of OTS products think not.

Yet we would need to carry out a great deal of comparison in order to claim that the old operational situation is representative of the new. Moreover, it is not uncommon for an undetected software fault to cause a failure for the first time after years of operation, simply because the relevant logical path had never previously been exercised. Thus, the constraints of the standards on 'proven-in-use' arguments have some justification.

Moreover, OTS items not developed to the standards' rigorous processes, or without evidence that they have been, may also be subjected to testing, realistic operational trials, and simulated conditions. If the source code is available it may be inspected. Does such product evaluation outweigh the fact that the development processes may not have been in conformity with the requirements of the standards? The OTS product proponents think so. After all, a sound product can result from a flawed process (though the process actually used may not have been flawed). Further, it can be argued that a process not defined by the standards is not necessarily 'bad'.

Yet, the standards fall back on the development process with justification. As shown by Littlewood and Strigini (1993), the extent to which the reliability of software (both OTS and bespoke) can be proved by testing is severely limited - not because appropriate tests cannot be devised, but because adequate testing cannot be carried out in cost-effective time. Thus, proponents of the standards argue that development-process evidence is an essential (though not always sufficient) part of any safety justification. There is also a precautionary principle argument: if safety cannot be demonstrated, it should not be assumed - and this leads to the rejection of reliance on product evaluation because of the intractability of the task.

Yet, it is not only the task of product evaluation that is intractable. Devising a development process that is 'perfect' for the intended application, and managing it perfectly, are impossible demands. Moreover, although the standards advise on what to do in order to achieve safety, they do not yet offer guidance on what must be demonstrated in order to make safety claims. So perhaps the OTS proponents have a point.

Further, although standards' appeal to the development process has technical justification, the lobby for OTS products still challenges the validity of the approach. The failure of software development projects is legendary, and even those employing the most rigorous processes have frequently not only exceeded both budget and time, but also produced systems that have been shown at an early stage of testing or operation to be unsuitable or of poor quality. Rigorous processes are considered cumbersome and unable, of themselves, to 'deliver the goods'. Thus, there is a view that reliance on such processes to achieve safety could be misplaced. Simpler, faster, less costly methods are sought, and these are perceived to exist in the commercial market where intense competition keeps costs low and propels technological advances that are quickly introduced into products. But, while this argument may be persuasive, it does not go far enough, for safety applications demand safety assessment in advance, and this requires evidence. The debate must then return to the question of what form the evidence must, or may, take.

It has also been suggested that purchasers are reluctant to meet the costs of applying the processes mandated by IEC 61508 and MOD 00-55 (MOD, 1997) for the higher SILs. If this is the case, it will be difficult for regulators to enforce them. It will be interesting to monitor this issue as experience of use of the standards grows.

6 We Need a Convention for the Provision of Evidence

It is typical to assume that OTS implies a lack of evidence and that suppliers do not want to provide evidence to support a safety argument. But do we have to accept these assumptions? Some suppliers (for example, of operating systems) are keen to be respected in the safety-related systems community and provide evidence when asked to do so. This willingness should be encouraged, for there will be no evidence if it is not called for.

So, might we strive to develop the terms of a convention for the provision of relevant evidence? There are a number of categories of evidence that could be made available.

First, there is evidence of the development process, as required by the standards, to demonstrate compliance with defined good practice or with an accepted standard. Second, there is direct evidence about the OTS product itself, for example the test plans, cases and results, and information on experience of its use. And third, there is evidence to engender confidence in the company that developed the product. In many industries it is accepted practice for customers, or potential customers, to audit suppliers, either against a quality standard or to gain confidence in their competence or in particular processes. There is scope for auditing of suppliers of OTS products by potential purchasers in the safety-related systems domain - indeed, throughout the software purchasing community - to become more prevalent, not only against quality and safety standards, but also to assess their risk-analysis, business, and other processes.

There is also the need for evidence to support claims that a supplier makes about software, for example that it is of a given SIL. There are many possible interpretations of such claims (Redmill 2000), so a convention would need to embrace both definition and evidence.

Products may be claimed to be 'proven in use'. But such claims should not merely be stated; an argument and supporting evidence should also be provided. For instance, which functions are claimed to have been proven in use? What was the extent of the coverage of the use? What observational evidence is relied on for the claim? Which functions in the software were not used or are not claimed to be proven? Are these functions independent of those for which the claim is made, or could their failure or operation affect them? Was the same version retained in the same configuration, without change, throughout the period of use, and is it the version now being offered? We need a convention for the content and form of proven-in-use arguments, and it might also include some design information, such as which functions are claimed to be partitioned from others (so as to be independent of them) and how the partitioning is achieved.

A possible difficulty in agreeing and applying a convention is that the safety-critical systems community does not wield significant influence on suppliers in the commercial market. But this is not an adequate reason for doing nothing. It is likely that many suppliers would see benefits in providing evidence about their products, as is currently the case in the supply of real-time operating systems. Where such suppliers lead, others are likely to follow if asked to do so.

7 Discussion

Modern safety standards emphasise the requirement for development-process evidence in support of a claim that software is suitable for use in a safety-related system. As a usual feature of OTS products is the unavailability of such evidence, adherence to the safety standards would in most cases preclude their use. However, many system designers, suppliers, and purchasers perceive advantages in employing OTS systems and components, and, in spite of counteracting disadvantages, some of which apply across the life cycle, the demand is resulting in their use.

Can their use in safety-related systems be justified? Development-process evidence is suggestive rather than conclusive, and the proponents of OTS products claim that product evidence, derived from testing and experience of use, can be just as valid. But these forms of evidence are not conclusive either. Software and complex electronics cannot, in general, be proved in cost-effective time by testing, and even small remaining faults can have catastrophic consequences. Further, there are constraints that make proven-in-use arguments for safety difficult to claim. The product to be used must be unchanged from, and in the same configuration as, that which has been proved, and the intended application must be the same as, or very similar to, that for which proof is claimed.

Thus, neither assessment of the development process nor evaluation of the product can, in the current state of the art, provide proof of safety. Perhaps they never will, for safety is application-specific and even 'good' products can threaten it in inappropriate or unexpected circumstances. Wise experts recognise that evidence should be sought from all three sources - but this does not resolve the debate on whether OTS products can be acceptable when development-process evidence is totally absent.

Posing the issue in these terms leads to a frequently unarticulated side of the COTS debate, which is the implicit challenge to the validity of modern safety standards: if COTS products without evidence of their development processes are found, retrospectively, to be suitable for safety-related applications, how can the standards' rigorous requirements for development-process evidence be sustained? They will have been shown, at least in the worst cases, to be irrelevant. As use of OTS components and systems increases, this issue is likely to become more prevalent.

In recent years safety thinking has advanced from basing an assumption of safety on reliable functionality to basing the management of safety on an understanding of the risks involved. Risk can never be reduced to zero, and we must strive for increased confidence in safety rather than certainty. The level of confidence required depends on the situation. The debate about where the evidence should come from has plausible arguments on both sides. A lack of development-process evidence should not in all cases preclude the use of OTS items, but in all cases safety should be based on an understanding of the risks and rational decisions about their tolerability. Based on argument, one form of evidence may be replaced by another, but it would be a retrograde step in safety management if the need for evidence were disposed of altogether. Then it really would be the case that 'costs override threats to safety'.

It is argued in this paper that we need to define, and begin to put in place, a convention on what evidence should be provided by OTS-product suppliers in order that arguments for their safety, in the context of specific systems and applications, may be developed. Such a convention should also cover the evidence required in support of other safety-related claims made by suppliers, for example that software, or a system based on software, meets a SIL or that it has been proven in use.

At a time when technological innovations are forcing us to revise our attitudes to both the economics and the methods of software development and use, the issues surrounding the use of OTS products must be confronted. The COTS debate is both necessary and healthy for the safety-related systems industry, and it will affect not only the ways in which we develop safety-related software and systems in the future, but also the ways in which we use them and justify their use. The effort being expended on investigating how the use of OTS products can be justified will mean that, even without the prospect of early resolution, the debate is likely to have many fruitful side effects, both theoretical and practical.

Acknowledgement

Thanks to Jim Armstrong for comments that were influential on this paper.

References

Armstrong, J., 2001. The Risks of a SOUP Diet. *Safety Systems, The Safety-Critical Systems*

Club Newsletter, 10, 2, January.

Bishop, P. G., Bloomfield, R. E., and Froome, P. K. D., 2001. *Justifying the Use of Software of Uncertain Pedigree (SOUP) in Safety-related Applications. HSE Contract Research Report No. 336/2001*. Health and Safety Executive, UK.

Dawkins, S. K., and Riddle, S., 2000. Managing and Supporting the Use of COTS. In Redmill F and Anderson T (eds): *Lessons in System Safety - Proceedings of the Eighth Safety-critical Systems Symposium, Southampton, UK*. Springer-Verlag, London.

Frankis, D., and Armstrong, J., 2001. *Software Reuse in Safety-Critical Applications - Summary Final Report*. Advantage Technical Consulting Report. March.

IEC 2000. *International Standard IEC 61508: Functional Safety of Electrical/Electronic/Programmable Electronic Safety Related Systems*. International Electrotechnical Commission, Geneva

Inquiry Board, 1996. *ARIANE 5 Flight 501 Failure - Report by the Inquiry Board*. Paris, 19 July

Jones, C., Bloomfield, R., E., Froome, P. K. D., and Bishop, R. G., 2001. *Methods for Assessing the Safety Integrity of Safety-related Software of Uncertain Pedigree (SOUP). HSE Contract Research Report 337/2001*. Health and Safety Executive, UK.

Littlewood, B., and Strigini, L., 1993. Assessment of Ultra-high Dependability for Software-based Systems. *CACM* 36, 11, 69-80.

MISRA, 1994. *Development Guidelines for Vehicle Based Software*. The Motor Industry Research Association, UK

MOD, 1996. Defence Standard 00-56: *Safety Management Requirements for Defence Systems Containing Programmable Electronics, Part 1: Requirements*. Ministry of Defence, Glasgow, UK

MOD, 1997. Defence Standard 00-55: *Requirements for Safety Related Software in Defence Equipment, Part 1: Requirements*. Ministry of Defence, Glasgow, UK

Redmill, F., 1998. IEC 61508: Principles and Use in the Management of Safety. *Computing & Control Engineering Journal*, 9, 5. IEE, London

Redmill, F., 2000. Safety Integrity Levels - Theory and Problems. In Redmill F and Anderson T (eds): *Lessons in System Safety - Proceedings of the Eighth Safety-critical Systems Symposium, Southampton, UK, 2000*. Springer-Verlag, London