This is a preprint of an article published in *Software Process: Improvement and Practice*, volume 6, issue 1, pp. 47-62, 2001,

Processes for Software in Safety Critical Systems

O Benediktsson[1], R B Hunter[2] and A D McGettrick[2]

[1]University of Iceland, Dunhaga 5, Reykjavik, Iceland
[2]Department of Computer Science, University of Strathclyde, 26 Richmond Street, Glasgow G1 1XH, UK

ABSTRACT
Two complementary standards are compared, both of which are concerned with the production of quality software. One, IEC 61508, is concerned with the safety of software intensive systems and the other, ISO/IEC TR 15504, takes a process view of software capability assessment. The standards are independent, though both standards build on ISO/IEC 12207. The paper proposes a correspondence between the safety integrity levels (SILs) of 61508 and the capability levels (CLs) of 15504, and considers the appropriateness of the 15504 reference model as a framework for assessing safety critical software processes. Empirical work from the SPICE trials and COCOMO II is used to support the arguments of the paper as well as to investigate their consequences. The development of a 15504 compatible assessment model for software in safety critical systems is proposed.

Keywords
Process assessment, safety critical software, international standards

1 INTRODUCTION

Software plays an increasing role in control systems. Safety critical control systems are used extensively in modern society, and failure of such systems can lead to personal injury and even death. A number of disasters of this nature have been reported. It would seem that in many cases such disasters might have been avoided if more rigorous processes had been used in the development of the software for such systems. This applies to all parts of the software process from acquisition and specification, through development, and into deployment and maintenance. The two main aims of this paper are

- to investigate whether a correspondence can be proposed between the safety integrity levels defined for safety critical systems, and the minimum process capability levels required to achieve these safety integrity levels
- to investigate the adequacy of the emerging software process assessment framework as a basis for assessing safety critical software processes

Two comprehensive suites of standards that are of special interest in this context are

CEI/IEC 61508: 1998-2000, *Functional safety of electrical/electronic/programmable electronic safety-related systems. Parts 1-7* (IEC 1998-2000)

ISO/IEC TR 15504, 1-9 :1998, *Information technology – Software process assessment – Parts 1-9* (ISO/IEC 1998)

These standards will be referred to here as 61508 and 15504 respectively. 15504 is also sometimes known as SPICE after the SPICE (Software Process Improvement and Capability dEtermination) project which is playing a major part in the development of the standard. 61508 has been published in parts, some in 1998 and some in 2000. 15504 is currently a technical report of type 2. Work to produce an international standard is well under way at the time of writing. All five parts of the standard should be approved by around the year 2002.

The 61508 standard defines four discrete safety integrity levels (SILs) for specifying the safety integrity requirements of the safety functions to be allocated to safety-related systems. Safety integrity level 4 (SIL4) is the highest level of safety integrity and safety integrity level 1 (SIL1) is the lowest. The SIL's relate to the probability of a failure as shown in table 2.2.

Part 3 of 61508 deals with software issues and is the focus of attention here. The software process framework used is

an extension of some of the processes in the ISO/IEC 12207 Software Life Cycle Processes standard (ISO/IEC 12207). Recommendations are given as to the required use of processes (e.g. configuration management), specific activities (e.g. code review) and techniques (e.g. structured programming). The recommendations become more specific and stringent as the SIL level increases.

The 15504 standard “provides a framework for the assessment of software processes. This framework can be used by organisations involved in planning, managing, monitoring, controlling, and improving the acquisition, supply, development, operation, evolution and support of software.” (15504 Part 1)

At the heart of the 15504 standard is the reference model described in part 2 of the document set. The reference model defines a two-dimensional model of processes and process capability that forms the basis of any 15504 conformant assessment model. The first dimension is the *process dimension*. The processes are grouped into five process categories and are similar to the processes in ISO/IEC 12207. The second dimension is the *capability dimension* which characterises the level of capability that an organisation unit has attained for a particular process, or which may be used by the organisation unit as a target to be attained. (15504 Part 2)

The reference model defines six discrete *process capability levels* (CL0-CL5). *Process attributes* are features of a process that can be evaluated, providing a measure of the capability of the process. Capability levels 1-5 each have either one or two process attributes associated with them. Capability level 5 (CL5) has the highest level of process capability and capability level 0 (CL0) has the lowest. At CL0 (incomplete process) the activities are carried out in unorganised and incomplete fashion while at CL5 (optimising process) continuous process monitoring against effectiveness and efficiency goals is used to produce quantitative feedback to improve the process.

The 15504 standard "provides a structured approach for the assessment of software processes for the following purposes

- by or on behalf of an organisation with the objective of understanding the state of its own processes for process improvement;
- by or on behalf of an organisation with the objective of determining the suitability of its own processes for a particular requirement or class of requirements;
- by or on behalf of one organisation with the objective of determining the suitability of another organisation's processes for a particular contract or class of contracts." (15504 Part 1)

The framework for process assessment encourages self-assessment.

The main questions that are addressed in this paper are the following:

- If an organisational unit intends to develop software of a specified safety integrity level, can 15504 process capability profiles be stipulated that would be the minimum required in order that the software could be developed effectively.
- Does the 15504 reference model provide an adequate framework for the assessment of safety critical software processes.

Section 2 of the paper is an overview of IEC 61508 and section 3 is an overview of the standard ISO/IEC TR 15504. Section 4 compares the two standards with regard to the adequacy of the 15504 capability levels for the production of safety critical software, and the appropriateness of the 15504 reference model for the assessment of safety critical software. Section 5 examines some empirical evidence regarding the relationship between process capability and software criticality. The conclusions of the paper are outlined in section 6.

2. OVERVIEW OF IEC 61508

2.1 Background

IEC 61508 is a generic international standard that addresses the functional safety of systems, and primarily systems developed using electrical, electronic and computer technology. Increasingly the trend in industry is towards the inclusion of programmable electronics in safety related systems and the standard reflects that shift. The emphasis is very much on a *systems* perspective - it is the system that achieves a level of safety - and the requirements on other sub-systems and components are derived from an analysis of the systems requirements and the system structure.

The generic nature of the standard is twofold:

- it is not sector specific and so is applicable to systems developed using different technologies, or indeed a range of technologies
- it provides the basis for the development of sector specific standards

2.2 History and Status

IEC 61508 is a development of the earlier IEC 1508. These documents have been under development for around 15 years through Sub-Committee 65A of the

International Electrotechnical Commission. Within the safety industry the production of the document is seen as a very significant development which captures many of the concerns and principles, as well as the good practice, that underpins the development of safety critical and safety related systems. From a professional and liability perspective, the very existence of such a standard is a fundamental contribution to the community; the guidance embodied in the standard reflects best practice and has the potential and expectation that adherence to the standard may help to absolve the engineer of any charge of negligence in the event of a disaster.

2.3 Nature of the Standard

The standard makes reference to the EUC - equipment under control. The standard aims to provide guidance on how to ensure that this operates in a safe manner. A number of fundamental principles are used:

- the approach is based on risk
- there is a safety life cycle
- safety integrity levels are used

The standard is presented in seven parts:

Part 1 General requirements

Part 2 Requirements for E/E/P (Electrical /Electronic/Programmable Electronic) safety related systems

Part 3 Software requirements

Part 4 Definitions and abbreviations - explaining terms used in the standard

Part 5 Examples of methods for the determination of safety integrity levels - provides guidance on applying ideas from parts 1 and 2 in determining safety integrity levels

Part 6 Guidelines on the application of parts 2 and 3

Part 7 Overview of techniques and measures

2.4 Hazard and Risk Analysis

The important tasks of ensuring safety presupposes an analysis of the entire set of risks associated with a system, and a quantifiable evaluation of them. The standard advocates a systematic approach to hazard identification, hazard analysis and risk assessment. The techniques recommended in addressing these include techniques such as *event tree analysis, failure mode effects (and criticality) analysis* and *cause consequence analysis*. Of course the application of each of these needs to be guided by well established best practice. See for instance (Leveson 1995).

The identification of risk leads naturally to application of the ALARP principal, the cornerstone of much of the UK legislation in safety matters. This requires that 'risk be as low as reasonably practicable'; where the cost is manageable there is a requirement to apply risk reduction techniques to minimise risk. For systems that are ultimately built the overall risk must fall within the class of being tolerable; other possibilities are unacceptable.

Essentially the design process is largely governed by risk considerations. The standard advocates the normal three stage approach of hazard removal, hazard reduction and hazard control. SIL levels can also be used to guide architectural design issues.

2.5 The Safety Life Cycle

The framework for 61508 is a life cycle model - a safety life cycle model which is superimposed on a systems life cycle model. This is described in terms of a set of phases, each of which is shown in table 2.1 along with a short explanation of what it entails

Phase	Description
1. Concept	develop an understanding of the EUC
2. Overall Scope	determine the boundary of the EUC
3. Hazard and Risk Analysis	determine hazards and risks
4. Overall safety requirements	develop specification
5. Safety requirements allocation	allocate safety functions
6. Overall operation and maintenance planning	plan operation and maintenance
7. Overall safety validation planning	facilitate safety validation
8. Overall installation and commissioning	plan installation
9. E/E/PE safety-related system realisation	create system
10. Other technology safety related systems realisation	create other technology systems
11. External risk reduction facilities: realisation	create risk reduction facilities
12. Overall installation and commissioning	install and commission
13. Overall safety validation	validate that safety met
14. Overall operation, maintenance and repair	maintain ensuring safety
15. Overall modification and retrofit	ensure safety after modification
16. Decommissioning or disposal	ensure safety at end of life

table 2.1 Safety Life Cycle Model

2.6 Safety integrity levels

The 61508 standard incorporates 4 different safety integrity levels, or SIL levels. A safety integrity level is defined as

The likelihood of a safety-related system satisfactorily performing the required safety functions under all the stated conditions, within a standard period of time.

In defining the criteria associated with these levels a distinction is drawn between

(a) low demand systems, where the system is invoked spasmodically to deal with some situation that needs to be addressed, e.g. protection systems in a nuclear power plant; here the probability of failure of each demand is defined

(b) high demand or continuously operating safety related systems; in this case the probability of failure per hour is identified.

The definitions of the four SIL levels are given in terms of the probability of failure. For the low demand case this is phrased in terms of the average probability of failure, whereas for the high demand case, this is expressed as the probability of failure per hour of operation.

The probabilities are outlined in Table 2.2.

SIL level	low demand operation	continuous or high demand operation
1	>= 10^{-2} to 10^{-1}	>= 10^{-6} to 10^{-5}
2	>= 10^{-3} to 10^{-2}	>= 10^{-7} to 10^{-6}
3	>= 10^{-4} to 10^{-3}	>= 10^{-8} to 10^{-7}
4	>= 10^{-5} to 10^{-4}	>= 10^{-9} to 10^{-8}

table 2.2 probabilities of failure corresponding to SIL levels in IEC 61508

3 OVERVIEW OF ISO/IEC 15504

3.1 Document set

ISO/IEC 15504: Information technology - Software Process Assessment, is an emerging standard concerned with software process assessment, software process improvement and supplier capability determination. It is the responsibility of ISO/IEC SC7/WG10. Associated with WG10 has been the SPICE (Software Process Improvement and Capability dEtermination) project with responsibilities

- to develop a working draft for a standard for software process assessment.
- to conduct industry trials of the emerging standard.
- to promote the technology transfer of software process assessment into the software industry

The emerging standard was published in 1998 as a Technical Report (type-2) or a TR2. It may be referred to as ISO/IEC TR 15504 and has nine parts

Part 1 : *Concepts and introductory guide*

Part 2: *A reference model for processes and process capability*

Part 3: *Performing an assessment*

Part 4: *Guide to performing assessments*

Part 5: *An assessment model and indicator guidance*

Part 6: *Guide to competency of assessors*

Part 7: *Guide for use in process improvement*

Part 8: *Guide for use in determining supplier process capability*

Part 9: *Vocabulary*

3.2 Process dimension

As mentioned already, the SPICE model has two dimensions, the *process dimension* and the *capability dimension*. The process dimension of the model defines 40 processes which comprise and categorise the software life cycle. These processes belong to five process categories described in table 3.1

Some of the processes are subdivided into component processes. For example

ORG.2 Improvement

may be considered as consisting of

ORG.2.1 Process establishment

ORG.2.2 Process assessment

ORG.2.3 Process improvement

and so on.

Process category	Processes included	Example
Customer-Supplier	processes that directly impact the customer	*CUS.3 Requirements Elicitation*
Engineering	processes that directly specify, implement or maintain the software product	*ENG.2 System and Software Maintenance*
Support	processes that may be employed by any of the other processes	*SUP.1 Documentation*
Management	processes that contain generic practices that may be used by anyone who manages any type of project or process within the software life cycle	*MAN.3 Quality Management*
Organisation	processes that establish the business goals of the organisation and develop process, product and resource assets that will help the organisation achieve its business goals	*ORG.5 Measurement*

table 3.1 - process category

3.3 Capability dimension

The Capability dimension of the 15504 model differs from that of the Capability Maturity Model (CMM) (Paulk 1995), which preceded it, in that it is a *continuous* rather than a *staged* model. The principal difference is

Capability level	Description of capability level	Attributes
level -0 incomplete	the process, if implemented, fails to achieve its process outcomes	none
level-1 performed	the process is implemented and achieves its outcomes	*process performance*
level-2 managed	the process executes in a managed fashion based on defined objectives	*performance management* *work product management*
level-3 established	the process is defined and based on software engineering principles so that it is capable of achieving its process outcomes	*process definition* *process resource*
level-4 predictable	the process performs consistently within limits to achieve its process outcomes.	*process measurement* *process control*
level-5 optimising	the process changes dynamically and adapts to meet current and projected business goals effectively.	*process change* *continuous improvement*

table 3.2 - capability levels

attribute	level	meaning
process performance	1	the extent to which the process achieves the process outcomes by transforming identifiable input work products to produce identifiable output work products
performance management	2	the extent to which the performance of the process is managed to produce work products that meet the defined objectives
work product management	2	the extent to which the performance of the process is managed to produce work products that are appropriately documented, controlled and verified
process definition	3	the extent to which the performance of the process uses a process definition based upon a standard process to achieve its process outcomes
process resource	3	the extent to which the process draws upon suitable resources that are appropriately allocated to deploy the defined process
measurement	4	the extent to which product and process goals and measures are used to ensure that performance of the process supports the achievement of the defined goals in support of the relevant business goals
process control	4	the extent to which the process is controlled through the collection, analysis and use of product and process measures to correct, where necessary, the performance of the process to achieve the defined product and process goals
process change	5	the extent to which changes to the definition, management and performance of the process are controlled to achieve the relevant business goals of the organisation
continuous improvement	5	the extent to which changes to the process are identified and implemented to ensure continuous improvement in the fulfillment of the relevant business goals of the organisation

table 3.3 - attributes

that in a continuous model, such as that defined by 15504, each process is assessed at each of the capability levels from 1-5, whereas in a staged model, such as that defined by the CMM, only a subset of the processes are assessed at each of the levels.

The capability levels of 15504 and their associated attributes are shown in table 3.2. The meanings of the attributes are given in table 3.3.

3.4 ISO/IEC 15504 assessment

A 15504 conformant process assessment will assess a subset of the forty processes in each of the five process categories, for each of the attributes associated with a continuous subset of the capability levels starting at level 1. The result of assessing a single process against one attribute is one of the values *fully*, *largely*, *partially* or *not*.

The 15504 reference model also defines *the capability level achieved by a process* as being the highest capability level for which the corresponding attributes are *largely* or *fully* achieved, while at the same time all attributes corresponding to lower capability levels are *fully* achieved. In this paper it is mainly process capability level with which we will be concerned. If may be worth pointing out that a process may fail to achieve a capability level for one of two reasons

- one of the attributes of the level in question is not *largely* or *fully* achieved
- one of the attributes at a lower level is not *fully achieved*

Some empirical work (SPICE 1999) suggests that the first reason for not achieving a capability level applies about two thirds of the time and the second reason about one third of the time.

4 COMPLEMENTARY STANDARDS

4.1 Comparison of Standards

In this section we provide a brief comparison of the two standards (ISO/IEC TR 15504 and IEC 61508) and try to identify an equivalence between them. Both standards are referenced to ISO/IEC 12207. In the case of 15504, 12207 provides the basis for the process dimension while 15504 adds a capability dimension to the process dimension of 12207. In the case of 61508 the processes of 12207 are extended to cover safety critical applications.

As the capability levels of 15504 increase through CL0 to CL5, the ability of the process to produce high quality

products in a quantitatively controlled manner increases. In a similar way, as the safety integrity increases through SIL1 to SIL4 the need for high quality products with few defects, and therefore low failure rates, increases. It seems reasonable therefore to look for a relationship between the CLs of 15504 and the SILs of 61508. In what follows, we deduce that in order to attain even the lowest SIL, CL2 at least is required for the development process, and we further postulate that to achieve SIL3, CL3 at least is required, and to achieve SIL4, CL4 at least is required. Since 61508 is mainly concerned with the development processes, which correspond to the engineering process category in 15504, we suggest that Table 4.1 shows the minimum CL required of the ENG.1 process (and its component processes) for each SIL. The last two rows of the table are in italic to indicate that they are postulated rather than established through a comparison of the standards documents.

software safety requirement	**minimum capability for Development process**
SIL1	CL2
SIL2	CL2
SIL3	*CL3*
SIL4	*CL4*

table 4.1 61508 SIL related to 15504 CL for the Development process

We begin by showing

i) that CL1 at least is required to produce software of SIL1,

ii) that CL2 at least is required to produce software of SIL1,

the second condition being clearly stronger than the first one.

In order to demonstrate i) we give some insights into the nature of the controls that are applied to the development process by 61508, at all SIL levels.

The standard 61508 requires that the following work products are produced (part 3, table 1): software safety requirements specification; software safety validation plan; software architecture design description; software architecture integration test specification; software/ programmable electronics integration test specification; software architecture design description; software architecture integration test specification; software/ programmable electronics integration test specification development tools and coding standards; selection of development tools; software system design specification; software system integration test specification; software module design specification; source code listing; code review report; software module test results; verified and tested software modules; software system integration test results; verified and tested software system; software architecture integration test results; programmable electronics integration test results; verified and tested integrated programmable electronics; software operation and modification procedures; software safety validation results; validated software; software modification impact analysis results; software modification impact analysis results; software modification log; appropriate verification report - depends on phase; software functional safety assessment report.

It is therefore clear that the ENG.1 Development process must produce the basic work products and so requires the 15504 *process performance attribute* (see table 3.3). This demonstrates that i) above holds.

To demonstrate ii) consider the following. Certain requirements of 61508 map on to particular processes of 15504 (which are shown in brackets). The following are fragmentary examples

- (ENG.1) Development process - " to create a software architecture that fulfils the specified requirements for software safety ... to review and evaluate the requirements placed on the software by the hardware architecture of the E/E/PE safety-related system ... to select a suitable set of tools, including languages and compilers, for the required safety integrity level ... to design and implement software that fulfils the specified requirements for software safety ... to verify that the requirements for software safety (in terms of the required software safety functions and the software safety integrity) have been achieved" (61508, part 3, 7.4.1)
- (SUP.1) Documentation process - "the first objective ... is to specify the necessary information to be documented in order that all phases ... can be effectively performed" (part 1, 5.1); "all relevant documents shall be revised, amended, reviewed, approved and be under ... document control scheme" (part 1, 5.2.11)
- (SUP.2) Configuration management process - "guarantee that all necessary operation have been carried out to demonstrate that the required software safety integrity has been achieved ... apply change-control procedures... document the following information to permit subsequent audit: configuration status, release status, the justification and approval of all modifications ... " (part 3, 6.2.3)
- (SUP.4) Verification process - "to test and evaluate the outputs from a given software safety lifecycle

phase to ensure correctness and consistency ... the verification of software shall be planned ... concurrently with the development, for each phase of the software safety lifecycle, and this information shall be documented ... the software verification shall be performed as planned ..." (part 3, 7.9.2)

- (SUP.5) Validation process - "planning shall be carried out to specify the steps, both procedural and technical, that will be used to demonstrate that the software satisfies its safety requirements ... details of when the validation take place ... the measures (techniques) and procedures that shall be used for confirming that each safety function conforms ... " (part 3, 7.3.2)

These 61508 requirements are seen to imply that ENG.1 must have the *performance management* and the *work product management* process attributes of 15504 (see table 3.3). This demonstrates ii) above

In order to support the postulations contained in the last two lines of table 4.1 we proceed as follows. Appendices A and B of Part 3 of the 61508 standard give a number of highly recommended (HR) development practices (techniques/measures) for each SIL. The deployment of a HR practice is mandatory in the sense that the rationale for not using an HR practice should be detailed during safety planning and agreed with the assessor. Once a HR practice appears at a given SIL level it is also present at higher levels. Some examples HR practices are given below by their entry SIL level:

- SIL1 – use of structured design methods, strongly typed programming language, coding standards, functional black box testing, performance testing, and walk-through/design reviews
- SIL2 – use of a certified language translator, and a library of verified modules, using semi-formal design methods, dynamic testing, static verification, boundary value analysis, performance modelling, control flow analysis, and design reviews
- SIL3 – use of specification and design tools, cause failure analysis, structure-based testing, fault tree analysis, finite state machine modelling, time Petri nets, decision tables, and symbolic execution
- SIL4 – use of formal methods for requirements specification and design, probabilistic testing, formal proofs, performance modelling, and Fagan inspections.

A Development process with capability of CL2 is seen to give an adequate lower bound of capability for performing the HR practices at SIL1 and SIL2.

In the considerations below concerning SIL3 and SIL4 it is assumed that non trivial software is being developed and that the organisational unit in question is working on safety critical systems on an ongoing basis.

The very high reliability required of SIL3 software calls for an almost defect free delivered system. To accomplish this in an effective way, we are postulating in table 4.1 that the ENG.1 Development processes needs to be *managed*. In other words the *process definition attribute* and the *process resource attribute* are both required to be present so that CL3 is required.

Extensive process and product metrics must be collected and statistical quality control employed to assure that SIL4 ultra reliable software is defect free. This is seen to call for the presence of the *process measurement* and *process control* attributes of CL4 for effective deployment of the Development process (ENG.1). Thus our postulate that to attain the SIL4 level, CL4 is required.

.

4.2 Process assessment for safety critical software

One of the purposes of this paper is to explore the adequacy of the 15504 standard when viewed from the standpoint of developing software for safety critical applications, as seen from the perspective of IEC 61508.

The reference model of 15504 describes the processes that form its process dimension in terms of *process purposes* and *process outcomes* together with some explanatory notes. Thus it is more concerned with the *what* rather than the *how* of process execution. It is a 15504 compliant assessment model which provides the *how* (in terms of indicators etc. that must be present) to the reference model's *what*. 61508, on the other hand, is concerned with *how* processes are executed in terms of highly recommended development practices etc. and, to that extent, bears some resemblance to an assessment model (though of course it is not one).

In this section we will look at parts of the 15504 reference model and consider whether its requirements (the what) are compatible with the requirements of 61508 (the how). In doing so we should bear in mind two fundamental aspects of 61508, namely that it

- adopts a systems view rather than a purely software perspective
- employs a quantitative approach to the measurement of safety and utilises the concept of safety integrity levels.

Let us consider some of the issues by referring to parts of the reference model, examining these and passing comment on the underlying issues from a safety critical systems perspective.

ENG 1.1 System requirements analysis and design process

This process is a component of the ENG.1 development process and leads to a number of outcomes three of which are highlighted below in italics with supporting comment in the context of 61508

requirements of the system will be developed that match the customer's stated needs

These requirements will need to include non-functional requirements that address the quantification of safety needs leading to an identification of overall integrity levels. They will need to address safety issues and principles, e.g. no single fault should cause the overall system to fail. They must identify what happens when a failure does occur. In short there needs to be expansion of 'the customer's stated needs' in the context of safety so as to be more specific about the safety requirements. Much of this will typically benefit from preliminary hazard and risk analysis, and all that this entails.

a solution will be proposed that identifies the main elements of the system

Underlying this, there is often a considerable range of steps that need to be carried out. Having identified hazards and the associated risks, it is common practice to consider risk removal, risk reduction and risk control. These and other matters augmented by decisions about implementation in hardware or software together with the allocation of responsibilities for safety will have a huge bearing on the design of the overall system.

the requirements will be allocated to each of the main elements of the system

The requirement to quantify safety and to identify safety integrity levels will mean that performance figures need to be allocated to software and to the other components. The associated safety case should demonstrate, perhaps by a number of independent means (to increase confidence in the results), that the allocation and the attainment of the performance figures will indeed lead to a system that achieves the required levels of safety. There is then a subsequent requirement to demonstrate that the software indeed meets its performance level. This has implications for the whole of the rest of the software engineering process. In addition, of course, there is the related question about whether the underlying hardware and the system software (including compilers and other tools) meet the necessary levels of quality.

ENG.1.3 Software design process

Again, the 15504 Reference Model provides a set of outcomes that need to be achieved at this stage and again some observations can be made about these in the context of 61508.

a detailed design will be developed that describes software units that can be built and tested

Implicit in the IEC 61508 standard is the belief that different safety integrity levels are achieved or facilitated in an efficient and cost effective manner by choosing different and appropriate design approaches and methods. Thus the software design process becomes crucial. Apart from addressing issues such as fault tolerance and incorporating this into the design, there is the implication that any implementor requires an understanding of the relative strengths and weaknesses of design methods generally and is able to choose an appropriate cocktail of these to suit the needs at hand.

consistency will be established between software requirements and software design

Here there is the obvious consistency involving the functional specification with the requirement to look at error situations but there is also the quantification of safety integrity levels and the implied reliability levels that impose all kinds of demands on the implementor.

ENG.2 System and software maintenance process

Using the same approach as above, let us consider part of this section of the 15504 Reference Model.

the impact of organisation, operations and interfaces on the existing system in operation will be defined

Here it is important to consider the impact on safety: prior to any change an impact analysis is needed; part of this is likely to involve hazard and risk analysis again, i.e. a fundamental reassessment of the possible impact of any change that is made to the system.

As mentioned earlier, the development of safety critical systems is a systems issue, i.e. it is not solely about software. It is about hardware, software, people issues, tools, management practice, and so on. Typically safety is a descriptor applied to an entire system and it is important for the engineering process to reflect this. So there are systems engineering issues that address hardware, software, communications, people issues, etc. The concern here has focussed on the software dimension.

The examples given above should be regarded as illustrations of insights needed by anyone using the 15504 reference model for the design and development of safety critical systems. Our conclusion is that the reference model does indeed supply an appropriate framework, but that it would benefit from the development of an associated Assessment Model that would specifically address the stringent demands of the safety critical systems area. This would then capture the set of indicators - the base practices, the management practices and the work products - that typically are used to provide

a demonstration that best practice has been followed, and to provide appropriate evidence and assurance in the context of safety critical systems development. This view is consistent with the observations made earlier on the specific parts of the reference model. The assessment model would, of course, need to be aware of the SIL level involved since the work products etc will depend on the activities that were highly recommended.

One conclusion that does stem from much of this is the question of the sophistication of the user or customer. Since much of the reference model is phrased in terms of this, it is a rather crucial parameter.

5 EMPIRICAL EVIDENCE

Empirical evidence associating process capability and safety criticality may be found in

- data from the SPICE trials
- the COCOMO II model

5.1 Data from the SPICE trials

The development of the 15504 standard is unique in two ways

- the 'technical report' route has been used to develop the standard
- the process assessment framework is being extensively trialled at each stage of development

The SPICE trials are in three phases, the first two of which have been completed. The results of phase-1 of the trials have been published in (Woodman 1996), the preliminary results of phase-2 have been published in (Hunter 1998) and the final analysis of phase-2 in (SPICE 1999). Phase-2 of the trials were based on the document set known as SPICE version 2 (the Preliminary Draft Technical Report, PDTR) which is slightly different from the TR (Technical Report) version described in section 3 of this paper. However, the changes are mainly in the process dimension and not at all in the capability dimension which is what we will be most concerned with here.

The data collected from phase-2 of the SPICE trials were not restricted to the results of the trial assessments. A number of other types of data were also collected such as

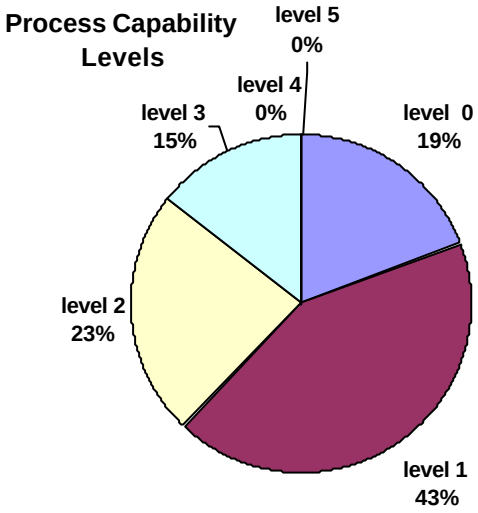
- data describing the characteristics of the Organisational Unit involved in each trial
- data describing the characteristics of the projects involved in each trial
- feedback on the assessment process from the sponsors and assessors

In particular data was collected concerning the criticality of the software produced by each of the projects. The degree of criticality for each project with respect to *safety, economic loss* and *environmental impact* was collected. For safety critical projects, the risk was assessed to be at one of four levels

1. small damage to property
1. damage to property (few people injured)
2. threat to human lives
3. many people killed.

For the purposes of this section we will describe projects at levels 3 or 4 to be of *high criticality* and those at levels 1 and 2 to be of *low criticality*. In the same way we will consider *processes* associated with projects of high criticality to be processes of *high criticality*, and processes associated with projects of low criticality to be processes of *low criticality*.

As far as the capability ratings were concerned, virtually all processes were assessed up to and including level 3 therefore in what follows it makes sense to group together those processes which were at level 3 (at least) and those processes which were at level 4. There were no processes assessed at level 5.



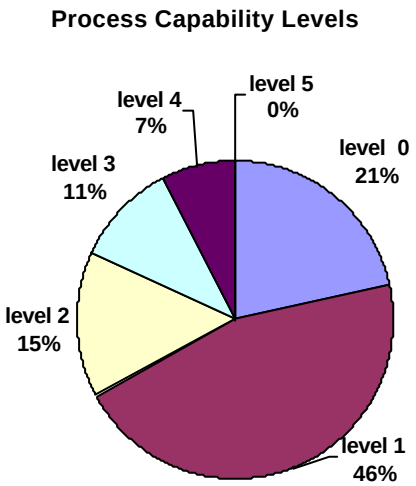
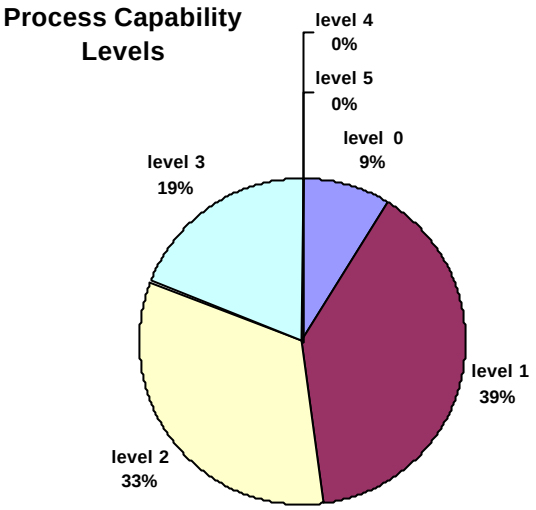


figure 5.1 - process capability levels of 447 non-highly safety critical process instances (at top) and the 121 which were highly safety critical (at bottom)



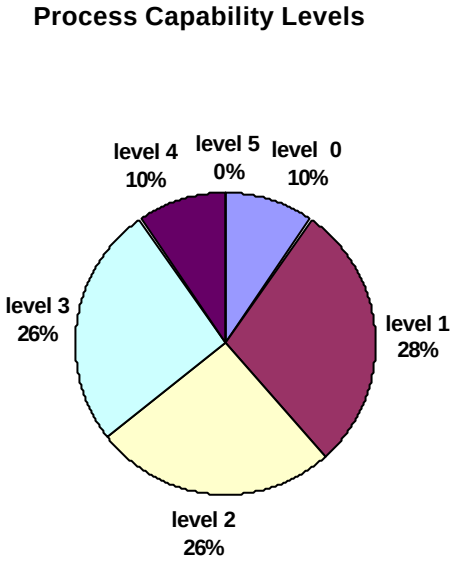


figure 5.2 process capability levels for the 162 non-highly safety critical process instances in ENG category (at top) and for the 31 highly safety critical process instances in ENG category(at bottom)

In phase 2 of SPICE some 70 process assessments were performed involving 169 projects and a total of 691 process instances. Of the 691 process instances, 568 had safety criticality information associated with them. 121 of these process instances were of high safety criticality and 447 were not of high safety criticality. Figure 5.1 compares the distribution of capability levels of the non highly safety critical processes with the distribution of capability levels of the highly critical process instances. The highly safety critical process instances had a higher proportion of process instances with a capability level of 3 or 4 than the non-highly safety critical process instances (18% as against 15%).

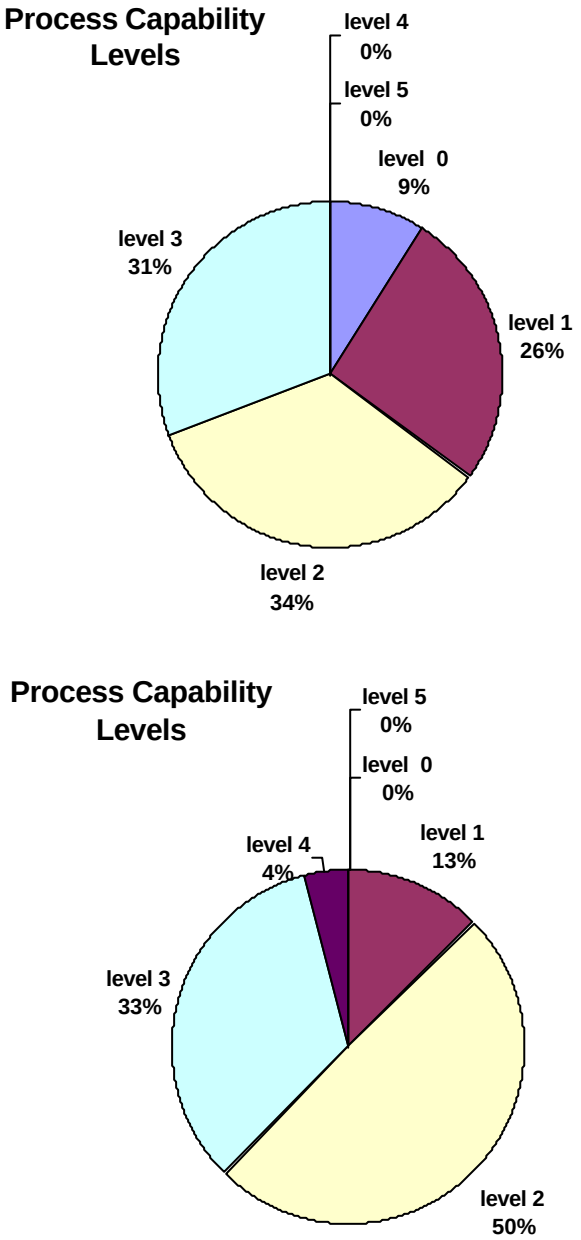


figure 5.3 process capability levels for the 157 non-highly safety critical process instances associated with medium and large sized projects (at top) and for the 24 highly safety critical process instances associated with medium and large sized projects (at bottom)

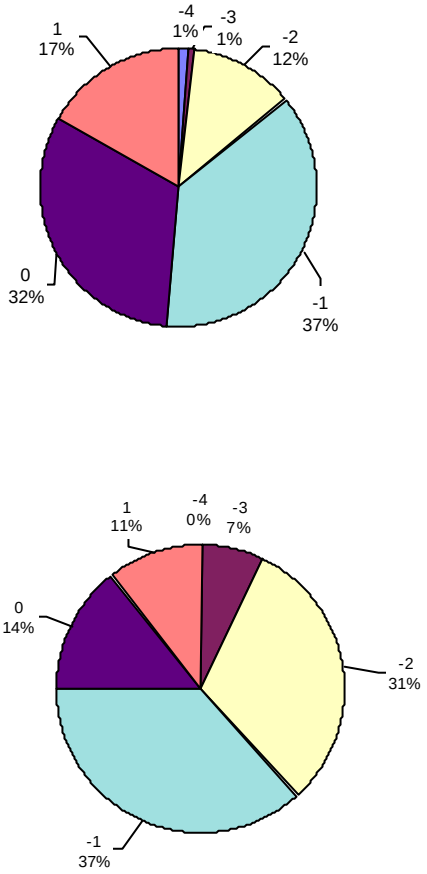


figure 5.4 Gap values for the engineering category processes (top) and those of the other categories (bottom)

The highly safety critical process instances also have a higher proportion of processes with capability levels of 0 and 1 than the non-highly safety critical process instances (67% as against 62%).

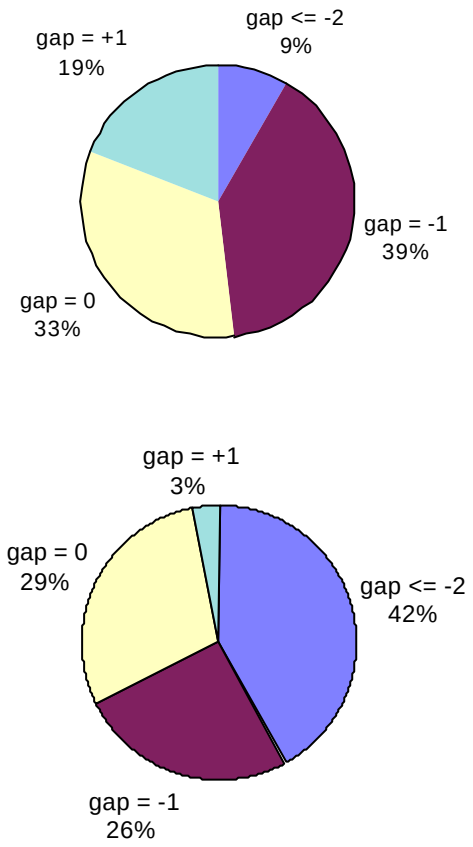


figure 5.5 Gaps for SILs 1-2 (top) and SILs 3-4 (bottom)

If, however, instead of considering all the process categories defined by the SPICE project, we confine ourselves to the processes in the Engineering category, a different impression is given - this seems a reasonable thing to do since the engineering processes are the processes emphasised by the 61508 standard. Figure 5.2 compares the process capabilities of the 162 process instances which were not highly safety critical in the ENG category with the 31 which were highly safety critical. For each of the capability levels 2-4, the percentage of processes with capabilities equal or above that level is consistently greater for the highly safety critical process instances in the ENG category than for the non-highly safety critical process instances in this category. Also if we consider only medium and large sized projects (defined as peak staff >14) we find that the capability of processes associated with high safety critical projects is consistently higher than for non-highly safety critical projects, see figure 5.3.

In table 4.1 a mapping between the SIL's of 61508 and the CL's of the 15504 standard was proposed. Using this mapping and equating SIL levels 1-4 with the Safety Levels collected during the SPICE trials we can define the 'gap' between CL required for the degree of criticality of the software and the actual CL as given by

$$gap = CL_{actual} - CL_{desired}$$

so that (ideally) the gap should be zero, or even positive. As might be expected from previous analysis, processes in the engineering category tend to have a more favourable value of 'gap' than processes in other categories. See figure 5.4

Figure 5.5 shows how the gaps relate to the SIL's. Due to the nature of the problem, the gaps cannot exceed 2 in the case of SILs1-2 and cannot exceed 4 in the case of SILs3-4. It is perhaps worrying to see that in developing software for SIL3-4 systems, around 42% of the process instances had an associated 'gap' of -2 or more.

5.2 COCOMO II

The COCOMO (COnstructive COst MOdel) model has been employed for almost two decades for effort and schedule estimation (Boehm 1981). Recently the model has been enhanced considerably. The enhanced model is known as COCOMO II (COCOMO II 1999). The COCOMO II model is based on empirical results and is driven by over 20 parameters. Two of the parameters are of particular interest here. They represent the reliability (RELY) requirement of the product and the process software process maturity (PMAT). Keeping all other parameters constant the effect of these two parameters according to the model is

$$PM = a * RELY * Size^{(b + 0.01*PMAT)}$$

where PM is the effort estimate in person months, a and b are computed from the remaining parameters and the estimated Size is measured in thousand noncommentary source lines of code (KSLOC).

The range of values for RELY is from “very low” (0.82) when the software failure will cause slight inconvenience to the “very high” (1.26) when failure causes risk to human life. For the development of safety critical software in the 61508 context, it would be of interest to extend the COCOMO model to take the SIL levels into account. Clearly both the RELY factor and the SIL level affect the development effort in a measurable way.

In (COCOMO II.1999) the process maturity factor PMAT is estimated to have the values shown in table 5.1.

	VLO	LO	NOM	HI	VHI	XHI
CMM level	CMM Level 1 (lower half)	CMM Level 1 (upper half)	CMM Level 2	CMM Level 3	CMM Level 4	CMM Level 5
PMAT	7.80	6.24	4.68	3.12	1.56	0.00

table 5.1 The PMAT values

The procedure for determining the PMAT value may be based on a process assessment using the Capability Maturity Model (CMM) developed by the Software Engineering Institute. (Paulk 1995) While the current version of the CMM (version 1.1) differs from ISO/IEC 15504 by being a staged model rather than a continuous model, and being concerned with the maturity of an organisation rather than with the capability of an organisation's processes, the Integrated CMM (CMMI 2000) now emerging provides both a staged an a continuous view of the model. The continuous view of the model is similar to 15504 and, in its final form, is likely to be compliant with it.

An alternative way of determining the PMAT value is to rate the frequency of achievement the goals associated with individual software processes and to take a weighted average of those ratings (COCOMO II 1999).

The effect of PMAT on its own can be seen by normalisation with respect to the effort at Level 5:

$$PM/PM_5 = Size^{(0.01*PMAT)}$$

Table 5.2 shows the relative effort as computed by this equation for a range of project sizes:

The table shows the effect of the PMAT scaling factor on COCOMO II effort estimation. The development effort is most effective if the development organisational unit is operating at CMM Level 5. Improvement in the CMM level decreases the effort. For example, the effort for a project of size of 100 KSLOC is seen to decrease by some 7%-10% when the CMM level is improved by one.

Table 5.2 can be taken to give an indication of how 15504 capability will effect productivity. Increased process capability will increase productivity in any given project. This is more pronounced for larger projects.

CMM Level and PMAT value	10 KSLOC	100 KSLOC	1000 KSLOC	10000 KSLOC
CMM L1-low 7.80	1.20	1.43	1.71	2.05
CMM L1-high 6.24	1.15	1.33	1.54	1.78
CMM L2 4.68	1.11	1.24	1.38	1.54
CMM L3 3.12	1.07	1.15	1.24	1.33
CMM L4 1.56	1.04	1.07	1.11	1.15
CMM L5 0.00	1.00	1.00	1.00	1.00

table 5.2 Relative development effort as a function of PMAT and Size

It is to be noted that the COCOMO parameters represent statistical best fit to the COCOMO model using actual metrics and characteristics for a large number of projects. Clark used the data from 112 projects ranging in size from 2.6 to 1264 KSLOC with project effort ranging from 6 to 11,400 Person Months in his investigation of the effect of PMAT on the COCOMO model. (Clark 1997).

6 CONCLUSIONS

The 61508 standard provides requirements and recommendations for the development of safety critical software components while the 15504 standard provides a process assessment framework from which the capability levels of software processes may be determined. A relationship between the SILs of 61508 and the 15504 CLs has been suggested for the Development process (table 4.1) : SIL1 and SIL2 require CL2, SIL3 requires CL3, and SIL4 requires CL4 as a minimum capability level for effective software development.

The 15504 trials show (fig. 5.2a) that for non-highly safety critical engineering processes (SIL1-2) 52% have a capability rating of *CL2* or higher. For the highly safety critical engineering processes (SIL3-4) 36% have a capability rating of *CL3* or higher (fig 5.2b). So there is a considerable gap between the necessary capabilities postulated and the actual capabilities, and the gap widens for the higher criticality requirements even though the capabilities are markedly better. For large projects (peak staff>14) the situation is better, with 65% compliance for the non- highly safety critical and 37% for the highly safety critical process instances (fig. 5.3). (It should be noted that the SPICE trials do not in any way represent an industry bench-marking as the participation in the trials is on a voluntary basis.)

An interesting view on the effect of process maturity is given by the COCOMO II estimation model. From table 5.2 it can been seen that the development effort decreases

with increasing CMM maturity levels. This effect gets more pronounced as the project size increases, which is reasonable as larger projects will need relatively more control than smaller ones. This suggests that instantiation of software processes should be project size dependent and, in a similar way, perhaps process assessment should also be project size dependent. This might provide a way of relaxing the 15504 capability criteria for small projects.

Table 6.1 shows the possible 'gaps' between the required 15504 Capability Level and the corresponding Safety Integral Level.

SIL\CL	CL0	CL1	CL2	CL3	CL4
SIL1	-2	-1	0	+1	+2
SIL2	-2	-1	0	+1	+2
SIL3	-3	-2	-1	0	+1
SIL4	-4	-3	-2	-1	0

table 6.1 Possible 'gap' sizes

The worst situation is when the gap is -4 corresponding to an organisation using development processes at CL0 to produced software intensive systems with a required SIL of 4. The question may well be asked whether it is at all reasonable for an organisation to develop safety critical software where the gap is larger than -2 say. It might be reasonable if the project is very small - but even then, the development would not comply with the requirements of the 61508 standard. It is well known that many software projects are never completed and this may, in part, be because the processes belonging to the organisations involved do not have sufficient capability for the job in hand.

It would appear, therefore, that current ideas on process assessment and improvement have a contribution to make to the development of safety critical software in the following senses

- there is evidence from the SPICE data that, at least for engineering processes, software producers tend to use processes of high capability in connection with the development of software for safety critical systems
- there is evidence from COCOMO II that increased capability of the processes involved will decrease the effort and cost required to produce very reliable systems

The main conclusions of the paper are

- the correspondence postulated between the capability levels of 15504 and the safety integrity levels of 61508 suggest that current engineering practice is not adequate for the production of software for safety critical systems. (The alternative conclusion that the correspondence postulated needs to be relaxed somewhat is difficult to sustain.)
- there is a good case for the production of an ISO/IEC 15504 conformant assessment model intended specifically for the assessment of software processes to be used in the production of software for safety critical systems.

REFERENCES

Billings, C., Clifton J., Kolkhorst B., Lee E., Wingert W. B. 1994. 'Journey to a mature software process', *IBM Systems Journal*, volume 33, no. 1.

Boehm, B. W. 1981 *Software Engineering Economics*, Prentice Hall.

Clark, B.K. 1997. *The Effects of Software Process Maturity on Software Development Effort*. Ph.D. Thesis, Graduate School, University of Southern California.

CMMI, 2000 http://www.sei.cmu.edu/cmm/cmms/cmms.integration.html

COCOMO II 1999. http://sunset.usc.edu/COCOMOII/cocomo.html.

Dorling, A. and Wickberg, H. 1999. ' Successful Process Improvement: New Approaches and Best Practice Experience'., *Proceedings Sixth European Conference on Software Quality*, Vienna.

Hunter, R. B. 1998. 'SPICE Trials Assessment Profile', *IEEE Software Process Newsletter*, number 12.

IEC 61508 1998-2000. *Functional safety of electrical/electronic/programmable electronic safety related systems*, International Electrotechnical Commission, Geneva.

ISO 9001 1994. Quality systems. Model for quality assurance in design, development, production, installation, and servicing.

ISO/IEC 12207 1995. *Information Technology—Software Life-Cycle Processes*, ISO/IEC.

ISO/IEC, TR 15504 1998. 1-9, *Information technology – Software process assessment – Parts 1-9*, ISO/IEC.

Leveson, N. 1995 *Safeware: System Safety and Computers*, Addison Wesley.

Paulk M. C. , Weber C. V., Curtis, B., Chrissis, M. B. 1995. *The Capability Maturity Model: Guidelines for Improving the Software Process*, Addison Wesley.

Redmill, Felix 1998. IEC 61508 - principles and use in safety management, *Computing and Control*, 9,5, 205-213, October.

SPICE 1999. *Phase 2 Trials Final Report*, volume 1, SPICE project.

Wood, A. 1996. 'Predicting Software Reliability', *Computer*, Vol. 29, No. 11.

Woodman, I., Hunter, R. 1996. 'Analysis of Assessment Data from Phase One of the SPICE Trials, *IEEE Software Process Newsletter*, number 6.