04－R 005

リアルタイムシステムのソフトウェアの安全性に関する調査研究

―ソフトウェア安全性調査研究ワーキンググループ最終報告書―

平成５年３月

JIPDEC

財団法人 日本情報処理開発協会

序に代えて

近年、コンピュータリゼーションとともに、クリティカル分野におけるコンピュータおよび安全機能を担うセーフティクリティカルソフトウェアに対する関心が高まっております。ソフトウェアは、長所としての柔軟性がコンピュータの応用領域を拡大してきましたが、信頼性確保や保守の面から、それが短所としても指摘され、クリティカル分野では、人命や財産に重大な影響を及ぼすことが懸念されるようになってきました。

当協会では、このような状況から、平成3年度より2年計画で，特に制御系のシステムなどリアルタイムシステムにおける安全性の問題を取り上げ、「ソフトウェア安全性調査研究ワーキンググループ」を設けて調査研究を実施いたしました。

平成3年度は、ソフトウェアの安全性に関する基本概念、フェールセーフシステムにおけるソフトウェア問題、大規模ソフトウェアの信頼性確保の問題、ソフトウェア安全性に関する規格化の動向（代表的なものとして、ＩＥＣ規格案「プログラマブル電子システムの機能別安全性」）、鉄道、航空宇宙、原子力発電におけるソフトウェアの信頼性・安全性対策などについて調査研究を行いました。

第2年目の本年度は、ソフトウェアの安全性に関する国際規格案ISO/IEC JTC1/SC7 N917：「産業用安全関連システムの分野で利用されるコンピュータ用ソフトウェア」の規定内容（要求事項）についての検討および解説の作成、鉄道、航空宇宙、原子力発電の分野におけるソフトウェア安全性規格（または規格案）と同国際規格案との規定内容の比較評価などを行いました。この調査研究の結果、クリティカル分野におけるセーフティクリティカルソフトウェアの開発に関する規格では、ソフトウェアの安全性をより向上させるために今後は形式的ソフトウェア開発法とソフトウェア構成管理の採用を推奨しております。わが国においても早急にこの分野の対応策が要請されるところであります。

本報告書は、平成4年度の成果を「リアルタイムシステムのソフトウェアの安全性に関する調査研究」として取りまとめたものです。

最後に、本調査研究にあたって、ご指導くださいました向殿正男教授（明治大学理工学部）、片山禎昭氏(東芝アドバンストシステム(株))、峰尾欽二氏（日本ユニシス(株)）、松原友夫氏（コンサルタント）をはじめ、ご協力を頂きましたワーキンググループの委員各位に厚く感謝の意を表します。

平成5年3月

財団法人　日本情報処理開発協会

ソフトウェア安全性調査研究ワーキンググループ　名簿

（順不同、敬称略）

(1)	内海正文	三菱原子力工業(株)　軽水炉統括部 電気計装設計部　計装制御チーム
(2)	太刀川寛	日本信号(株)　与野事業所 鉄道信号事業部　信号技術部　開発グループ　課長
(3)	中村英夫	(財)鉄道総合技術研究所 中村研究室（運転制御）　室長
(4)	松尾谷徹	日本電気(株)　C＆C第一システム開発本部 ソフトウェア生産技術部　技術課長
(5)	松本甲太郎	科学技術庁　航空宇宙技術研究所 数理解析部　データ処理研究室　主任研究官
(6)	鍛治勝三	(財)日本情報処理開発協会 開発研究室　主任研究員

総　目　次

序に代えて ---------- i

ソフトウェア安全性調査研究ワーキンググループ　名簿 ---------- ii

第１部　ソフトウェア安全性規格の概要 ---------- 1

Ⅰ．ソフトウェア安全性規格の必要性と課題 ---------- 3

Ⅱ．ソフトウェア・セーフティとその規格化の現状 ---------- 13

第２部　ＩＳＯのソフトウェア安全性規格（案） ---------- 39

Ⅰ．規格（案）の規定内容 ---------- 41

Ⅱ．規格（案）の解説 ---------- 114

第３部　特定分野のソフトウェア安全性規格との比較評価 ---------- 177

Ⅰ．航空宇宙分野 ---------- 179

Ⅱ．鉄道分野 ---------- 187

Ⅲ．原子力分野 ---------- 197

第４部　文献紹介 ---------- 203

Ⅰ．ＩＥＣ暫定規格の概観：プログラマブル電子システムの機能別安全性 ---------- 205

Ⅱ．信号保安システムに関する相互受入れ ---------- 230

Ⅲ．セーフティ・クリティカル・コンピュータ・ソフトウェア用の英国防規格の開発の根本的理由 ---------- 247

Ⅳ．潜在的に危険な産業でのソフトウェア安全性検定 ---------- 267

Ⅴ．参考文献一覧 ---------- 289

付録　ソフトウェア安全性調査研究ワーキンググループの議事内容一覧 --- 329

第１部　ソフトウェア安全性規格の概要

中　目　次

Ⅰ．ソフトウェア安全性規格の必要性と課題

Ⅱ．ソフトウェア・セーフティとその規格化の現状

I. ソフトウェア安全性規格の必要性と課題

細　目　次

1. はじめに

2. 安全性検定とソフトウェア導入形態

3. ソフトウェア安全性規格の必要性
   (1) 複雑性
   (2) 誤り感度性
   (3) テストの困難性
   (4) 故障の相関性
   (5) 専門資格の欠如
   (6) プロセスの感度性

4. ソフトウェア安全性技術の現状
   (1) 検証／評価
   (2) ソフトウェア構造設計技法
   (3) ソフトウェア開発／管理プロセスの改善と維持
   (4) 設計ダイバーシティ
   (5) 形式的方法

5. ソフトウェア安全性規格の課題
   (1) 既に安全性規格を持っている分野
   (2) ソフトウェアにより安全性の問題が生じる分野
   (3) 安全性技術の課題

I. ソフトウェア安全性規格の必要性と課題

1. はじめに

航空・原子力・鉄道・自動車・医療など潜在的に危険な分野におけるシステムは、信頼性だけでなく安全性が強く要求されている。信頼性と安全性は異なった概念である。信頼性とは装置が機能を果たすことであり、安全性とは装置の故障や誤操作など予期しないハザードから、人間の死傷または財産の損失を与えるような状態が起こらないことである。

安全性の考え方はソフトウェアが導入されることとは別に、航空・原子力・鉄道など機械系システムにおいて発展してきた。それぞれの分野において何らかの安全性に関する基準を持っており、新たなシステムを稼働させるには、安全性基準を達成し、かつ監督機関が行う許認可を受ける必要がある。

一方、コンピュータとソフトウェアを用いた機械制御技術は急速な勢いで進歩し、導入されつつある。従来から行われている電気機械系による装置制御と比較し、コンピュータ制御は、飛躍的な制御の自由度と柔軟性を経済的に実現するためである。人の関わる大型の機械装置を制御することは、潜在的な危険性を持っており、この分野におけるソフトウェアは、システムの安全性に強く関与することになる。

本研究WGは、平成3年度から開始し、ソフトウェア安全性について調査・研究を行ってきた。平成3年度は主にソフトウェア安全性について技術的な側面を中心に調査を行い、一部ソフトウェア安全性規格についてもサーベイを行い報告書にまとめた。平成4年度はソフトウェア安全性規格に的を絞り、今回報告書としてまとめた。

ソフトウェア安全性の問題を、安全性規格に絞った背景は、安全性に関する水準が産業固有の安全性基準により決まっているという現実と、安全性の問題がソフトウェア単独の問題ではなく、システム的に捕らえる必要があるからである。システムの一部としてソフトウェアの安全性を捕らえることは、当該産業分野における安全性規格の中に、ソフトウェア安全性をどのように取り込むかの問題で

もある。

航空・原子力・鉄道など、それぞれの産業分野における安全性基準の中に、ソフトウェア安全性基準が必要であり、また先進的な分野ではすでに基準が作られている。しかし、ソフトウェア安全性に関する技術は、未成熟でかつ様々な技術があり、それぞれの産業分野で個別に検討するのは不合理である。

特に問題と思われるのは、ソフトウェアの導入が安全性規格の整備より早く行われる分野である。ソフトウェア技術の導入は機械系の制御の自由度と柔軟性を可能にし、その機能的な側面が経済的な競合の中で過度に進行する危険性をもっている。産業用ロボット、土木機械、遊園地の大型遊具などいたるところにコンピュータ制御による機械系システムは導入されつつある。

このような背景の中で、原子力プラントなど先進的な分野での実績を基にした、ソフトウェア安全性の国際規格原案が審議されている。本報告書の第2部で紹介する国際規格原案は、個別の産業分野における安全性規格にソフトウェア安全性を引用するためのメタ規格である。本研究WGはソフトウェア安全性規格のテーマとしてIEC/TC65/SC65A/WG9で作成された65A(Secretariat)122 “Software for Computers in the Application of Industrial Safety-Related Systems” を取り扱っている。この規格原案はISO/IEC JTC1/SC7でも審議されている。なおこの規格原案は現在審議中のものであり、厳密な意味で規格ではない。ソフトウェア安全性規格を検討する上での研究対象として取り上げている。

本報告書の第3部では、既にソフトウェアを導入している先進的な分野における安全性規格の現状について報告する。

2. 安全性検定とソフトウェア導入形態

航空・原子力・鉄道など潜在的に危険な産業では、安全性の面から何らかの検定を受けることが必要とされている。例えば民間航空機の耐空性（airworthiness）や原子力発電プラントにおける許認可（licence）がこれに相当し、国家あるいは第三者機関などの監督機関が検定を行っている。これらの検定は、コンピュータやソフトウェアが導入されるずうっと以前から行われている。

安全性検定の最も安全な運用は、「安全が実証されていないものは安全でない。」とする考え方である。ソフトウェアの安全性検定に関し、学界や産業界の明確な意見の一致はなく、ソフトウェア検定とは何であり、どのように実施すべきか意見が別れる。このような状況の中で航空・原子力・鉄道などの分野におけるソフトウェア導入は、安全性検定や安全性規格にソフトウェアをどのように取り込むかの議論と共に、非常に慎重に行われてきた。

一方、技術的な見地から、リレーやアナログ電子回路による制御装置と、コンピュータ制御を比較すると、信頼性・経済性など広範囲においてコンピュータが優れていることは明かである。またコンピュータ技術を利用する以外に実現出来ない複雑な操作上の機能（例えば誤操作に対するチェックやリカバリー機能、あるいは操作補助情報の提供など）があり、コンピュータの導入はシステムの信頼性・安全性向上に大きく寄与する面がある。

安全関連分野におけるソフトウェア導入は、導入を慎重に行おうとする既存の安全性検定と、ソフトウェアを導入するメリットの中で慎重に行われている。詳細な状況は第3部で説明するが、原子力や航空、鉄道などの分野で確実に実績を上げている。

ソフトウェア安全性に対する安全性検定や安全性規格を調査してみると、それぞれの国家、あるいは産業分野によって安全性の考え方が多少異なっており、別々に開発される傾向が強い。例えば原子力プラントであれば英国、米国、仏国により評価基準やその考え方が異なっている。鉄道分野では欧州の中でも、国によってそれぞれ異なる状況である。

対象となる分野によってシステムとしての安全性検定が異なることは有り得る、しかしソフトウェア安全性検定として見れば共通問題でもある。安全関連ソフトウェア検定に関するメタ規格があれば、多くの産業界にとって非常に大きなメリットが生まれる。これは品質保証をソフトウェアに適用するためのガイドである、ISO 9000-3が果たしたような役割である。第2部で取り上げたISO/IEC JTC1/SC7 N917はまさにこのような背景から生まれたものである。この規格原案は審議中のものであり、確定したものではないが良く審議されている。

この規格原案は、ソフトウェア検定のためのメタ規格として位置付けられるものである。実際の適用は、規格をテーラリングし、対象となる産業個別分野の規格で用いられることを想定してある。この規格の基盤は原子力プラントなど非常に高度な安全性を要求される分野の実績と、今後主流となる形式的仕様化技術（formal specification）など新技術の２つの面から書かれている。もう一つの特徴は安全性の水準として４段階のインテグリティ・レベル（Integrity Levels；その後、独立の国際規格として審議されることになった。）を用い、規格の適用範囲を広げている。

３．ソフトウェア安全性規格の必要性

従来からある、産業分野それぞれの安全性検定の中に、ソフトウェアの安全性検定が別途必要となるのは、ソフトウェアが他の構成要素と大きく異なるからである。安全関連システムにおけるソフトウェアについて考えるには、先ずソフトウェアがどのような形態でシステムに組み込まれているかを明らかにする必要がある。安全関連システムを一般化すると、図１に示すように制御下にある装置（EUC: Equipment Under Control）とそれを制御する安全関連システム（SRS: Safety-Related System）から構成される。このＳＲＳまたはその一部としてコンピュータとソフトウェアが使用されている。

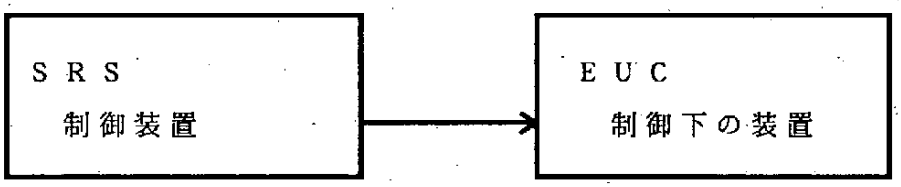


図１　安全関連システムの構成

安全性の設計は図２に示すような手順で行われる。まず装置が持っている潜在的な危険についてハザード解析を行い、識別されたハザードのリスク水準を決定するリスク評価を行う。こうして求められたリスクに対しＳＲＳの安全性要求仕様が決められる。ＳＲＳに対する安全性要求仕様には２つの要素がある。一つは機能面からの要件であり、例えば誤操作に対する保護機能や、フェールセーフの機能に相当する。もう一つはインテグリティ面からの要件であり、それは信頼性

の要件や故障時の他への影響に関する要件（故障状態）などに対応する。インテグリティ面からの要件は図に示すように、機能面の要件に対しても作用する。

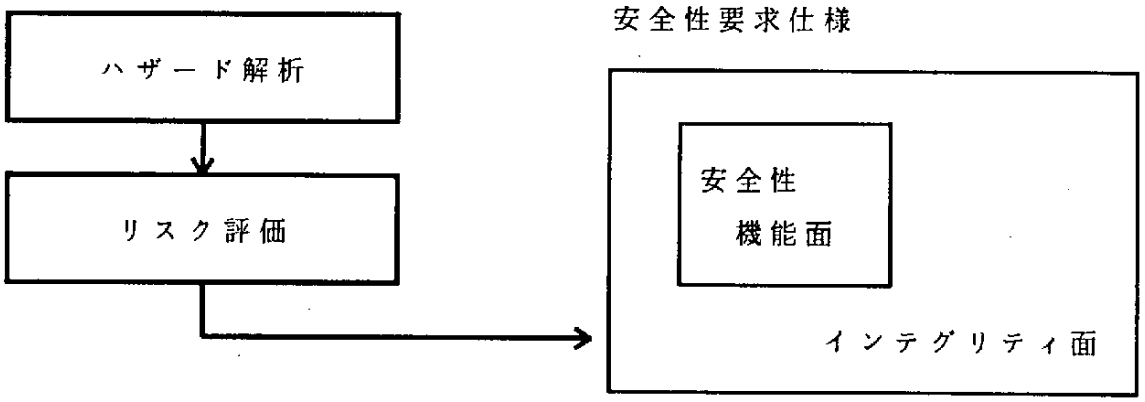


図２　安全関連システムの要求定義

ソフトウェアに対する安全性の機能要件は、ソフトウェア以外の手段による安全性機能要件と同じように取り扱うことが出来る。ソフトウェア固有の問題となるのは、主としてインテグリティ面からの要件である。インテグリティから見たソフトウェアの主な問題は、信頼性と故障状態の２つの特性である。

信頼性とは、ソフトウェアの欠陥による所望機能が動作しないことの頻度であり、故障状態とは、ソフトウェアの故障がどのような振る舞いをし、他にどんな影響を与えるかの特性である。従来部品、例えばリレーの故障状態は接点が接触しないか、あるいは切断出来ないかの２つのモードであり、統計的な根拠から動作時の頻度を予測することが可能である。一方、ソフトウェアの故障状態は非常に多様で計り知れない面がある。まったく関係が無いと思われる２つのプログラムの一方が、作業用メモリーを異常に消費するなどの障害で、他のプログラムの動作を阻害させることもあり得る。ソフトウェアの故障状態に関する、ソフトウェア工学上の研究は、ほとんど進んでいないのが現状である。

信頼性と故障状態の２つの問題と絡んでソフトウェアには次のような特徴がある。

(1) 複雑性

ソフトウェア・ロジックはハードウェアの論理回路と比較しても遙かに複雑である。1,000個のリレーによるロジックより、たった１ＫＢのプログラムの方が遙

かに複雑なロジックを作っている。

(2) 誤り感度性

ハードウェア機器は、誤り時の動作範囲が限定される。ソフトウェアの場合、例えばデジタル信号処理における1ビット誤りであっても、誤りの原因とは別に、取り扱うデジタル信号の位置だけで、出力の正負が逆転したり、何千倍もの大きさの誤った制御信号を発生させ得る。

(3) テストの困難性

ソフトウェアに対する完全なテストは、その複雑性から不可能である。しかも処理の連続性が保証されていない。処理の連続性とは、例えば電圧に対するセンサーが設定された域値を検出する場合、域値の前後で確認すればテストとして十分である。ところがソフトウェアで域値を処理する場合、域値で動作することを確認しても、すべての入力で動作が保証されることにはならない。ソフトウェアでは10^{100}程度の組み合わせはいくらでもある。

(4) 故障の相関性

高信頼性技術として冗長構成が用いられる。ソフトウェアに対して冗長構成を実現するためには、設計行為そのものの2重化が必要となる（設計のDiversityと呼ばれる）。しかも、そのようにして設計を多重化したとしても、設計者は誤り易いところで、同じような誤りを起こす確率が高い。よって冗長技術による信頼度向上にも限界がある。

(5) 専門資格の欠如

要求されたインテグリティ水準を実現するための、ソフトウェア工学上の技術、およびその技術を行使出来る専門技術者に対する専門資格が成熟していない。それどころか、インテグリティ水準を満たすために必要な技術、例えば形式的仕様化技法などについて見れば、現状のソフトウェア技術者にとってそれが難解である理由から、導入に消極的な状況さえうかがわれる。

(6) プロセスの感度性

ソフトウェアのインテグリティ水準は、ハードウェアの構成要素のように、設計あるいは、部品選定段階だけで決まるだけでない。コーディングやテスト、さらに保守などソフトウェア・ライフサイクル上のプロセスがことごとく関与している。コーディング段階、あるいは保守段階において、たった一人のプログラマ

が、誤った1行のプログラムを行うことにより、システムの安全性を壊滅的な状態にすることも可能である。

ソフトウェアはこのような複雑な特性を持つており、少なくともハードウェアと同じような方法で安全性検定を行うことは出来ない。つまりソフトウェアに対し、システムの一構成部品として、その機能を直接確認する方法でソフトウェア検定を行うだけでは不十分である。また、ソフトウェアの開発プロセスだけに基づいてソフトウェア検定を行うのも妥当ではない。

4. ソフトウェア安全性技術の現状

ソフトウェア安全性を決定するのは、ソフトウェアの信頼性と故障状態の2つである。残念ながらどちらも完全な実現技術が存在しておらず、次に示すような技術を組み合わせることにより安全性を高めることが可能である。

(1) 検証／評価

ソフトウェアの信頼性は、ソフトウェアの誤りに起因しており、検証／評価は、誤りを徹底的に除去することにより、信頼性を高める技術である。開発ライフサイクルの各フェーズにおいて、徹底的な検証と妥当性確認を行う方法である。設計段階でのレビュー、機能テスト、独立検証、評価などにより、欠陥を除去し、また信頼性を確認する。

評価の一環として、故障の木解析手法（SFTA: Software Fault Tree Analysis）やFMEA（Failure Mode and Effects Analysis）などが行われている。これは設計段階における要求インテグリテイ水準を求めるためではなく、作られたソフトウェアに対し、そのインテグリテイ水準を評価するためである。

開発技術だけで信頼性を実現するのは不可能であり、検証／評価技術は安全性検定において大きな割合を占めている。

(2) ソフトウェア構造設計技法

モジュールや機能の独立性を高めるソフトウェア構造設計技法が重要視されている。また独立性をプログラムのコーディング段階でも維持するため、使用する言語やプログラミング規格に対する制約を課している。

割り込み処理の禁止、リカーシブ処理の禁止なども、ソフトウェアの故障状態を複雑にしないために行う経験的な設計ノウハウである。

(3) ソフトウェア開発／管理プロセスの改善と維持

ソフトウェア開発や保守における一連の活動に対し、直接的な技術だけでなく、要員の資格や教育、使用する開発環境、検証技術、構成管理、役割分担と組織、管理の方法など様々なプロセスを体系的に捕らえ、ソフトウェアの信頼性や生産性を向上させるアプローチである。

(4) 設計ダイバーシティ

ソフトウェアにおける冗長構成に相当する技術で、リカバリー・ブロック、N-自己検査プログラミング、N-バージョン・プログラミングなどの技法が提案され一部で実用化が行われている。また従来装置と併設し、コンピュータ・システムの安全性実績を確認する方法も多く用いられている。

(5) 形式的方法

ソフトウェアの検証や評価を厳密に行おうとした時、基本となるソフトウェアの仕様が曖昧であることが本質的な問題である。形式的な言語や数学的な論理記述によって、ソフトウェアの仕様化を厳密に定義する。仕様の数学的な証明や、仕様の詳細化を行いながら、プログラムを設計する方法である。適用分野によりLOTOS、VDM、Zなどの方法論／記述言語が存在する。英国を中心とした欧州で研究と実用化が進んでいる。ソフトウェアの設計においても、また検証においても形式的方法は有望な技術である。

5. ソフトウェア安全性規格の課題

先に述べたように、コンピュータとソフトウェアの導入は、非常に早い勢いで進もうとしている。安全関連分野におけるソフトウェア安全性規格として次のような課題がある。

(1) 既に安全性規格を持っている分野

ソフトウェア安全性に関する共通的な取り組みとして、ここで示した国際規格のような共通の枠組みが求められている。国際規格として完成させる課題と、それを産業分野別の安全性規格に派生させる課題とがある。

(2) ソフトウェアにより安全性の問題が生じる分野

コンピュータとソフトウェアによって制御された機械系システムは、マイクロコンピュータ技術の飛躍的な発展により急速に進みつつある。従来の制約条件であった経済的な障壁は、既に逆転すらしている。コンピュータ技術による機械系の制御機能は、従来機能の置き換えから始まり、急速に機能の拡大が行われる傾向にある。

このような変化の中で、結果的にみてソフトウェアの誤りやインテグリテイ水準の低さに起因する問題が生じる可能性を持っている。原子力に対する過度の警戒と同じようなことが、コンピュータとソフトウェアに対して生じないよう、産業界としてソフトウェア安全性規格、あるいはそれに相当する活動が行われる必要がある。

(3) 安全性技術の課題

ソフトウェアの安全性に関するソフトウェア工学上の取り組みは、まだ不十分である。特にわが国では、軍事、航空宇宙、原子力と言った安全性での先進的な産業規模が小さく、かつ閉鎖的である。ソフトウェア安全性に関する認識や研究は非常に低い状況である。

この分野で特に目立つのが、英国など欧州における技術力の高さであり、わが国との差は１０年以上と思われる。ソフトウェア安全性技術はソフトウェア生産における基盤技術の一つであり、大きな課題である。

Ⅱ. ソフトウェア・セーフティとその規格化の現状

細　目　次

1．はじめに

2．概念と定義

3．規格化の目的

4．ソフトウェアの安全規格の現状

4．1　ISO/IEC規格案：ISO/IEC JTC1/SC7 N917

4．2　英国防省(ＭＯＤ)規格案：INT DefStan 00-55

4．3　その他の安全規格（案）

(1) ANSI/IEEE-ANS-7-4.3.2-1982

(2) 米国防省(ＤＯＤ)規格：MIL-STD-882B

(3) RTCAガイドライン：DO-178A

(4) IEC/SC65A/WG10規格案

(5) IEEEワーキング・グループ P1228規格案

5．おわりに

参考文献

Ⅱ. ソフトウェア・セーフティとその規格化の現状

1. はじめに

航空・鉄道・原子力・軍事分野での機械およびプロセスの制御には、実績のある従来の電気機械系に取って代わって（ディジタル・）コンピュータが使用されてきている。これれらの分野での制御・安全系の故障は即事故に結び付き、生命を危うくする[Leveson 1986, Neumann 1989, Leveson 1991, Wichmann 1992, Bhansali 1993]。航空機や鉄道での最近の事故ではコンピュータの故障、特にソフトウェアの誤りが原因ではないかと指摘されている*1。それ故に、コンピュータ／ソフトウェアを安心（信頼）して市民生活および公共の福祉の向上に利用するには、高いディペンダビリティ（信頼性、アベイラビリティ、保守性、セーフティ、セキュリティを包括した概念）が確保されていなければならない。また同時に、ソフトウェア開発技術、特に形式的方法の限界についても正しく認識・理解しておかなければならない。基本的には、安全を保つのは人間（の責任）であって、機械（の責任）ではない。技術も非常に重要ではあるが、最も重要なことは、他の品質以上に安全性に関して組織が強い関心を持ち、実施責任（Responsibility）と結果責任（Accountability）を負うことである[Leveson 1992]。

ソフトウェア単独では誰にも被害を与えず、本来的（inherent）には危険物ではない。即ち、ソフトウェアそれ自身では、ダイナマイトのように爆発もしないし、電気のように利用者を感電死させることもない*2。しかし、安全機能がソフ

---

*1：安全性が高度に高められた複雑なシステムでは、一般に事故は複合原因によってもたらされるので、どれか１つを主原因あるいは根本原因だと断定（特定）するのが困難になってきている。その結果、たいていの場合、主原因をヒューマン・エラーとして決着させている。航空機事故の大多数はパイロットの誤りが原因であるとほとんど普遍的に信じられているが、米空軍が飛行中に発生した681件の緊急事態について調べたところによると、装置や保守上の欠陥が原因である緊急事態の内、659件を乗務員がリカバリしており、パイロットの誤りは10％にしかすぎなかった[Leveson 1992]。

*2：それ故に、本稿の主題でもある“ソフトウェア・セーフティ”(Software Safety)という用語の定義は、従来の安全性から直観的に連想される定義とはかなり異なっている。第２章の「概念と定義」を参照されたし。

トウェアに存在する場合には、ソフトウェアはシステム安全性の性質を継承することになる。これらの安全機能には、制御、安全保護、監視、クリティカル情報の表示などが含まれる[Bhansali 1993]。

従来の安全工学では、主として、電気機械系での偶然的な故障（Random failure）にいかに対処して安全を保つかを主題としている。ソフトウェアの場合は偶然的な故障ではなく、系統的な故障（Systematic failure）（即ち、設計誤り）であり[Parnas et al. 1990, Leveson 1992]、従来の安全工学では十分には対処できない。それ故に、ソフトウェアを対象とする安全工学を新たに構築する必要がある。

主として安全性が問題となる、安全第一の分野へ適用されるソフトウェアは特にセーフティ・クリティカル・ソフトウェア（Safety-critical software）とか、安全関連ソフトウェア（Safety-related software）とよばれ、ソフトウェア工学での主要な一分野を形成しつつある。

以下では、このような問題意識に立脚して、①ソフトウェアのセーフティ（以下では、安全性ということにする）に関係する「重要な概念・用語の定義」、②規格化の目的について述べ、③最後に、現在、国際標準化機構などで精力的に検討・作成・制定が進められている「ソフトウェア安全性に関する規格」（ISO/IEC規格案、英国防省（ＭＯＤ）規格案、ANSI/IEEE-ANS規格、米国防省（ＤＯＤ）規格、ＲＴＣＡガイドライン、IEC/SC65A/WG10規格案、IEEEワーキング・グループ P1228規格案）の規定内容および動向について紹介する。

2. 概念と定義

本稿では、まず最初にソフトウェアの安全性を考え・検討する上で必要な重要概念・用語の定義を紹介する[MIL 1984, AFISC 1985, Leveson 1986, MOD 1989, EIA 1990, MOD 1991, Gonzalez Sanz 1991, ISO/IEC 1991]。なお、原子力分野での安全性に関する概念・用語はその分野特有の具体的な定義となっている。

概念を定義するに際しては、その概念が満たすべき諸性質（property）を項にして規定すべきであって、別の技術用語（単なる言い替え）や実現手段を用いて定義することは避ける必要がある[Leveson 1992]。実現手段をその定義の中に入れると、実現手段の有効性を総合的に比較評価せずに、その実現手段を利用すれ

ば、その性質が達成されたと仮定される傾向にあるからである。

ソフトウェアは安全性や信頼性の観点から見ると、従来のハードウェアと本質的に異なる点が多いので[Parnas et al. 1990]、ここで紹介する安全確保に関して技術的蓄積のある主としてハードウェア分野での定義を、ソフトウェアの本質・特性を反映していかに読み替えるかが課題となろう。安全性の維持・向上はソフトウェア単独では実現不可能で、ハードウェアと一体となったシステム的なアプローチを必要とし、ハードウェア技術者との協力は不可欠であり、ソフトウェア固有の安全性に関する用語の定義・使用は避ける必要がある。

以下では、重要概念・用語をアルファベット順に定義・紹介する。

＊事故（Accident）

(1) 死、損傷、環境破壊、または物質破壊の原因となる不測の出来事／事象または一連の出来事／事象[MOD 1989, MOD 1991]

(2) 結果として、死、損傷、健康・環境被害、または機器もしくは財産の損失をもたらす予定外の出来事／事象（event）または一連の出来事／事象[IEEE 1990]

＊事故条件（Accident condition）

(1) 放射性物質の放出が適切な設計で定められた受け入れ可能な運転限界範囲内に保たれている状態からの逸脱（deviation）[IAEA 1988b]

＊クリティカル属性（Critical attribute）

(1) その属性が制御不能になった場合に、システムの開発または機能に多大な影響を及ぼす属性[EWICS 1988]

＊ハザード（Hazard）

(1) 事故に至る可能性のある条件[MOD 1989, MOD 1991]

(2) ある特定の環境条件が揃えば、事故に至る可能性のある条件（即ち、状態）の集り[Leveson 1990]

＊検　査（Inspection）

(1) 製品やサービスの1つまたはいくつかの特性を測定、精査、テスト、ゲージ合わせ（gauge、寸法・量などを測定すること）をし、それらの結果と規

定された要求事項と比較し、適合しているかどうかを判定する活動[ISO 1984]

＊プログラマブル電子システム（PES: Programmable Electronic System）

(1) 制御、安全保護、または監視の目的で、センサ（sensor）および／またはアクチュエータ（actuator）が結合した1台またはそれ以上の中央処理装置（CPU: Central Processing Unit）に基づくシステム；用語ＰＥＳには、一体となった複数のシステム、マイクロプロセッサ、プログラマブル・コントローラ（PC: Programmable Controller）、プログラマブル・ロジック・コントローラ（PLC: Programmable Logic Controller）、そして、その他のコンピュータに基づく機器が含まれる[IEC 1989]。

＊品質保証（QA: Quality Assurance）

(1) 製品またはサービスが所与の品質要件を満たしていることの妥当な信頼感を与えるために必要な計画的および体系的活動のすべて[ISO 1984]

＊品質監査（Quality Audit）

(1) 品質活動およびそれに関連する結果が前もって計画した事項と合致しているかどうか、そして、これらの計画した事項が効果的に実施され、かつ、目的達成のために適切なものであるかどうかを決定する体系的かつ独立的な精査[ISO 1984]

＊信頼性（Reliability）

(1) アイテムが与えられた条件で規定の期間中、要求された機能を果たすことのできる性質[JIS信頼性用語]

(2) ［ソフトウェア信頼性］ソフトウェアが、規定された条件下で規定の期間中、システムの故障（Failure）を起こさない確率。この確率は、入力およびシステムの利用とからなる関数であると同時にソフトウェア内に存在する欠陥（フォールト；Fault）からなる関数でもある。システムに対する入力が欠陥に遭遇するか否かによって、その存在の有無が決まる[ANSI 1983]。

＊信頼性プログラム（Reliability program）

(1) 信頼性目標値の設定およびそれを実現するための技術的、管理的な手順の計画体系[JIS信頼性用語]

＊リスク（Risk）

(1) 事故の厳しさとその尤度の尺度[MOD 1989, MOD 1991]

(2) システム・ハザードが事故の原因となる尤度（likelihood）と、その事故の厳しさ（severity）とを組み合わせた尺度[IEEE 1990]

(3) ①ハザードが発生する尤度、②そのハザードが事故に至る尤度、そして、③そのような事故に伴う最悪の可能性のある起こり得る損失とからなる関数[Leveson 1990]

(4) 規定されたハザードな事象の頻度、即ち、確率と、その結末との組み合せ。リスクの概念は、２つの要素、即ち、ハザードが起こる頻度／確率と、そのハザードな事象の結末とから必ずなっている[ISO/IEC 1991]。

＊安全性（Safety）

(1) 人間の死傷または資財の損失もしくは損傷を与えるような状態がないこと。信頼性では任務遂行のための機能上の故障を対象にするが、安全性では人間・資財に損失・損傷を与える危険な状態を対象とする[JIS信頼性用語]。

(2) 機器の損害もしくは破壊、または人間の損傷もしくは死の原因となる可能性のある諸条件が発生しないこと[MIL 1984]。

(3) はなはだしい放射性ハザードから現場要員、一般市民、そして環境を保護するための適切な運転条件の達成、事故の予防、または事故の結末の軽減[IAEA 1988a]

(4) その故障によって発生する危険から生命および手足、環境そして財産を保護する、または危険そのものを除去すること[EWICS 1988]。

(5) 人命に危害を与える受け入れられないリスクがないこと[EOQC 1989]。

(6) 危害または損害のリスクが受け入れ可能な水準内に保たれている状態[ISO 1989]

(7) システムが規定の条件下で、人命、経済、または環境が危険にさらされる状態に至ることがない期待[ISO/IEC 1991]

(8) 安全性はハザードで定義され、ハザードを除去あるいは制御する方法を見つけることによって安全性の問題に取り組む。これには２つのアプローチが可能である。一つはハザードの発生を除去するか、あるいは最小化するアプローチである。もう一つはハザードが発生した際の損傷あるいは損害を予防するためにハザードを制御するアプローチである[Leveson 1992]。

＊セーフティ・クリティカル機構（SCF: Safety Critical Features）

(1) システムに関連する受け入れられないハザードを除去したり、あるいは、これらのハザードに起因するリスクを軽減して、これらのハザードを受け

入れられる水準にまでにするためにシステムが保持しなければならない機構[MOD 1989]

(2) 防衛装置から受け入れられない安全性リスクを除去し、防衛装置からその他のリスクを軽減して、これらのリスクを許容水準にまでにするために必要な機構[MOD 1991]

＊セーフティ・クリティカル・ソフトウェア（SCS: Safety Critical Software）

(1) 安全性インテグリティとして最高水準が要求される機能あるは構成要素を実現するのに使用されるファームウェアを含むソフトウェア[MOD 1991]

(2) システム中で使用されているソフトウェアで、受け入れられないリスクの原因となる可能性のあるソフトウェア；ＳＣＳには、①そのソフトウェアの運転または運転故障がハザードな状態に至る可能性のあるソフトウェア、②ハザードな状態から回復することを目的とするソフトウェア、そして、③事故の厳しさを軽減することを目的とするソフトウェアが含まれる[IEEE 1990]。

＊セーフティ・クリティカル・システム（SCS's: Safety Critical Systems）

(1) 人命または財産を危険にさらす可能性のある環境下で利用されるシステム[EWICS 1988]

＊安全性インテグリティ（SI: Safety Integrity）

(1) プログラマブル電子システム（PES）が規定されたすべての条件下で、規定された期間中、安全機能を達成する尤度[ISO/IEC 1991]

(2) コンピュータだけでなく、システム全体に依存して、制御システムが安全に作動する（これにはフォールトが発生した時に安全に停止することが含まれる）能力[EIA 1990]

(3) セーフティ・クリティカル・システム、機能、あるいは構成要素が規定された利用方法の範囲内で規定されたすべての条件下で、要求されたセーフティ・クリティカル機構を達成する尤度[MOD 1991]

＊安全関連ソフトウェア（SRS: Safety-Related Software）

(0) セーフティ・クリティカル・ソフトウェア（SCS）と同義語または広義語

(1) システムが人命、経済、または環境を危険にさらさせないことを確かなものにするソフトウェア[ISO/IEC 1991]

＊安全関連システム（SRS's: Safety-Related Systems）

(0) セーフティ・クリティカル・システム（SCS's）よりも広義の用語

(1) 安全関連システムは、他のものの力を借りず、独力で、主として、①予期されたハザードな事象が発生するのを防ぎ、②もし仮に、（予期されない）ハザードな事象が発生した際には、その影響を軽減する目的で使用されるシステムである。そして、安全関連システムには、運転員など安全関連の要員もおそらく含まれるであろう[IEC 1989]。

(2) 原子力分野では、①は制御系、そして②は安全（保護）系に分類している。

＊安全系（Safety system）

(1) ①原子炉の安全な運転停止を保証する、②炉心からの残留熱の除去を保証する、または③想定された運転状況の発生、即ち、事故条件発生の結末に限界を設けるという条件付きで、安全が最優先であるシステム[IAEA 1988 b]

＊ソフトウェア安全性（Software safety）

(1) ソフトウェアがシステムの文脈（system context）下で実行された時に受け入れることのできないレベルのリスクを発生させることがないことを確かなものにすること。システム安全技術者は安全性問題をシステムとして解決することを強調するためにソフトウェア安全性よりもソフトウェア・システム安全性という用語を好んで使用している[Leveson 1991]。

(2) ソフトウェアの設計の際に、システムの安全性を向上させるために積極的な処置を講じ、そして、システムの安全性を低下させる可能性のある誤りに対して、その誤りによるリスクを許容可能な水準になるまで除去あるいは制御していることを保証し、検証する目的でソフトウェアに対してシステム安全性工学の技法を適用すること[AFISC 1985]。

＊ソフトウェア安全性インテグリティ（SSI: Software Safety Integrity）

(1) プログラマブル電子システム（PES）中に存在するソフトウェアがすべての規定された条件下で、規定された期間中、安全機能を達成する尤度[ISO/IEC 1991]

＊ソフトウェア安全性プログラム（Software safety program）

(1) システムの安全性を保証する系統的なアプローチ[IEEE 1990]

＊ソフトウェア・システム安全性（SSS: Software Systems Safety）

(1) 「ソフトウェア・システムの設計・構築・使用・保守」および「セーフティ・クリティカルなハードウェア・システムとの統合」においてのシステムの安全性の最適化である[MIL 1984]。即ち、ＳＳＳの定義の意図する所

は、ソフトウェアがトータル・システムの環境下で、受け入れることのできないレベルのリスクを発生、またはハザード状態の原因になることなしに、機能することを保証することである。

＊システム安全性（System safety）

(1) 科学的、管理的、工学的な諸原則を適用して、システムのライフサイクルのすべてのフェーズにおいて、軍事作戦有効性（Operational effectiveness）、時間、そしてコストに関する諸制約条件を満足させて、システムの安全性を最適にすること[MIL 1984]。

＊システム安全性プログラム（System safety program）

(1) システムのライフサイクルのすべてのフェーズにおいて、タイムリーでコスト有効度の高いやり方でシステム安全性要件を満足させ、軍事作戦有効性を向上させるために必要なシステム安全性面での管理的および工学的な作業および活動のすべて[MIL 1984]

3. 規格化の目的

規格化は次に示すようにたくさんの目的にかなう[Wallace et al. 1992]。

①対照基準（Reference point）または評価基準（Criteria）
②製品要件
③高品質で経済的な製品
④技術水準の目標・改善

①は複数のシステムを比較する際の物差し・尺度として利用される。規格を各国がバラバラに作成し、保持することは国際的な通商活動の障害になる恐れがある。そのため国際規格は経済面での公平な国際競争にとってとても重要になってきている。近い将来（たぶん1993年）、ＥＣ（European Community）諸国に製品を売る場合、最低限の品質規格を満たすことが必要となる。

②は製品の要件を規定する一つの手段となる。発注者は最小の労力で要件を規定できることを望んでいる。特定のアプリケーション用に受け入れられている規格を指定し、それに適合することを要求することにより簡素化する。

③では、規格が共通の要件を提供することにより、より高品質の製品をより経済的に製造することを可能とする。

④は工学慣行（固有技術）の技術水準を改善する一つの手段である。従来のような技術の後追いの規格化ではなく、技術開発の目標を設定した規格である。ソフトウェアのセキュリティと安全性に関する規格はこの範疇に属する規格である。

4．ソフトウェアの安全規格の現状

安全性が問題となるアプリケーション領域では、システム要件（特に安全性要件）の実現にソフトウェアを適用する場合、開発者はそのソフトウェアが安全である（ハザードの原因とはならない）根拠（Rationale）や証拠（Evidence or Fact）を提示して、購入者や第三者（市民、規制当局など）に妥当な信頼感を与えなければならない。そのためには、それに準拠してソフトウェアを開発すれば合理的な範囲内で安全性が保証される安全規格・法規が必要である。現在の技術水準では、ソフトウェアの安全性・信頼性を完全に検査・保証する手段がないので、安全性要件規格（検査規格、製品規格）だけでなく、工程で安全性品質を作り込む開発のやり方（ソフトウェア・プロセス）（品質管理規格、プロセス規格）をも含めた広範なソフトウェア安全性・開発規格（ガイドライン）が要請される。

原子力や航空機の事故影響は地球規模（グローバル）であるので、安全性に関する規格は、各国がばらばらに制定し、保持することは、高度な安全性の維持、円滑な運用・適用、通商活動などを妨げるので、国際的な調和のとれた統一規格が望ましい（例えば、品質保証／品質管理に関してはISO 9000シリーズ(通称)が制定された。）。ＥＣでは、1992年のＥＣ統合、1993年の市場開放を目指して、安全関連制御システムの規格化を重要作業項目として域内の各国の国家規格を整合ヨーロッパ規格（Harmonised European Standards）に一致させる作業を精力的に進めている[Bell & Smith 1990]。

なお、本稿では、ソフトウェアの開発面に焦点を当てているので、製品が利用者に渡った後での、運用・保守での安全性を確保するための規格・規則も非常に重要ではあるが（運用と保守を対象とする規格はまだない）、ここではその重要性を指摘するのにとどめ、割愛する。

以下では、現在、国際・国家標準化機構、業界、学会、各国の国防省などで精力的に検討・作成・制定が進められているソフトウェア安全性に関する次に示す安全規格について、規格化の内容・動向を簡単に紹介する[Dale 1989, Gonzalez Sanz 1991, Wright & Zawilski 1991, Wallace et al. 1992]。

①ISO/IEC規格案：ISO/IEC JTC1/SC7 N917
②英国防省(ＭＯＤ)規格案：INT DefStan 00-55
③ANSI/IEEE-ANS-7-4.3.2-1982
④米国防省(ＤＯＤ)規格：MIL-STD-882B
⑤ＲＴＣＡガイドライン：DO-178A
⑥IEC/SC65A/WG10規格案
⑦IEEEワーキング・グループ P1228規格案

4.1 ISO/IEC規格案：ISO/IEC JTC1/SC7 N917

"Software for Computers in the Application of Industrial Safety-Related Systems"（1991年11月発行）[ISO/IEC 1991]

本規格案は、元々はIEC/TC65/SC65A/WG9で作成された規格案で、IEC/TC65/SC65A/WG10で検討・作成されているシステム全体の安全性を対象とする規格（後述）[IEC 1989]と一対で利用され、産業での安全関連アプリケーションで利用されるソフトウェアに関して、設計、開発、構築、そして評価について扱っている（図1参照）。（英国では、同名の規格をBS89/33006DCと付番している。）（TC65は産業用プロセス計測制御を担当する技術委員会で、SC65Aはシステムの側面を担当する部会である。）第1章から第8章は、ソフトウェア安全性インテグリティ（SSI）とソフトウェア・ライフサイクルに向けられている。第9章から第17章では、安全関連ソフトウェア（SRS）の設計および評価の諸技法・手段を規定している。最後の第18章では、本規格が特定の産業を想定しない汎用規格（基本規格）であるので、特定の産業に特化したソフトウェア安全規格を作成する際の手引を与えている。付録では、各ソフトウェア安全性インテグリティ水準（IL: Integrity Level）を達成するのに使用すべき技法および手段を規定し、各技法を解説している。

1．序　文
2．適用範囲
3．引用規格
4．定　　義
5．目標および適合性
6．ソフトウェア安全性インテグリティ水準
7．要員とその責務
8．ライフサイクル問題と文書
9．ソフトウェア要求仕様
10．ソフトウェア構造
11．設計および開発
12．検　証
13．ソフトウェア／ハードウェアの統合化
14．ソフトウェア妥当性検査
15．評　価
16．品質保証
17．ソフトウェア保守
18．アプリケーション分野固有の規格を派生するための手引

図１　ISO/IEC N917規格案の目次構成

（ＩＬの定義を文字通りに解釈すると、安全性と信頼性とを同じと考えていると思われてもしかたがない[Leveson 1992]。即ち、I.Lは通常では、まさに信頼性水準の偽名にしかすぎず、安全性を安全保護装置の信頼性と定義している(この定義は原子力産業では行き渡った定義である。)。本規格のこの定義からすると、事故の原因となるハザードを根本的に除去するという安全性の第一原則(本質的安全(Inherent safety))が最重視されず、無視されることになってしまう。）

ＩＬの基本概念を鉄道分野に適用すれば、ＡＴＣ（Automatic Train Control；自動列車制御装置）とＡＴＳ（Automatic Train Stop；自動列車停止装置）がＩＬ４（最高水準）、ＣＴＣ（Centralized Traffic Control；列車集中制御装置）がＩＬ３、ＣＯＭＴＲＡＣ（Computer Aided Traffic Control System；列車運行管理システム）がＩＬ２、そして、各駅の旅客サービス・誘導システムがＩＬ１

（最低水準）に該当すると思われる。ＩＬ０はＭＩＳ（Management Information System；経営情報システム）に代表されるビジネス・ソフトウェアが該当し、非安全関連ソフトウェアと称される水準で、ソフトウェアの品質管理／品質保証の国際規格であるISO 9000-3[ISO 1990]あるいは同等規格の履行を勧告している。ＳＲＳの開発では、ISO 9000-3の適用は最低限と見なしている。即ち、セーフティ・クリティカル分野では、ISO 9000-3は最低基準である。

実際に達成されるＩＬは、開発要員の能力以外に、採用されるソフトウェア開発法、ソフトウェア構造、品質保証、プロジェクト管理などの要因によって決定付けられると考えられるが、それらを厳格に実践することによって、（現在の技術水準では、）どの水準のＩＬが達成されたかをテストで確認したり、定量的に測定することはできない[Bennett 1991]。

本規格案は、完成したソフトウェアを評価して品質保証をするのではなく、要求されたＩＬを実現するプロセス、特に使用すべき技法・手段を特定し、それを使用することにより自然とそのＩＬが達成されるという考え（前提）で作成されている（我々は、完全なソフトウェアであることを約束（保証）することはできないが、その有効性が立証された最善で利用可能な管理面(共通)および工学面(固有)の慣行（practice）を実行することを約束することはできる。）。それ故に、本規格では、５０ページ以上に渡って現在利用可能で、有用と考えられる約７０個の技法・手段（図２参照）[Finnie & Johnston 1990]を開発サイクルに対応させ、要求されたＩＬを実現するために使用すべき技法・手段とをマトリックス形式で規定している。なお、形式的方法に関しては、いくつかある有効な技法の一つと位置付けており[Gruman 1989]、むしろ、本規格案では、（システム・）インテグリティ水準、アプリケーションの種類、および産業分野を熟考して、技法・手段のセットの中から適切に選択した複数個の技法・手段からなるパッケージ（package）を使用することを推奨している[Bennett 1989]。

特に欧州勢は、形式的方法は安全性を確かなものとする問題に対する部分的な解決策として提案しているが、この方法の背後に存在する諸仮説が妥当である証拠がほとんどないのが現状である[De Millo et al. 1979, Leveson 1992]。規格に取り入れるためにも、ＮＶＰ（N-Version Programming）と同様に、その仮説や仮定の妥当性を検定するために様々な角度からの実験が必要となろう。ソフトウェア工学の技法やツールに関しては、F.P. BrooksやD.L. Parnasらが指摘したように銀の弾丸（Silver Bullet）はまだ発見されていないし、現時点では存在しない[Brooks 1987]。

・ハザード解析(CCD, ETA, FTA, FMECA, HAZOP)
・共通原因故障解析(CCFA)
・形式的方法(CCS, CSP, HOL, LOTOS, OBJ, 時間論理, VDM, Z)
・準形式的方法(ロジック／機能ブロック・ダイアグラム、シーケンス・ダイアグラアグラム、有限状態マシン／状態遷移図、時間ペトリ・ネット、決定表、真理値表)
・構造化方法論(JSD, MASCOT, SADT, SSADM, Yourdon)
・プロトタイピング／アニメーション(仕様の妥当性検査)
・防衛化プログラミング
・故障アサーション化プログラミング
・安全バッグ技法
・多様化プログラミング／N-バージョン・プログラミング(NVP)
・リカバリ・ブロック(RB)
・記録された実行済みケース
・モジューラ・アプローチ(情報隠蔽化／カプセル化)
・強い型付けをもつプログラミング言語
・形式的証明
・確率論的(統計的)テスト
・静的解析(チェック・リスト、制御フロー分析、データ・フロー分析、誤り推定法、Faganインスペクション、スニーク回路解析、記号実行、ウォークスルー／設計レビュー)
・動的解析、動的テスト
・機能テスト、ブラック・ボックス・テスト
・性能テスト
・インタフェース・テスト
・テスト・ケースの作成法(限界値分析法、同値分割法、誤り推定法、誤り蒔く法、内部構造に基づく法)
・マルコフ・モデル
・信頼性ブロック図
・モンテカルロ・シミュレーション
・プログラム構造図
・ソフトウェア構成管理

図2　使用を推奨している技法・手段(一部)

4.2 英国防省（MOD: UK Ministry of Defence）規格案: INT DefStan 00-55

"Requirements for the Procurement of Safety Critical Software in Defence Equipment"（1991年4月発行）[MOD 1991]

本国防規格案INT DefStan 00-55では、国防装置で利用されるセーフティ・クリティカル・ソフトウェア（SCS）の調達および開発での諸手順および技術的要件を扱っている（図3参照）[Pilsworth 1989, Brown 1990, Wichmann 1992]。本規格案での要素は、①安全性管理の手順（計画および制御）、②（手法およびツールを含んだ）ソフトウェア工学の実施基準、そして、③プロジェクト・ライフサイクルの各ステージである。それらに関し、本規格案では、独立安全性監査担当者が、安全性計画が正しく実施されているかを独立に監査し、本規格で規定した要件が満たされていることを独立に精査することを要求している。要求されているソフトウェア工学固有の慣行としては、ハザード解析、安全性インテグリティ解析（安全性リスク評価）、仕様化技法、設計技法、レビュー、静的コード解析、そしてV&Vテスト（Verification and Validation Test; 検証および妥当性検査テスト）である。また、本規格案では、SCSは形式的仕様・設計記述技法を使用して規定されることを要求し、そして、静的解析・検証（Analysability）を容易にするために、同時併行処理、割り込み、浮動小数点演算、再帰、動的メモリ割当てなどの使用の慣行は受け入れられない慣行として特定している。

安全性インテグリティの水準に関しては区分けしていない。即ち、ここでのソフトウェアは、セーフティ・クリティカルか、非セーフティ・クリティカルのいずれかである[Bhansali 1993]。

安全性を向上させる常套手段であるハザード解析などの安全性解析については、本規格と一対の規格であるINT DefStan 00-56で必須文書類と共に詳細に規定されている[MOD 1989, Pilsworth 1989]。

本規格案の最も顕著な特徴は、形式的方法（例えば、形式的記述言語Z, VDM, OBJ, FOREST）および形式的検証（例えば、静的解析ツール: SPADE, MALPAS）を課している点である[Gruman 1989, Bloomfield et al. 1991, Hughes & Cooling 1991]。（形式的方法は、コンピュータ・セキュリティおよびデータ通信の分野では受け入れられた慣行となっている[Wallace et al. 1992]。）形式的検証では、仕様・設計の作成プロセスで1ステップ毎に証明すべき主張（立証責任

まえがき
第1部　総　則
0．序　文
1．適用範囲
2．警　告
3．関連文書
4．定　義
5．競争、利用、改造および支援を可能とする要求事項
第2部　安全性管理
6．安全性に対する責任
7．ＭＯＤ安全性保証担当局
8．ハザード解析および安全性リスク評価
9．入　札
10．支　援
11．品質保証
12．リスク分析
13．要員の質
14．設計チーム
15．Ｖ＆Ｖチーム
16．独立安全性監査担当者
17．下請負契約協定
18．安全性計画
19．安全性レビュー
20．設計実施基準
21．安全性記録簿
22．文　書
23．納入物件に対する要求事項
24．構成管理
25．検定合格および役務への受け入れ
26．製　造
27．供　用
28．廃棄処分
第3部　ソフトウェア工学に関する実施基準
29．仕様定義
30．設　計
31．コーディング
32．形式的論証
33．動的テスト
34．既存ソフトウェアの利用
35．妥当性検査
36．ツールおよび支援ソフトウェア

図3　MOD INT DefStan 00-55規格案の目次構成

(Proof Obligation)という）（関数定義S、前提条件P、帰結条件Q；例えば、ホア論理での表現形式：P{S}Q）（検証条件ともいう）が構成（作成）され、それに対して形式的証明あるいは厳密な論法（Rigorous Argument)を利用して形式的論証（Formal Argument）を構成（作成）し、関数定義の正当性を証明する（立証責任を果たすという）。特に最上位の形式的仕様書が非形式的なソフトウェア要求仕様書に合致しているかを数学的に証明することは一般的に不可能であるので、形式的仕様書が妥当であるかをアニメーション技術を使って確認することを要求している[Hughes & Cooling 1991]。

なお、完全に強制的な本規格の導入目標は1995年に設定されている[Brown 1990]。

4.3 その他の安全規格（案）

(1) ANSI/IEEE-ANS (American National Standards Institute / The Institute of Electrical and Electronics Engineers - American Nuclear Society) 規格

ANSI/IEEE-ANS-7-4.3.2-1982: "Application Criteria for Programmable Digital Computer Systems in Safety Systems of Nuclear Power Generating Stations" (1982年発行)[ANS 1982]

本規格は、8年間以上利用され、何回か改定されている。ANSI/IEEE-ANS-7-4.3.2は原子力産業固有の規格で、ソフトウェア考慮事項とハードウェア考慮事項についてほぼ等分の割合で具体的な安全性要件を列挙している。規格自体は5ページにすぎないが、ソフトウェア安全性要件チェック・リストとV&Vの概要は非常に有用である。

原子力発電分野では、この他に安全系のソフトウェアを対象とする国際規格として1986年に発行されたIEC 880 "Software for Computers in the Safety Systems of Nuclear Power Stations"がある[IEC 1986]。また、IEC 880と一対の規格であるシステムとハードウェアの側面を扱った1987年に発行されたIEC 987 "Programmed Digital Computer Important to Safety for Nuclear Power Stations"もある。

(2) 米国防省（DOD: United States Department of Defense）規格
MIL-STD-882B: “System Safety Program Requirements”（1984年5月3日発行、1987年6月1日改定）[MIL 1984]

システム安全性プログラムであるMIL-STD-882は非常に古く、1969年に発行された。現時点までででは、本規格882Bが印刷されているが、882Cが882Bにとって替わるかもしれない。そして、882Dは検討中である。882Cはソフトウェア・システム安全性に関する作業用の規格案である。882Bの付録には、ソフトウェア・ハザード解析の要約（タスク・セクション300）と、システムのライフサイクルの各工程と安全性との関係があり、有用である。882Bには、安全性に関する主要な用語の定義、安全関連要員に対する要件、そして、安全性プログラムの機能の列挙がある。安全性に関するプログラム・タスク（Program task）は明確に規定されている。規格案882Cには、ソフトウェア・システム安全性に特に関連する一連のタスクが含まれている。

なお、本規格を補完する資料として、米空軍発行のハンドブック“AFISC SSH 1-1: System Safety Handbook Software System Safety”がある[AFISC 1985]。

(3) RTCA（Radio Technical Commission for Aeronautics）ガイドライン
DO-178A: “Software Considerations in Airborne Systems and Equipment Certification”（1985年3月発行）[RTCA 1985]

航空分野が最も積極的に安全関連ソフトウェア（SRS）を利用している分野である[Potocki De Montalk 1991]。航空機で利用するＳＲＳの製造および検証に関するガイドラインは、RTCA（米国政府および産業に対する諮問団体）とEUROCAE（European Organisation for Civil Aviation Electronics; 同類の欧州の諮問団体）-WG12とで作成された[Helps 1986, Pyle 1991]。（EUROCAEでは、同名のガイドラインをED-12Aと付番している。）本ガイドラインは、基本的には民間航空機を対象としているが、米国、英国、そして欧州の国防規格に影響を与えている。英国防省では、航空電子システム用の本ガイドラインを空輸システム用にテイラリングして、DefStan 00-31 “Development of Safety Critical Software for Airborne Systems”を作成している。そして、本文書はガイドラインではあるが、ＦＡＡ（US Federal Aviation Administration; 米連邦航空局）が認定し、ＦＡＡの認定を受ける際の事実上の基準として用いられている（ボーイング767SP-200用のソフトウェアの認定で使用された。）。

DO-178Aとして知られているRTCAガイドラインは、ソフトウェア開発の基本原則と、ソフトウェア検定（Software certification）[Dale 1989]のアプローチについて規定しているが、これらの基本原則や検定アプローチをどのように適用するかの詳細は述べていない。DO-178Bは開発中である[Potocki De Montalk 1991]。

本文書は、機能の致命度（Functional criticality）の分類法、機能の致命度に応じたソフトウェアの保証水準の選定、ソフトウェア開発の技法・手法、検定申請者の利用に適するV&V、システム／ソフトウェア要件の範囲、検定に関連する開発計画およびテスト、構成管理および品質保証のやり方、そして検定を正当化する勧告された文書を含めて、ソフトウェア工学での主要なタスクを実施する際のガイドラインを規定している。

機能の致命度としては、次に示すように3水準に分類されている。

①クリティカル（Critical、致命的）

なんらかの不具合や設計誤りが起きた場合、航空機の安全な飛行と着陸が妨げられるような機能をクリティカルと分類する。例えば、エンジンの状況を表示するMFD（Multi-Function Display）はこの水準に分類される。

②エッセンシャル（Essential、重大）

なんらかの不具合や設計誤りが起きた場合、航空機の運航を可能にする航空機の機能や搭乗員能力を減少させる機能をエッセンシャルと分類する。例えば、通信系はこの水準に分類される。

③ノン-エッセンシャル（Non-essential、軽微）

なんらかの不具合や設計誤りが起きた場合でも、航空機の機能や搭乗員能力を極端に損なわないような機能をノン-エッセンシャルと分類する。例えば、エアコンはこの水準に分類される。

(4) IEC/SC65A/WG10規格案：“Functional Safety of Programmable Electronic Systems, Generic Aspects”（1989年9月発行）[IEC 1989]

本規格案は、安全性が問題となる様々なアプリケーション領域でのシステム（PES）の設計および評価に利用可能な汎用規格（水平規格）である[Bell & Smi

th 1990]。（英国では、同名の規格をBS89/33005DCと付番している。）本規格案は、実施すべき技術に関連させて、安全関連システム（SRS's）で利用可能なセーフティ・ライフサイクル・モデル（①ハザード解析およびリスク評価、②安全性要求仕様、③SRS'sの特定(指名)、④設計および構築、⑤安全性の妥当性検査、⑥運転および保守、⑦システムの改造、⑨退役、⑩強化目的の交換(Retro-fit)）を特定している。システムの安全性を向上させる手段として、偶然的なハードウェア故障と系統的な故障の両方に対して十分な予防措置を提供することをもくろむ3つの“システム安全原則”（System Safety Element）（システム特性）（①システム構成(Configuration)(例えば、チャネルの多重化)、②安全関連ハードウェアの信頼度(例えば、安全防護装置(Safeguard)の信頼度)、③系統的インテグリティ(例えば、要員や開発作業・活動の質の向上)）を提案している。そして、本規格案では、安全性インテグリティ水準の概念を導入している。各々の特定のアプリケーションで受け入れられるリスクの水準は、適切な安全性インテグリティ水準を選択するのに使用される。

(5) IEEEワーキング・グループ P1228：“Standard for Software Safety Plans”（1990年11月発行）[IEEE 1990]

本規格は開発中であり、まだ規格案の作成段階までには至っていない[Wright & Zawilski 1991]。P1228-WGは、ソフトウェア安全性計画に関する規格を作成することを、活動目標として定めている。本WGでは、伝統的な品質保証慣行とは異なった、もっと特殊化した機構を利用してのセーフティ・クリティカル・ソフトウェア（SCS）の開発の必要性を認識している。P1228文書では、ソフトウェア安全性計画での強制的な要素を規定し、概念化段階から退役（retirement）までの全ライフサイクルを扱っている。管理面と工学慣行面の両方について検討されている。また、P1228文書には、ツールと技法を述べた付録が含まれている。しかし、構築手法については、特に勧告していない。なお、P1228文書では、他のIEEEソフトウェア工学規格が使用されることを前提にしている[Wallace et al. 1992]。

P1228文書では、形式的方法に関しては規定していないし、その使用を強制していない[Bowen & Stavridou 1992]。

5. おわりに

以上、ソフトウェアの安全性を向上させるために重要と考えられる、「概念・用語の定義」、「規格化の目的」、そして「ソフトウェアの安全規格」について概説したが、これらは現時点ではまだ十分に確立した概念・技術・規格とは言えないものであり、真に実用化にはまだ多くの年月を必要としよう。

ソフトウェア安全規格はソフトウェア・セキュリティ規格[DOD 1985]（通称オレンジ・ブック）と同様に現在のソフトウェア技術水準では達成できない要求を課している[Gruman 1989]。即ち、プロセス・インプルーブメント（工程改善）ではなく、メソドロジィ・イノベーション（方法論革新）を求めている。メソドロジィ・イノベーションに追従できないソフトウェア会社および国は国際的な技術競争に敗れ、淘汰されるであろう。

参考文献

[AFISC 1985] AFISC SSH 1-1: System Safety Handbook Software System Safety, Department of the Air Force, Headquarters Air Force Inspection and Safety Center, 5 September 1985.

[ANS 1982] ANSI/IEEE-ANS-7-4.3.2-1982: Application Criteria for Programmable Digital Computer Systems in Safety Systems of Nuclear Power Generating Stations, American Nuclear Society, 1982.

[ANSI 1983] ANSI/IEEE Std 729-1983: IEEE Standard Glossary of Software Engineering Terminology, August 9, 1983.

[Bell & Smith 1990] Bell, R. and Smith, S., "An Overview of IEC Draft Standards: Functional Safety of Programmable Electronic Systems," COMPASS '90, June 1990, pp.151-163.

[Bennett 1989] Bennett, P.A., "The March Towards Standards in Safety Related Systems," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.106-110.

[Bennett 1991] Bennett, P.A., "Forwards to Safety Standards," Software Engineering Journal, Vol.6, No.2, pp.37-40, (Mar. 1991).

[Bhansali 1993] Bhansali, P.V., "Survey of Software Safety Standards Shows Diversity," IEEE Computer, Vol.26, No.1, pp.88-89, (Jan. 1993).

[Bloomfield et al. 1991] Bloomfield, R.E., Froome, P.K.D., and Monahan, B.Q., "Formal Methods in the Production and Assessment of Safety Critical Software," Reliability Engineering and System Safety, Vol.32, Nos.1/2, pp.51-66, (1991).

[Bowen & Stavridou 1992] Bowen, J.P. and Stavridou, V., "Formal Methods and Software Safety," IFAC SAFECOMP '92, Oct. 1992, pp.93-98.

[Brooks 1987] Brooks, F.P., Jr., "No Silver Bullet - Essence and Accidents of Software Engineering," IEEE Computer, Vol.20, No.4, pp.10-19, (Apr. 1987).

[Brown 1990] Brown, M.J.D., "Rationale for the Development of the UK Defence Standards for Safety-Critical Computer Software," COMPASS '90, June 1990, pp.144-150.

[Dale 1989] Dale, C., "Software Safety Certification in Potentially Hazardous Industries," in Software Certification, ed. by de Neumann, B., London:Elsevier Applied Science, Reading, 1989, pp.100-112.

[De Millo et al. 1979] De Millo, R.A., Lipton, R.J., and Perlis, A.J., "Social Processes and Proofs of Theorems and Programs," Communications of the ACM, Vol.22, No.5, pp.271-280, (May 1979).

[DOD 1985] DOD 5200.28-STD: Department of Defense Trusted Computer System Evaluation Criteria, Department of Defense, Washington, D.C., Dec. 1985.

[EIA 1990] SEB6-A(EIA): System Safety Engineering in Software Development, Electronic Industries Association, EIA Safety Engineering Bulletin No.6-A, Apr. 1990.

[EOQC 1989] EOQC-Glossary Committee, Glossary of Terms Used in the Management of Quality, 6th Ed., June 1989, Bern, Switzerland.

[EWICS 1988] EWICS-TC7, Guidelines on Safety Related Measures to be Used in Software Quality Assurance, EWICS-TC7, Position Paper 8, 1988.

[Finnie & Johnston 1990] Finnie, B.W. and Johnston, I.H.A., "Practical Experience in the Assessment of Existing Safety Critical Computer Based Systems," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.95-98.

[Gonzalez Sanz 1991] Gonzalez Sanz, G., "Standardization for Safety Software: Current Status and Perspectives," SESAW '91, San Diego, May 1991, pp.84-105.

[Gruman 1989] Gruman, G., "Software Safety Focus of New British Standard," IEEE Software, Vol.6, No.3, pp.95-96, (May 1989).

[Helps 1986] Helps, K.A., "Some Verification Tools and Methods for Airborne Safety-Critical Software," Software Engineering Journal, Vol.1, No.6, pp.248-253, (Nov. 1986).

[Hughes & Cooling 1991] Hughes, T.S. and Cooling, J.E., "Real-Time Systems - Animation Prototyping of Formal Specifications," The Third IEE/BCS International Conference on Software Engineering for Real Time Systems, Cirencester, Sept. 1991, pp.51-56.

[IAEA 1988a] IAEA, Code on the Safety of Nuclear Power Plants: Quality Assurance, International Atomic Energy Agency, Safety Standards, Safety Series, Vienna, 1988.

[IAEA 1988b] IAEA, Code on the Safety of Nuclear Power Plants: Design, International Atomic Energy Agency, Safety Standards, Safety Series, Vienna, 1988.

[IEC 1986] IEC 880: Software for Computers in the Safety Systems of Nuclear Power Stations, International Electrotechnical Commission, 1986.

[IEC 1989] IEC Draft Standard: Functional Safety of Programmable Electronic Systems: Generic Aspects; Part 1:General Requirements, IEC Reference:65A(Secretariat)96, Sept. 1989, pp.104.

[IEEE 1990] IEEE-P1228, Software Safety Plans, IEEE-P1228, Draft Standard, Nov. 1990.

[ISO 1984] ISO 8402: Quality - Vocabulary, First Edition, June 15, 1984.

[ISO 1989] ISO TC-176/SC1, Quality - Vocabulary, ISO 8402, Draft Addendum 2, Oct. 12, 1989.

[ISO 1990] ISO 9000-3: Quality Management and Quality Assurance Standards, Part 3 - Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenance of Software, 1990.

[ISO/IEC 1991] ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety Related Systems, IEC Reference:65A(Secretariat)122, Nov. 1991, pp.88.

[Leveson 1986] Leveson, N.G., "Software Safety: Why, What, and How," ACM Computing Surveys, Vol.18, No.2, pp.125-163, (June 1986).

[Leveson 1990] Leveson, N.G., "Evaluation of Software Safety," ICSE-12, Nice, Mar. 1990, pp.223-224.

[Leveson 1991] Leveson, N.G., "Software Safety in Embedded Computer Systems," Communications of the ACM, Vol.34, No.2, pp.34-46, (Feb. 1991).

[Leveson 1992] Leveson, N.G., "High-Pressure Steam Engines and Computer Software," ICSE-14, Melbourne, Australia, May 1992, pp.2-14.

[MIL 1984] MIL-STD-882B: Military Standard System Safety Program Requirements, Department of Defense, Washington, D.C., 30 Mar. 1984.

[MOD 1989] UK Interim Defence Standard 00-56/1: Requirements for the Analysis of Safety Critical Hazards, Ministry of Defence, Directorate of Standardisation, Kentigern House, 65 Brown St., Glasgow G2 8EX, May 1989.

[MOD 1991] UK Interim Defence Standard 00-50: The Procurement of Safety Critical Software in Defence Equipment: Part 1:Requirements, Part 2: Guidance, Ministry of Defence, Directorate of Standardization,

Kentigern House, 65 Brown Street, Glasgow G2 8EX, 5 April 1991.

[Neumann 1989] Neumann, P.G., "Safety of Computer Systems in Aerospace Applications," Proceedings of AIAA Computers in Aerospace Conference Ⅶ, Oct. 1989, pp.970-974.

[Parnas et al. 1990] Parnas, D.L., van Schouwen, A.J., and Kwan, S.P., "Evaluation of Safety-Critical Software," Communications of the ACM, Vol.33, No.6, pp.636-648, (June 1990).

[Pilsworth 1989] Pilsworth, R., "Defence Standards 00-55 & 00-56," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.103-105.

[Potocki De Montalk 1991] Potocki De Montalk, J.P., "Computer Software in Civil Aircraft," COMPASS '91, June 1991, pp.10-16.

[Pyle 1991] Pyle, I.C., Developing Safety Systems: A Guide Using Ada, London: Prentice Hall International, Reading, 1991, pp.254.

[RTCA 1985] DO-178A: Software Considerations in Airborne Systems and Equipment Certification, Radio Technical Commission for Aeronautics, March 1985.

[Wallace et al. 1992] Wallace, D.R., Kuhn, D.R., and Ippolito, L.M., "An Analysis of Selected Software Safety Standards," IEEE AES Systems Magazine, Vol.7, No.8, pp.3-14, (Aug. 1992). Also available in COMPASS '92, June 1992.

[Wichmann 1992] Wichmann, B.A.(ed), Software in Safety-Related Systems, Chichester: John Wiley & Sons, Reading, 1992, pp.304.

[Wright & Zawilski 1991] Wright, C.L. and Zawilski, A.J., "Existing and Emerging Standards for Software Safety," SESAW '91, San Diego, May 1991, pp.112-118.

第２部　ＩＳＯのソフトウェア安全性規格（案）

中　目　次

Ⅰ．規格（案）の規定内容

＊本　文

＊図１～図５

＊技法および手段を選定するためのガイドライン

＊条に関する表

＊詳　細　表

＊技法の主題目録（省略）

Ⅱ．規格（案）の解説

I. 規格（案）の規定内容

ISO/IEC JTC1/SC7 N917
IEC/TC65/SC65A/WG9　参照番号65A(Secretariat)122　1991年11月発行

産業用安全関連システムの分野で利用されるコンピュータ用ソフトウェア

(Software for Computers in the Application of Industrial Safety-Related Systems)

＜最初の注意書き＞

本文書は、以前に作成された規格案に対して受理されたコメントがコメント文書 65A(Secretariat)110 として取りまとめられたので、それを反映して、IEC/TC65/SC65A/WG9が作成したものである。各国の国内規格委員会は、本規格案に対してコメントする際には、（WG10が作成した）文書65A(Secretariat)123: "Functional Safety of Programmable Electronic Systems - Generic Aspects"（プログラマブル電子システムの機能別安全性 - 汎用的な側面）について考慮することを勧告する。

［訳注］第1条の各条文の先頭にある[1.X.X]形式の項番は本規格案の項番ではなく、「解説編」でその条文との対応を明確にするために付加されたものである。

目　次

1．序　文
　1．1　本国際規格の適用方法
2．適用範囲
3．引用規格
4．定　　義
5．目標および適合性
6．ソフトウェア安全性インテグリティ水準
　6．1　目　標
　6．2　要　件
7．要員とその責務
　7．1　目　標
　7．2　要　件
8．ライフサイクル問題と文書
　8．1　目　標
　8．2　要　件
9．ソフトウェア要求仕様
　9．1　目　標
　9．2　要　件
10．ソフトウェア構造
　10．1　目　標
　10．2　要　件
11．設計および開発
　11．1　目　標
　11．2　要　件
12．検　証
　12．1　目　標
　12．2　要　件
13．ソフトウェア／ハードウェアの統合化
　13．1　目　標
　13．2　要　件
14．ソフトウェア妥当性検査
　14．1　目　標
　14．2　要　件
15．評　価
　15．1　目　標
　15．2　要　件
16．品質保証
　16．1　目　標
　16．2　要　件
17．ソフトウェア保守
　17．1　目　標
　17．2　要　件
18．アプリケーション分野固有の規格を派生するための手引
　18．1　目　標
　18．2　要　件

1. 序　文

[1.0.1] 本国際規格に関しては、ＩＥＣ国際規格65A(Secretariat)123：“Functional Safety of Programmable Electronic Systems - Generic Aspects”（プログラマブル電子システムの機能別安全性 - 汎用的な側面）と併せて読むべきである。

[1.0.2] 本国際規格は、個々のプログラマブル電子システム（PES: Programmable Electronic System）中に存在するソフトウェアに関する安全性インテグリティ（Safety integrity）を達成するのに必要な要件を提供する目的で作成された。本国際規格は専用マイクロプロセッサ、プログラマブル・ロジック・コントローラ（PLC: Programmable Logic Controller）、マイクロプロセッサ分散システム、そして、安全関連ＰＥＳ（Safety-Related PES）のその他の形態中に存在するソフトウェアを網羅している。

[1.0.3] 本国際規格は、“伝統的な”アプリケーション・プログラミングだけでなく、オペレーティング・システムおよびファームウェアをも含めたソフトウェアに適用される。アプリケーション・プログラミングには、高位言語、専用言語（例えば、ＰＬＣラダー・ロジック(Ladder Logic)）、そして、アセンブラ言語によるプログラミングが含まれる。

[1.0.4] 本国際規格での主要な概念は、ソフトウェア安全性インテグリティの各水準の概念である。水準が高くなればなるほど、ソフトウェアの仕様フォールト（Fault）や設計フォールトが原因であるＰＥＳの危険な故障（Failure）の発生確率がそれに比例して低くなる。

[1.0.5] 本国際規格では、安全性インテグリティの４つの水準に対して、技法／手段（Technique and measure）を創作した。ここで、１は最低水準で、４は最高水準である。非安全関連水準、即ち、０は比較のために含めてある。各々の安全性インテグリティ水準および非安全関連水準で要求される技法／手段が表形式で示されている。本国際規格では、所与のリスク（Risk）に対して適切な安全性インテグリティがどの水準かの手引は与えていない。この決定は、アプリケーションの特性、他のシステムがどの程度まで安全機能を果たすか、社会的要因、そして経済的要因をも含めた多くの要因に依存する。

[1.0.6] ソフトウェアに配分される安全機能およびそのソフトウェアに要求されるインテグリティ水準を規定することが、国際規格65A(Secretariat)123：“Functional Safety of Programmable Electronic Systems - Generic Aspects”（プログラマブル電子システムの機能別安全性 - 汎用的な側面）の役割りである。本国際規格では、これらの要件を達成するのに必要な手段を規定する。

[1.0.7] 国際規格65A(Secretariat)123では、［安全関連システムを開発するのに］次に示す手順を実施する系統的アプローチを要求している。

i) ハザード（Hazard）、リスク、そしてリスク評価基準を特定・識別する。
ii) リスク評価基準を満足させるのに必要なリスク軽減を特定する。
iii) 要求されたリスク軽減を達成するのに必要な安全防護（Safeguard）に関する全体的な安全性要求仕様を規定する。
iv) 適切なシステム構造（System Architecture）を選定する。
v) 仕様を妥当な安全性能（即ち、安全性インテグリティ）を保持した安全関連システム（Safety-Related System）に変換するのに必要とする技術的および管理的な活動を計画し、監視し、そして管理する。

[1.0.8] 国際規格65A(Secretariat)123では、アプリケーション用のリスク評価基準を満足させるのに必要なリスク軽減に関して、上で詳しく述べたとうりに、図1で図解した4つの水準の内、1つの水準を規定する予定である。リスクは、ハザードな事象（Hazardous event）の結末（即ち、厳しさ(Severity)）と頻度との両方を評価した結果に基づいている。また、図1は、当該仕様を分解し、（複数個の）安全関連システムと（複数個の）構成要素（Component）からなる設計案を作成した後で、安全性インテグリティの配分が実行されることを示している。結局は、こうすることによって、機能要件とともに、ソフトウェアに対して要求された安全性インテグリティ水準が達成される。ソフトウェアの安全性インテグリティ水準は、本規格の出発点となっている。

[1.0.9] 表面上は、運用信頼度が高いシステムは、自動的に安全なシステムであると仮定される。しかし、詳細な解析によると、高信頼度は必要ではあるが、安全性を確かなものにする（ensure）のには、それだけでは十分でないことが示されている。

[1.0.10] 高信頼性は、システムの目的と、意図された行動（サービス）とを指向している。即ち、システムが規定された仕事を果たす期待の度合である。信頼性要件は、システムが継続的にサービスを提供することに関係する。

[1.0.11] 安全性は、望ましくない結果をもたらす可能性のある事象やシーケンス（Sequence）の原因および結末に関係がある。安全性要件は、システムが事故（Accident）の原因にならなくすることに関係する。安全性要件では、ある事象が事故の原因となる可能性があるので、システムがハザードな状態、即ち、不安全状態に至ることがないことを確かなものにすること。

[1.0.12] 安全性の観点では、システムが安全性要件に違反しない限り、仮にシステムがその目的を果たさなくても、必ずしも問題とはならない。逆に、システムが高信頼ではあるが、不安全であることもありうる。

[1.0.13] 2台のコンピュータが常用冗長（Active redundancy）であるフェール・セーフ・システム（Fail-safe system）について考えてみる。2台のコンピュータの出力は、その内の1台のコンピュータの起こりうる故障を検出するために比較される。そのような故障が発生した後では、高信頼性あるいは高安全性のどちらを達成するかに応じて、その事態（situation）を取り扱う2つの戦略がある。

i) 安全性に関して、高水準が要求されている場合には、コンピュータは停止されなければならず、同時に、その（制御対象の）プロセス（Process）は安全フォールバック状態（Safe fall-back state）に至らせなければならない。たとえ故障したコンピュータがどれであるかを特定する確率が高くても、冗長性がないと、更なる誤りのある出力を検出するのは不可能である。

ii) 信頼性に関して、高水準を達成しなければならない場合には、故障したコンピュータを特定する可能性があるのならば、（多分）健全なコンピュータを使用して、いかなる冗長性もなく、運転を継続することは可能である。

[1.0.14] この例は、安全性と信頼性との間にはトレードオフ（trade-off）があることを示している。事実、設計中には、考慮すべきその他のトレードオフが存在する。また、コストだけでなく、アベイラビリティ（Availability；稼動率）、保守性（Maintainability）、そしてセキュリティ（Security）のようなシステム

品質特性についても考慮する必要がある。本国際規格では、これらのトレードオフに対する手引を与えることは規定範囲外としている。

［注］IEC/TC56/WG1では、信頼性、アベイラビリティ、保守性、安全性、そしてセキュリティのようなシステム品質特性の一式は、包括的な用語“ディペンダビリティ”（Dependability）と名称することを提案した。

[1.0.15] ソフトウェアは、それ自身の信頼性を有している。即ち、入力が投入された後で、意図された結果が得られる確率である。ソフトウェア安全性の場合には、全体のシステムを調べる必要がある。即ち、ソフトウェアは特定のアプリケーションの文脈下で調べられなければならない。整列プログラムの場合、もし誤りのある整列アルゴリズムが原因でハザードになる可能性がないのならば、整列用の安全プログラムについて言及しても意味がない。

[1.0.16] 現状の技術水準では、品質保証手法（いわゆる予防手段）あるいはソフトウェア・フォールト・トレラント・アプローチ（Software fault tolerant approach）が適用されても、システムの絶対的な安全性が達成されたことを保証（guarantee）することはできない。かなり複雑な安全関連システム中にフォールトが存在しない、特に仕様フォールトや設計フォールトがないことを証明する既知の方法がない。

[1.0.17] 高インテグリティ・ソフトウェアを開発する際に適用される諸原則は下記の通りである。

- トップ・ダウン設計手法（Top-down design method）
- モジュラリティ（Modularity）
- 開発ライフサイクルの各フェーズでの検証
- 検証済みのモジュールとモジュール・ライブラリ
- 明解な文書
- 監査可能な文書
- 妥当性検査テスト（Validation testing）

[1.0.18] 本規格では、ソフトウェア安全性インテグリティ水準が異なれば、これらの原則と関連する諸原理・原則を正しく適用していることを保証（assurance）する点に関しては、異なる水準を要求している。

１．１　本国際規格の適用方法

[1.1.1]　本国際規格の適用の際の機能的ステップは図２に示されており、下記の通りである。

i)　ＰＥＳの安全性要求仕様（書）を入手するか、あるいは作成する(第６条)。
ii)　ソフトウェアに配分されたすべての安全機能を特定し、インテグリティ水準を決定する（第６条）。
iii)　ソフトウェア用の安全性要求仕様をより詳細に規定し、それと併行して、ソフトウェアの構造を検討する。ここで、ソフトウェア構造仕様は当該ソフトウェア用に開発された基本安全性戦略である（第９条、第１０条）。
iv)　ソフトウェア品質計画（SQP: Software Quality Plan）、ソフトウェア・インテグリティ水準、そしてソフトウェア・ライフサイクルに従って、ソフトウェアを設計・開発し、テストする（第１１条）。
v)　ソフトウェアとターゲット・ハードウェア（Target hardware）とを統合する（第１３条）。
vi)　ソフトウェアの妥当性検査を行う（第１４条）。
vii)　妥当性検査済みのソフトウェアとソフトウェア妥当性検査報告書（SVR: Software Validation Report）をシステム技術者に引き渡し、全体のＰＥＳシステムに統合してもらう（第１４条）。
viii)　運用寿命中にソフトウェアの保守が要求された場合には、本国際規格を適切に再度実施する（第１６条、図５）。

[1.1.2]　ソフトウェア開発では、たくさんの活動が実行される。これらには、検証（第１２条）、評価（第１５条）、そして品質保証（第１６条）が含まれる。

[1.1.3]　本規格では、特定のソフトウェア開発ライフサイクルを強制していない。しかし、ライフサイクルおよび文書一式について勧告している（第８条、図３、図４）。

[1.1.4]　また、要件として、ソフトウェア開発に参画する要員の能力についても規定している（第７条）。

[1.1.5] ステップiii)以降では、要求されたインテグリティ水準にふさわしい技法／手段が選定される。この選定を支援するために、4つの安全性インテグリティ水準に対して様々な技法／手段を階級付けて、表形式で定式化している。表の引用欄は、各々の技法／手段について簡単な解説をしている主題目録（Bibliography）を指している。主題目録には、詳しく調べるための参考文献が示されている。

2. 適用範囲

2.1 本国際規格は、「ソフトウェア」と「ソフトウェアと“より大きいシステム”との間での相互対話（interaction）」のみに適用される。

2.2 本国際規格は、死亡、手足の損傷、そして病気を含めた人間の安全に主として関係し、安全関連システムで利用されるソフトウェアに集中している。とは言ってもやはり、故障の結末が重大な経済的あるいは環境的な影響を与える場合についても言及していることを認めている。そのようなケースの場合には、本国際規格は、装置あるいは製品に対する保護（Protection）を評価するのに適用できよう。

2.3 本国際規格は、

i) 汎用基盤であり、アプリケーションとはかかわりなく、安全関連ソフトウェア（SRS: Safety-Related Software）の開発と評価の両方に適用される；（［訳注］ここでは、略語ＳＲＳはSafety-Related Softwareの略語のみに使用し、Safety-Related SystemやSafety Requirements Specificationの略語としては使用しないことにする。）

ii) アプリケーション分野固有の勧告を開発する責任を有するＩＥＣ技術委員会や他の規格作成委員会に対して、アプリケーション分野固有の規格を開発するのを助長する；

iii) 安全機能を有し、しかし、アプリケーション分野固有の勧告を入手できないＰＥＳ用のソフトウェアに対する手引を提供する；

iv) “Functional Safety of Programmable Electronic Systems - Generic Aspects”（プログラマブル電子システムの機能別安全性 - 汎用的な側面）を取り扱っている国際規格65A(Secretariat)123と相互補完の関係にある；

v) プログラマブル電子装置（PE: Programmable Electronics）を含めて安全関連システムで利用されるソフトウェアに適用される。

2．4　本国際規格は、産業用制御システムで利用されるコンピュータで必要とするＳＲＳに適用される。そのようなシステムには、安全アクチュエーション系（Safety actuation system）、安全系支援機構、そして安全保護系が含まれている。これらのシステムは、専用マイクロプロセッサ、プログラマブル・ロジック・コントローラ（PLC）、マイクロプロセッサ分散システム、大規模中央処理装置システム、そして、ＰＥＳのその他の形態を使用して構築してもよい。

2．5　本国際規格は、アプリケーション・プログラミングだけでなく、安全関連システムの構築および開発で利用されるオペレーティング・システム、支援ツール、そしてファームウェアをも含めたソフトウェアに適用される。アプリケーション・プログラミングには、高位言語プログラミング、低位言語プログラミング、そして専用言語プログラミング（例えば、ＰＬＣラダー・ロジック・プログラミング、コンピュータ数値制御対象部品プログラミング）が含まれる。

3. 引用規格

次に示す規格は、本規格からの引用により、本規格の条項の一部をなす。本規格の発行時点では、ここに示す版の規格が有効である。すべての規格は改訂されることがあるので、本規格を使う当事者は、以下に列挙した規格の最新版を適用できるかを検討することが望ましい。ＩＥＣとＩＳＯの会員団体は、国際規格の有効な最新版の登録を保管している。

ISO 9001(1987)　Quality Systems - Model for Quality Assurance in Design/Development, Production, Installation and Servicing.
（品質システム - 設計／開発、製造、据付けおよびアフターサービスの品質保証モデル）

ISO 9000-3(1990)　Quality Management and Quality Assurance Standards, Part 3 - Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenance of Software.
（品質管理および品質保証の規格集、第３分冊 - ISO 9001のソフトウェアの開発、供給、保守への適用のための指針）

IEC 300(1984)　Reliability and Maintainability Management.
（信頼性および保守性管理）

IEC 605　Equipment Reliability Testing.
（装置信頼性試験）

IEC 706　Guide on Maintainability of Equipment.
（装置の保守性に関する手引）

IEC 812(1985)　Analysis Techniques for System Reliability - Procedure for Failure Mode and Effects Analysis.
（システム信頼性解析技法 - 故障モード影響解析の手順）

IEC 863(1986)　Presentation of Reliability, Maintainability and Availability Predictions.
（信頼性、保守性、アベイラビリティ予測の表現法）

IEC 880(1986)　Software for Computers in the Safety Systems of Nuclear Power Stations.
（原子力発電所の安全系で利用されるコンピュータ用ソフトウェア）

IEC 1014(1989)　Programmes for Reliability Growth.
（信頼度成長に関するプログラム）

IEC 1025(1990)　Fault Tree Analysis.
（故障の木解析）

IEV 191(1986)　International Electrotechnical Vocabulary
Document 1/56 Secretariat 193/167.
（国際電気標準会議の語彙文書 1/56 Secretariat 193/167）

56(Central Office)137　Analysis Techniques for System Reliability - Reliability Block Diagram Method.
Note: This standard will be published as IEC 1078.
（システム信頼性解析技法 - 信頼性ブロック図法
［注］：本規格はIEC 1078として発行される予定である。）

56(Central Office)148　Guide on Formal Design Review.
（公式設計レビューの手引）

56(Central Office)150　Reliability Growth Models and Estimation Methods.
（信頼度成長モデルと推定法）

56(Secretariat)273　Software Reliability and Maintainability Management.

（ソフトウェア信頼性および保守性管理）

56(Secretariat)280　Application of Markov Techniques.
（マルコフ技法の適用）

65A(Central Office)21 Programmable Controllers, Part 1 - General Information.
（プログラマブル・コントローラ、第１分冊 - 全般的な情報）

65A(Central Office)22 Programmable Controllers, Part 2 - Equipment Characteristics.
（プログラマブル・コントローラ、第２分冊 - 装置特性）

65A(Secretariat)90　Programmable Controllers, Part 3 - Programming Languages.
（プログラマブル・コントローラ、第３分冊 - プログラミング言語）

65B(Secretariat)150　Programmable Controllers, Part 4 - User Guidelines.
（プログラマブル・コントローラ、第４分冊 - 利用者指）

65B(Secretariat)138　Programmable Controllers, Part 5 - Manufacturing Message Specification.
（プログラマブル・コントローラ、第５分冊 - 製造メッセージ仕様書）

65A(Secretariat)123　Functional Safety of Programmable Electronic Systems - Generic Aspects.
（プログラマブル電子システムの機能別安全性 - 汎用的な側面）

4. 定　義

本国際規格における用語の意味については、以下の定義を適用する。

＊評価者（Assessor）

ソフトウェアに対するすべての安全性要件およびその他の要件が達成されているか否かを評価する責任を有する独立の人または機関。

＊コンピュータ数値制御装置（CNC: Computer Numeric Controller）

数値情報によって機械や装置を直接、効果的に制御するもの。

＊設計フォールト（Design Fault）

システムの設計中に起こる（人間の）失敗（failure）に起因する設計（コーデング、要求仕様作成）フォールト。設計フォールトは、ある特定の入力値が（サブ）システムに入力された時に、その生成された結果が仕様に適合（conform）せず、システム中でまだ検出されておらず、システムに内在して誤り（Error）の原因となる。これは、当該（サブ）システムの故障（Failure）を引き起こす。もし、同じ入力値が再び生起した時には、同じ誤りのある結果を生成することになる。

＊多様性（Diversity）

要求された機能を果たす異なる手段が存在する度合、例えば、別の物理原理、同じ問題を解決する別の方法が存在する度合。

＊誤り（Error）

システムが誤っているので、故障が起こる。それ故に、誤りとは、種々のシステム状態内で、たぶん故障を導きそうなシステム状態である。簡潔に言えば、誤りとは、合意した要求仕様からの検出された逸脱（deviation）である。（IEVの定義とは異なっている。）

＊フェール・セーフ（Fail-Safe）

予測された（あるいは規定された）装置（あるいはサービス）故障モードがシステム故障モードの原因となる時だけに限って、システムを安全フォールバック状態（Safe fall-back state）に至らせ、その状態を維持させるシステムに組み込まれた能力（capability）。

＊故障（Failure）

出荷されたサービスが規定されたサービスから逸脱している時に起こるシステム故障。ここで、仕様とは、期待されるサービスに関して合意された記述である。簡潔に言えば、故障とは、システムあるいはソフトウェア中に存在する誤りの顕在（manifestation）である。（ＩＥＶの定義とは異なっている。）

＊フォールト（Fault）

フォールトは誤りの原因である。（ＩＥＶの定義とは異なっている。）

［注］

故障、誤り、およびフォールトの定義に関しては、システムの文脈下で見なければならない。上の定義は、この階層的システム分解のある１つの水準に注目（refer）している。

このように、誤りは当該システム中に存在するフォールトの顕在であり、故障は、当該サービスに対する誤りの影響・効果である。あるシステム構成要素のフォールトは、見ている階層水準に応じて、２つの異なる見方（way）で考察可能である。

もし、当該構成要素自体がシステムと見なされる（低水準の視点）場合には、この故障は、当該構成要素中に存在するフォールトが原因となって内部的な誤りのある状態が発生した結果である。

次のより高いシステム水準から見れば、当該構成要素のその故障は、直上位のシステム水準での誤りのある状態や故障を発生させる原因となる可能性のあるフォールトである。

一目で、本国際規格での故障、誤り、およびフォールトの定義に関しては、ＩＥＶで規定した定義とは逸脱しているようにみえる。

これらのほんの少し異なる定義を使用した理由は、それらの定義がシステム階層構造（System structure）に関する明確な階層概念に注目しているという事実に起因している。

ＩＥＶで使用している用語“故障／誤り／フォールト”は、システム分解中での見ている当該水準を無視して、当該システムの構成要素のみに注目しているという点を認識することによって、ＩＥＶで規定した定義と本定義との連結が確立されるかもしれない。このことは、ある品目の故障が、この構成要素に関して１つのフェーリング（サブ）システム（Failing (sub-) system）と見なされるという視点に対応していることを意味している。別の言葉で言えば、次

のより高いシステム水準から見れば、この構成要素、即ち、品目はフォールティ（faulty）である。

＊フォールト・アボイダンス（Fault Avoidance）

システムの設計および構築中に、フォールトが入り込むのを避けることを狙いとする設計技法の使用。

＊フォールト・トレランス（Fault Tolerance）

継続的な正しい処理実行を提供するためにシステムに組み込まれた能力（capability）、即ち、ハードウェアあるいはソフトウェアの限定された個数のフォールトに直面した際に、規定された通りにサービスを提供すること。

＊独立会社（Independent Company）

独立会社とは、安全関連ソフトウェア（SRS）の主要な開発および保守を実施している会社と財政面、管理面、およびその他の資源面に関して、分離され、異なっており、かつ、これらの主要な活動に関して直接的な責任を有しない会社である。

＊独立部門（Independent Department）

独立部門とは、安全関連ソフトウェア（SRS）の主要な開発および保　守を実施している部門と財政面、管理面、およびその他の資源面に関して、当然分離され、異なっており、かつ、これらの主要な活動に関して直接的な責任を当然有しない部門である。

＊独立者（Independent Person）

独立者とは、安全関連ソフトウェア（SRS）の主要な開発および保守を実施している人達と財政面、管理面、およびその他の資源面に関して、当然分離され、異なっており、かつ、これらの主要な活動に関して直接的な責任を当然有しない人である。

［注］会社の組織構造および会社内で蓄積されている専門知識（In-company expertise）に応じて、独立性に関する要件を満たすために、外部の組織を利用してもよい。各アプリケーション分野固有の規格を開発する際には、独立性に関する厳密な度合について、当該アプリケーション分野に責任を有する委員会によって、その特定のアプリケーションを勘案して

明確に規定されること。

＊保守性（Maintainability）

所与の条件下で、修理あるいは元の状態を回復して、その品目に要求された機能を実行可能な状態にする品目の能力（ability）。

＊ＰＥＳの安全性要求仕様書（PES Safety Requirements Specification）

プログラマブル電子システム（PES）のすべての安全性要件およびその他の要件を含んでいる、国際規格65A(Secretariat)123に従って作成された文書。

＊プログラマブル電子システム（PES: Programmable Electronic System）

制御、安全保護、あるいは監視の目的で、プラントにあるセンサ（Sensor）および／またはアクチュエータ（Actuator）が結合された、１台以上のコンピュータからなるシステム。

＊プログラマブル・ロジック・コントローラ（PLC: Programmable Logic Controller）

特定の機能を実現するための命令を記憶するユーザ用のプログラマブル・メモリを保持している固体回路（Solid-state）の制御システム。

＊冗長性（Redundancy）

別の構成要素あるいはシステムが動作状態あるいは故障状態であるかとは係わりなく、要求された機能を実行可能なようにするための追加的な構成要素あるいはシステムが用意されている度合。冗長性は、同一の構成要素（同種冗長性）あるいは異なる構成要素（異種冗長性）によって実現可能である。

＊信頼度（Reliability）

システムが規定された条件下で、ある特定の期間中、要件に関する仕様書で記述された機能を果たす確率。

＊リスク（Risk）

規定されたハザードな事象の頻度、即ち、確率と、その結末との組み合せ。リスクの概念は、２つの要素、即ち、ハザードが起こる頻度／確率と、そのハザードな事象の結末とから必ずなっている。

＊安全状態（Safe State）

ある特定の前提および規定された条件下で、人命（手足および健康）、経済、あるいは環境が危険にさらされない規定されたシステム状態。

＊安全性（Safety）

システムが規定された条件下で、人命（手足および健康）、経済、あるいは環境を危険にさらす状態に導かない期待の度合。

［注］システム安全性の場合には、不安全状態を導く故障のすべての原因には、ハードウェア故障、ソフトウェア故障、電子干渉が原因である故障、人間との対話が原因である故障、そして、制御対象自体の故障が含まれる。これらの型の故障の範疇には、特にランダム・ハードウェア故障の範疇には、危険モードになる故障率（Failure rate）、あるいは安全保護システムが要求された時に運転を失敗する確率のような尺度によって定量化可能なものもある。また、システム安全性は、定量化できず、しかも定性的にしか考察できない多くの要因に依存する。

＊安全性インテグリティ（Safety Integrity）

プログラマブル電子システム（PES）が規定されたすべての条件下で、規定された期間中、安全機能を達成する尤度。

＊安全関連ソフトウェア（SRS: Safety-Related Software）

システムが人命（手足および健康）、経済、あるいは環境を危険にさらすことがないことを確かなものにするソフトウェア。

＊ソフトウェア（Software）

データ処理システムの操作に関係するプログラム、手順、規則、およびあらゆる関連する文書からなる知的な創作物（ISO規則集：ISO/2382/1、第2版、1984.10.1を参照）。

＊ソフトウェア・ライフサイクル（Software Lifecycle）

ソフトウェアの必要性が認識された時に開始し、そのソフトウェアがもはや利用されなくなった時に終了する期間中に発生する諸活動。典型的には、ソフトウェア・ライフサイクルは、要求仕様作成フェーズ、開発フェーズ、テスト・フェーズ、統合化フェーズ、設置フェーズ、そして保守フェーズからなる。

＊ソフトウェア安全性インテグリティ（Software Safety Integrity）

プログラマブル電子システム（PES）中に存在するソフトウェアがすべての規定された条件下で、規定された期間中、安全機能を達成する尤度。

＊システム（System）

システムは、設計に従って相互に対話する一連の構成要素からなるものと定義されている。システムの構成要素は、（サブシステムと呼ばれる）別のシステムであることもある。そのような構成要素（サブシステム）は、該当する水準に依存して、下記のものかもしれない。

・制御するシステム、あるいは制御されるシステム
・ハードウェア、ソフトウェア、人間との間での相互対話

＊妥当性検査（Validation）

統合化されたソフトウェア・システムを対象にして、安全性、機能、性能、およびインタフェースに関する要件に適合しているか否かを確認するために行われるテスト。

＊検証（Verification）

ソフトウェア開発工程の各フェーズでの成果物が、前のフェーズで規定された要件すべてを満たしているか否かを決定する過程。検証には、テストを含めなければならない。

5. 目標および適合性

5.1 本条以降の各条では、当該条の目標および要件が詳細に示されている。

5.2 （ＳＲＳが）本国際規格に適合していることを示すには、当該条の要件の各々が規定された当該インテグリティ水準に応じて満足され、その結果、当該条の目標が満たされたことを示さなければならない。

5.3 要件が、用語“インテグリティ水準として要求される度合”で限定されている場合、要件を満足させるのに、一連の技法／手段を使用することができることを示している。

5.4 5.3項が該当する場合には、本国際規格で詳細に示した諸表は、要求されたインテグリティ水準にふさわしい技法／手段の選定を容易にするのに利用されるべきである。

5.5 もし、ある技法／手段が、表中でＨＲ（Highly Recommended；強く勧告）として階級付けされている時で、それを使用しない場合には、その技法類を使用しない正当事由を、適当と考えられる品質計画書あるいはソフトウェア構造仕様書（SAS: Software Architecture Specification）に詳細に記述すること。

5.6 もし、表中にない技法／手段の使用を提案した時には、当該条の特定の要件および全体的な目標を満たしているかの点に関し、その技法／手段の有効性（Effectiveness）および適切性（Suitability）について正当化し、適当と考えられる品質計画書あるいはソフトウェア構造仕様書（SAS）にその正当事由を記録すること。

5.7 特定の条の当該要件と表中の各々の技法／手段に関する準拠性（Compliance）については、本規格で要求した文書類を査読し、立会いテスト（Witnessing test）を行って、評価すること。

5.8 本国際規格では、技法とその正しい適用とを一括して使用することを要求している。これらの技法は、表で使用が要求され、その技法の詳細は主題目録にある。しかし、本国際規格で勧告した技法／手段を使用したからといって、要求されたインテグリティ水準が達成されることを保証（guarantee）してはいない。

6. ソフトウェア安全性インテグリティ水準

6.1 目　標

6.1.1　本条の目標は、ソフトウェアにインテグリティ水準を配分することである。

6.2 要　件

6.2.1　下記の内容を含むＰＥＳの安全性要求仕様書が作成されていること。

- 安全機能
- システムの構成（Configuration）あるいは構造（Architecture）
- ハードウェア信頼性要件
- 安全性インテグリティ要件
- 容量（Capacity）および応答時間性能
- 装置と操作員とのインタフェース
- その他の機能およびその属性

［注］ＰＥＳの安全性要求仕様の開発に関してはIEC 65A(Secretariat)123の要件に従って実行されるべきである。ソフトウェアに対する安全性インテグリティ水準に関してはIEC 65A(Secretariat)123の要件に従って規定すべきである。

6.2.2　ソフトウェアに対して要求される安全性インテグリティ水準は、ソフトウェアに関係するリスクの水準に基づいて決定されること。

6.2.3　考慮すべきリスクは、下記に示すハザードな結末に関係するリスクである。

- 死　亡
- 損傷あるいは病気
- 環境汚染
- 財産の損失あるいは損害

6.2.4　リスクは定量的な確率で、あるいは定量的な確率の組み合せで規定可能であるが、しかし、同様なやり方で、ソフトウェア安全性インテグリティを規定するのは不可能である。それ故に、本国際規格の場合には、ソフトウェア安全性インテグリティは、下記の5つの水準の1つで規定すること。

水準	ソフトウェア安全性インテグリティの説明
4	非常に高い
3	高　い
2	中　位
1	低　い
0	非安全関連

6.2.5　ソフトウェア安全性インテグリティ水準は、ソフトウェア要求仕様書（第9条）で規定すること。

6.2.6　ソフトウェア安全性インテグリティ水準として0に分類されたソフトウェア、即ち、非安全関連ソフトウェアの場合には、供給者および／または開発者は、ISO 9000規格シリーズに準拠した品質システムを少なくとも使用し、これらの規格の関係する部分、即ち、ソフトウェア製品用のISO 9000-3規格あるいは同等規格（Equivalent Standard）を履行することを勧告する。

7. 要員とその責務

7.1 目　標

7.1.1　本条の目標は、ソフトウェアに関して責務（Responsibility）を有するすべての要員について、これらの責務を果たす能力があることを確かなものにすることである。

7.2 要　件

7.2.1　ＳＲＳの設計、開発、評価、据付け、保守、あるいは管理の活動に参画するすべての要員は、適切に訓練され、経験があり、そして適切な資格を有していること。必要な厳格な訓練および一定の資格については、ここでは規定されておらず、特定の産業ごとに解釈することが要求される。

7.2.2　設計、開発、評価、据付け、保守、あるいは管理の活動に参画するすべての要員の訓練、経験および資格は、特定のアプリケーションおよび要求されているインテグリティ水準を勘案して、正当化されていること。

7.2.3　7.2.2項に含まれる正当事由は、適切な文書、例えば、品質計画書に記録されていること。正当事由には、下記のそれにふさわしい分野の能力に関する証拠が含まれていること。

i)　アプリケーション分野にふさわしい工学
ii)　ソフトウェア工学
iii)　コンピュータ・システム工学
iv)　安全工学
v)　法体系および規制面

7.2.4　インテグリティ水準として要求される度合に応じて、ソフトウェアを対象とする評価者を任命すること。

7.2.5　評価者には、ソフトウェアの評価を実行する際に必要とする十分な権限が与えられていること。

7.2.6　評価者は、設計／開発チームおよびすべてのプロジェクトに関係する文書にアクセスすることができること。

8. ライフサイクル問題と文書

8.1 目　標

8.1.1　本条の目標は、ソフトウェアの開発を体系化して、規定されたフェーズおよび活動に組み入れることである。

8.1.2　本条の別の目標は、ソフトウェアのライフサイクルを通じて、ソフトウェアに関係するすべての情報を記録することである。

8.2 要　件

8.2.1　ソフトウェア開発の1つのライフサイクルを選定し、本国際規格の第16条に従って、品質計画書に詳細に記述すること。

8.2.2　品質保証手順は、ライフサイクル活動と併行して実行されること。

8.2.3　ソフトウェア・ライフサイクルの各フェーズは、基本タスクに分割すること。基本タスクは、良く規定された入力、出力、そして活動からなる。

8.2.4　各フェーズは、検証報告書で完了していること。

8.2.5　すべての文書は、設計工程の進展と併行して、継続的に拡張が可能なように構造化されていること。

8.2.6　各文書は、ユニークな参照番号が与えられ、他の文書との規定された関係を保持していること。

8.2.7　すべての文書の内容は、その文書の特定の目的に適した形式で記録されていること。

8.2.8　品質計画書にその正当事由が記述されていない場合には、図3および図4で示した開発ライフサイクルの1つを利用し、下記に示す関連する文書を作成すること。

フェーズ	文書
ソフトウェア・システム要求仕様作成フェーズ	ソフトウェア要求仕様書 (Software Requirements Specification)
	ソフトウェア要求仕様テスト仕様書 (Software Requirements Test Specification)
	ソフトウェア妥当性検査計画書 (Software Validation Plan)
	ソフトウェア構造仕様書 (Software Architecture Specification)
	ソフトウェア／ハードウェア統合化計画書 (Software/Hardware Integration Plan)
	ソフトウェア要求仕様検証計画書 (Software Requirements Verification Plan)
	ソフトウェア要求仕様検証報告書 (Software Requirements Verification Report)
ソフトウェア設計フェーズ	ソフトウェア設計仕様書 (Software Design Specification)
	ソフトウェア設計テスト仕様書 (Software Design Test Specification)
	ソフトウェア設計検証計画書 (Software Design Verification Plan)
	ソフトウェア設計検証報告書 (Software Design Verification Report)
ソフトウェア・モジュール設計フェーズ	ソフトウェア・モジュール設計仕様書 (Software Module Design Specification)
	ソフトウェア・モジュール・テスト仕様書 (Software Module Test Specification)
	ソフトウェア・モジュール検証計画書 (Software Module Verification Plan)
	ソフトウェア・モジュール検証報告書 (Software Module Verification Report)

コード化フェーズ……ソース・コードおよび支援文書
(Source Code and supporting documentation)
コード検証計画書
(Code Verification Plan)
コード検証報告書
(Code Verification Report)

ソフトウェア・テストフェーズ……ソフトウェア・モジュール・テスト報告書
(Software Module Test Report)
ソフトウェア設計テスト報告書
(Software Design Test Report)
ソフトウェア要求仕様テスト報告書
(Software Requirements Test Report)

ソフトウェア／ハードウェア統合化フェーズ……ソフトウェア／ハードウェア統合化報告書
(Software/Hardware Integration Report)

ソフトウェア妥当性検査フェーズ…ソフトウェア妥当性検査報告書
(Software Validation Report)
ソフトウェア評価報告書
(Software Assessment Report)

ソフトウェア保守フェーズ……ソフトウェア保守記録簿
(Software Maintenance Records)
ソフトウェア保守日誌
(Software Maintenance Log)

8.2.9 8.2.8項が該当する場合には、当該開発ライフサイクルと交差して実行されている諸活動（例えば、検証）に関する個別の計画書あるい報告書の代わりに、各細区分（subsection）との対応が明確になっている1つにまとめた単一の計画書あるいは報告書を使用することがある。

［注］検証には、設計レビュー、インスペクション（Inspection）およびテストが含まれる。

9. ソフトウェア要求仕様

9.1 目　標

9.1.1　本条の目標は、ＰＥＳの要求されたインテグリティ水準を達成するのに必要な「ソフトウェアの要求されたインテグリティ水準」および「ソフトウェア要件」に関して、ＰＥＳの安全性要求仕様書から引き出してソフトウェアに対する完備な要求仕様のセットを規定することである。

9.2 要　件

9.2.1　ソフトウェア要求仕様はＰＥＳの安全性要求仕様書と、品質計画書中で引用しているあらゆる要件から引き出すこと。

9.2.2　ソフトウェア要求仕様は、プロジェクトではなく、製品に関する要求された特性で表現されていること。これらの特性には、ソフトウェアに対して要求された機能、信頼性、そして安全性に関する特性が含まれる。

9.2.3　インテグリティ水準として要求される度合に応じて、ソフトウェア要求仕様書は、下記に示すような方法で表現され、構造化されていること。

i)　明確で、厳密で、議論の余地がなく、検証可能で、保守が容易で、そして実現可能である。

ii)　国際規格65A(Secretariat)123から引き出されたすべてのＰＥＳの安全性要求仕様に立ち返ることが可能である。

9.2.4　ソフトウェア要求仕様書には、全体のシステムの設計、運転および保守に参画する要員すべてが理解可能なような複数の様式での表現および記述が含まれていること。

9.2.5　ソフトウェア要求仕様書は、下記に示す内容で表現されていること。

i)　安全機能および要求されたインテグリティ水準
ii)　容量および応答時間性能
iii)　ＰＥＳシステムと操作員とのインタフェース
iv)　その他の機能または属性

9.2.6　ソフトウェア要求仕様書では、直接結合が存在する、あるいは計画されているどのような場合においても、制御下にある装置（EUC）の内側あるいは外側のいずれかに存在する当該システムとあらゆる他のシステムとの間のすべてのインタフェースを特定し、文書化すること。

9.2.7　問題とされる運転モード（Mode of operation）すべてがソフトウェア要求仕様書に詳細に記述されていること。

9.2.8　ＰＥ（プログラマブル電子装置）で問題とされる挙動モード（Mode of behaviour）すべてが、特に、故障挙動がソフトウェア要求仕様書に詳細に記述されていること。

9.2.9　ハードウェアとソフトウェアとの間の制約条件すべてがソフトウェア要求仕様書で特定され、文書化されていること。

9.2.10　ソフトウェア要求仕様書には、ソフトウェアが自己監視を可能とする要件と、ＰＥＳの安全性要求仕様書で要求された度合に応じて、ソフトウェアがＰＥＳハードウェアに対する監視を可能とする要件が含まれていること。

9.2.11　ソフトウェア要求仕様書には、ＰＥＳの安全性要求仕様書で要求された度合に応じて、安全機能に対する周期テストの要件が含まれていること。

9.2.12　ソフトウェア要求仕様書には、すべての安全機能に対して全体のシステムの運転中にテスト可能にする要件が含まれていること。

9.2.13　ソフトウェアがＰＥＳのインテグリティを達成するのに関連する機能を果たすことを要求されている場合には、これらのことについて、ソフトウェア要求仕様書ではっきりと特定されていること。

9.2.14　ソフトウェアが非安全関連機能を果たすことを要求されている場合には、これらのことについて、ソフトウェア要求仕様書ではっきりと特定されていること。

9.2.15　ソフトウェア要求仕様テスト仕様書は、ソフトウェア要求仕様書に基づいて開発され、供給者および顧客の同意が得られていること。

１０．ソフトウェア構造

１０．１目　標

10.1.1　本条の目標は、要求された安全性インテグリティ水準を考慮して、ソフトウェア要求仕様書で規定した要件を達成するソフトウェア構造を選定することである。

10.1.2　本条の別の目標は、ＰＥＳ構造によってソフトウェアに課せられた要件をレビューすることである。

10.1.3　本条の別の目標は、ハードウェアとソフトウェアとの間での相互対話に関し、安全性に重大な影響をもたらすものを特定し、評価することである。

１０．２要　件

10.2.1　提案されたソフトウェア構造が、ソフトウェア供給者および／または開発者によって確定され、ソフトウェア構造仕様書に詳細に記述されていること。

10.2.2　ソフトウェア構造仕様書では、ソフトウェア要求仕様を要求されたインテグリティ水準で達成するソフトウェアの実現可能性について考察すること。

10.2.3　10.2.2項を適用した後でのＰＥＳの安全性要求仕様書に対して要求されたいかなる変更に関しても、ＰＥＳ開発者との合意に達し、ソフトウェア構造仕様書に記録されること。

10.2.4　ソフトウェア構造仕様書では、ハードウェアとソフトウェアとの間での相互対話すべてに関し、安全性に重大な影響をもたらすものを特定・評価し、詳細に記述すること。

10.2.5 ソフトウェア構造仕様書では、すべてのソフトウェア構成要素を特定し、これらの構成要素に関して下記の点を特定すること。

i) これらの構成要素は新規、既存、あるいは所有権があるかどうか。
ii) これらの構成要素は既に検証済みであるかどうか。もし検証済みならば、その検証条件はどうであったか。
iii) 各々の構成要素は安全関連のものか、そうでないものか。
iv) 構成要素のソフトウェア・インテグリティ水準はどうか。

10.2.6 ソフトウェアが安全機能と非安全機能の両方を実現している場合で、各機能間で十分な独立性がない時には、そのソフトウェアは安全関連として取り扱うこと。

10.2.7 ソフトウェア構造によって、当該アプリケーションの安全関連部分を最小にすること。

10.2.8 ソフトウェアが異なるインテグリティ水準を有する複数個の安全機能を実現している場合で、異なるインテグリティ水準を有する安全機能の間で十分な独立性があることを示せない時には、ソフトウェアすべては最も高いインテグリティ水準に属するものとして取り扱うこと。

10.2.9 ソフトウェア構造仕様書では、インテグリティ水準として要求される度合に応じて、ソフトウェアの開発、統合化、検証、妥当性検査、そして保守に関する戦略を特定すること。

10.2.10 ソフトウェア構造には、適切な冗長性および多様性をも含めて、フォールト・トレランス・ソフトウェア設計戦略とフォールト・アボイダンス・ソフトウェア設計戦略の両方が含まれていること。

10.2.11 ソフトウェア構造仕様書では、フォールト・トレランス・ソフトウェア設計戦略の選択とフォールト・アボイダンス・ソフトウェア設計戦略の選択とのバランスに関する正当事由が記述されていること。

10.2.12 ソフトウェア構造仕様書では、ソフトウェア要求仕様を要求されたインテグリティ水準で達成するのに各ソフトウェア・ライフサイクル・フェーズ中で必要とするこれらの技法／手段を特定すること。

10.2.13 ソフトウェア構造仕様書には、ソフトウェア要求仕様を要求されたインテグリティ水準で満足させる統合セットから選択された技法／手段に関する正当事由が記述されていること。

［注］ソフトウェア構造仕様書では、基本安全性戦略は当該ソフトウェア用に開発されるということである。利用されるライフサイクルに依存して、要求仕様作成フェーズよりも設計フェーズの一部として見なしてもよい。

１１．設計および開発

１１.１ 目　標

11.1.1 本条の目標は、ソフトウェア要求仕様書に基づいて、規定された安全性インテグリティ水準を保持するソフトウェアを創成することである。

11.1.2 副目標：設計：本条の副目標は、要求された安全性インテグリティ水準を達成し、かつ、解析容易な、検証容易な、そして保守容易なソフトウェアを設計し、開発することである。

11.1.3 副目標：ソフトウェア・エンジニアリング環境：本条の副目標は、要求された安全性インテグリティ水準を達成するために、ソフトウェアの全ライフサイクルを通じて、検証、妥当性検査、評価および保守の活動を支援する言語およびコンパイラを含めて適切なツール・セットを選定することである。

１１.２ 要　件

11.2.1 ソフトウェア要求仕様書およびソフトウェア構造仕様書は設計工程の開始よりも前に入手可能であること。

11.2.2 もしソフトウェア要求仕様書に「安全関連機能および非安全関連機能」、あるいは「異なるインテグリティ水準を有する複数個の安全関連機能」のいずれかが含まれていた場合には、ソフトウェア設計仕様書で、これらの機能の間での独立性をどのようにして達成するかを示すこと。

11.2.3 生成されたソフトウェアは規模および複雑さの点で最小であるべきである。

11.2.4 要求された安全性インテグリティ水準に応じて、選択された設計手法は下記の点を容易にする特性を有していること。

i)　複雑性を制御する抽象化、モジュラリティおよびその他の機構

ii) 下記の点に関する明確で厳密な表現

・機能（性）
・構成要素間での情報フロー
・情報に関連する順序およびタイミング
・同時実行性（Concurrency）
・データの構造および属性

iii) 人間側の理解容易性

iv) 検証および妥当性検査

11.2.5 選択された設計手法はソフトウェアの保守を容易にする特性を有していること。その様な特性には、モジュラリティ、情報隠蔽化、そして情報のカプセル化が含まれていること。

11.2.6 8.2.8項に従って、ソフトウェアの設計および開発は複数個のフェーズに分割し、下記に示す文書に記録すること。

フェーズ	文書
ソフトウェア設計フェーズ	ソフトウェア設計仕様書 ソフトウェア設計テスト仕様書
ソフトウェア・モジュール設計フェーズ	ソフトウェア・モジュール設計仕様書 ソフトウェア・モジュール・テスト仕様書
コード化フェーズ	ソース・コードおよび支援文書
ソフトウェア統合化	ソフトウェア統合化計画書 ソフトウェア統合化報告書

11.2.7 ソフトウェア設計には、制御フローおよびデータ移動に対する自己監視が含まれていること。故障が検出された時には、適切なアクションが取られること。

11.2.8 ソフトウェア設計はソフトウェアをモジュールに分解することに基づいていること。各モジュールごとにソフトウェア・モジュール設計仕様書およびソフトウェア・モジュール・テスト仕様書があること。

11.2.9 もし標準のあるいは既に開発済みのソフトウェアが当該設計の一部として利用される場合には、そのことが明確に特定され、文書化されていること。ソフトウェア要求仕様書を満たしているかの点に関し、そのソフトウェアの適切性について正当化し、その正当事由を記録した別の報告書を作成すること。適切性は同種のアプリケーションでの申し分のない運転の証拠に基づいているか、コード化フェーズで新規に開発されるソフトウェアが受けると期待されるのと同様な検証および妥当性検査手順を受けたかであること。

11.2.10 検証済みのソフトウェア・モジュールが存在する可能性がある場合には、検証済みのソフトウェア・モジュールを設計で利用すること。

11.2.11 各々のソフトウェア・モジュールは読み易く、理解容易で、そしてテスト容易であること。

11.2.12 要求された安全性インテグリティ水準を達成するために、ソフトウェアの全ライフサイクルを通じて、言語、コンパイラおよび構成管理ツールを含めて適切なツール・セットを選定すること。

11.2.13 適用可能な場合には、自動テスト・ツールおよび統合開発ツールを利用すること。

11.2.14 選定されたプログラミング言語は認知された国家／国際規格に対する検定に合格（Certificate of Validation）している、あるいは、その目的に適合（Fitness for purpose）していることを詳細に述べた評価報告書があるトランスレータ／コンパイラを保持していること。

11.2.15 選定されたプログラミング言語は完全に、そして曖昧さなく規定されているか、あるいは、曖昧さなく規定された機構に限定されていること。

11.2.16 選択された言語は下記の要件を満たしていること。

i) プログラミング言語は当該アプリケーションの特性を考慮して選定されること。

ii) 選択された言語はプログラミング誤りを特定するのを容易にする機構を含んでいること。

iii) 選択された言語は設計手法と調和する機構を含んでいること。

11.2.17 11.2.15項を満たすことができない場合には、任意の別の言語がその目的に適合しているかの点に関して正当化し、ソフトウェア構成仕様書にその正当事由を記録すること。

11.2.18 コーディング規格を開発し、すべてのソフトウェアの開発に利用すること。

11.2.19 当該ソフトウェアの開発で利用したコーディング規格は評価者によってレビューされていること。

11.2.20 コーディング規格では良いプログラミング慣行を規定し、不安全な言語機構を禁止し、そしてソース・コードの文書化に関する手順を述べること。最小限、下記に示す情報をソース・コード文書の中に含めること。

・作者名
・コードの内容説明（Description）
・入力および出力
・構成の履歴

11.2.21 ソフトウェア統合化計画を、要求仕様作成フェーズ、設計フェーズ、そして開発フェーズの開発作業と併行した作業で策定し、詳細化すること。

11.2.22 ソフトウェア統合化計画書には、下記の事項が文書化されていること。

i) ソフトウェアの各々の統合化水準グループへの分割
ii) テスト・ケースおよびテスト・データ
iii) 実行されるテストの型
iv) テスト環境、テスト・ツール、テスト構成、テスト・プログラム
v) テストの完了を判定するテスト評価基準

11.2.23 ソフトウェア統合化報告書はテストの結果と、ソフトウェア統合化計画書で規定した目標および評価基準が満たされたか否かを述べたものとして作成されること。もし故障が発見された場合には、その故障理由が記録されていること。

11.2.24 ソフトウェア統合化報告書は監査可能な様式であること。

11.2.25 ソフトウェア統合化の期間中でソフトウェアに対して行われたいかなる修正あるいは変更に関しても影響調査（Impact study）を実施して、影響を受けるすべてのモジュールを特定し、そして必要な再検証活動を実施すること。

11.2.26 テスト・ケースおよびそのテスト結果を記録すること。なお、後続の分析を容易にするために機械可読な形式が望ましい。

１２．検　証

１２．１ 目　標

12.1.1 本条の目標は、インテグリティ水準として要求される度合に応じて、所与のフェーズの成果物を、そのフェーズの入力として提供された成果物および規格と照らし合わせて、その正当性および一貫性を確かなものにするためにテストし、評価することである。

１２．２ 要　件

12.2.1 検証計画は、ソフトウェア・ライフサイクルの各フェーズ毎に、開発作業と併行した作業で確定され、適切で、適当な文書に詳細に記述されていること。

12.2.2 各フェーズ用の検証計画書には、当該フェーズに対する検証工程で利用されるすべての評価基準、技法およびツールについて文書化されていること。

12.2.3 各フェーズ用の検証計画書には、当該フェーズの入力として提供された成果物および規格と照らし合わせて、当該成果物の正当性および一貫性を確かなものにするために実行される諸活動について記述されていること。

12.2.4 検証計画は、下記に示す事項に向けられていること。

i)　検証戦略および検証技法の選定
ii)　ソフトウェア・テスト装置の選定および利用
iii)　検証活動の選定および文書化
iv)　検証装置から直接得られた、およびテストから得られた検証結果の評価
v)　信頼性要件の評価

12.2.5 各々の開発フェーズにおいて、機能要件、信頼性要件、性能要件、そして安全性要件が満たされたことを実証すること。

12.2.6 検証は、インテグリティ水準として要求される度合に応じて、独立部隊によって遂行されること。

12.2.7 各々の検証の結果は、監査可能な様式で保持されていること。

12.2.8 各々の検証活動の後で、検証報告書として、ソフトウェアが検証に合格したか否か、また故障が発見された場合にはその故障理由を述べたものが作成されること。検証報告は、下記に示す事項に向けられていること。

i) ソフトウェア要求仕様書、ソフトウェア設計仕様書、あるいはソフトウェア・モジュール設計仕様書に適合していない品目
ii) 設計規格に適合していない品目
iii) 品質保証手順に適合していない品目
iv) 当該問題に完全には適応していないモジュール、データ、構造およびアルゴリズム

12.2.9 8.2.8項に従って、以下に示す条を実行すること。

12.2.10 ソフトウェア要求仕様検証：ソフトウェア要求仕様書が確定された後で、次のフェーズ、即ち、ソフトウェア設計フェーズが開始する前に実行される検証は、下記に示す事項に向けられていること。

i) 安全性要件、機能要件、信頼性要件、性能要件、およびその他の要件を規定したＰＥＳの安全性要求仕様書と、安全性に関する品質計画書とから作成した諸要件を満たしているかの観点でのソフトウェア要求仕様書の十分性（Adequacy）
ii) ソフトウェア要求仕様書に対するテストとしてのソフトウェア要求仕様テスト仕様書の十分性

検証では、下記に示す文書間の非両立性（Uncompatibility）をチェックすること。

iii) ソフトウェア要求仕様書とＰＥＳの安全性要求仕様書
iv) ソフトウェア要求仕様書とソフトウェア要求仕様テスト仕様書

12.2.11 ソフトウェア設計検証：ソフトウェア設計仕様書が確定された後で、次のフェーズ、即ち、ソフトウェア・モジュール設計フェーズが開始する前に実行される検証は、下記に示す事項に向けられていること。

i) ソフトウェア要求仕様書およびソフトウェア構造仕様書を満たしているかの観点でのソフトウェア設計仕様書の十分性
ii) ソフトウェア設計仕様書に対するテスト・ケース・セットとしてのソフトウェア設計テスト仕様書の十分性
iii) ソフトウェア要求仕様書に対する一貫性および完備性の観点でのソフトウェア設計仕様書の十分性

検証では、下記に示す文書間の非両立性をチェックすること。

iv) ソフトウェア設計仕様書とソフトウェア要求仕様書
v) ソフトウェア設計仕様書とソフトウェア設計テスト仕様書
vi) ソフトウェア設計テスト仕様書とソフトウェア要求仕様テスト仕様書

12.2.12 ソフトウェア・モジュール検証：ソフトウェア・モジュール設計仕様書が確定された後で、コード化フェーズが開始する前に実行される検証は、下記に示す事項に向けられていること。

i) ソフトウェア設計仕様書を満たしているかの観点でのソフトウェア・モジュール設計仕様書の十分性
ii) ソフトウェア・モジュール設計仕様書に対するテスト・ケース・セットとしてのソフトウェア・モジュール・テスト仕様書の十分性
iii) ソフトウェア設計仕様書のソフトウェア・モジュールへの分解と、下記の事項に関するソフトウェア・モジュール設計仕様書

・要求された性能の実現可能性
・以後の検証でのテスト容易性
・開発チームおよび検証チームに対する可読容易性
・以後の発展を可能とする保守容易性

検証では、下記に示す文書間の非両立性をチェックすること。

iv) ソフトウェア・モジュール設計仕様書とソフトウェア設計仕様書
v) ソフトウェア・モジュール設計仕様書とソフトウェア・モジュール・テスト仕様書
vi) ソフトウェア・モジュール・テスト仕様書とソフトウェア設計テスト仕様書

12.2.13 コード検証：インテグリティ水準として要求される度合に応じて、ソース・コードは、ソフトウェア・モジュール設計仕様書、コーディング規格および品質保証手順に適合していることを確かなものにするために検証されること。

12.2.14 ソフトウェア・モジュール・テスト：各々のモジュールは、自モジュールに対してテストすべきことを規定したソフトウェア・モジュール・テスト仕様書を保持していること。これらのテストでは、各々のモジュールが意図した機能を実行し、意図しない機能を実行していないことを実証すること。

12.2.15 ソフトウェア統合化テスト：すべてのモジュールをソフトウェア・システムとして統合化する一環として、ソフトウェアは、ソフトウェア設計テスト仕様書に則ってテストされること。これらのテストでは、すべてのモジュールが意図した機能を実行し、意図しない機能を実行していない様に正しく相互対話していることを実証すること。

12.2.16 ソフトウェア要求仕様テスト：ソフトウェア・システムがソフトウェア設計テスト仕様書を満たした後で、ソフトウェア・システムは、ソフトウェア要求仕様テスト仕様書に則ってテストされること。これらのテストでは、ソフトウェア要求仕様書中の要件すべてが正しく実行され、ソフトウェア・システムが意図しない機能を実行していないことを実証すること。

12.2.17 テスト・ケースおよびそのテスト結果を記録すること。なお、後続の分析を容易にするために機械可読な形式が望ましい。

１３．ソフトウェア／ハードウェアの統合化

１３．１ 目　標

13.1.1　本条の目標は、ソフトウェアをターゲット環境に統合化することである。

13.1.2　本条の別の目標は、意図したインテグリティ水準の要件を満たす目的で、ソフトウェアとハードウェアとの間の両立性を確かなものにして、ソフトウェアとハードウェアとを結合することである。

１３．２ 要　件

13.2.1　ソフトウェア／ハードウェア統合化計画を、要求仕様作成フェーズ、設計フェーズ、そして開発フェーズの開発作業と併行した作業で策定し、詳細化すること。

13.2.2　ソフトウェア／ハードウェア統合化計画書には、下記の事項が文書化されていること。

i)　システムの各々の統合化水準グループへの分割
ii)　テスト・ケースおよびテスト・データ
iii)　実行されるテストの型
iv)　テスト・ツール、テスト支援ソフトウェア、テスト構成記述を含むテスト環境
v)　テストの完了を判定するテスト評価基準

13.2.3　ソフトウェア／ハードウェア統合化計画書では、開発者が自社の構内で遂行可能な諸活動と、ユーザ側の設置場所にアクセスする必要がある諸活動とが区別されていること。

13.2.4 ソフトウェア／ハードウェア統合化計画書では、下記の各々の活動が区別されていること。

i) ソフトウェア・システムのターゲット・ハードウェアへの併合化
ii) （すべての）PESの統合化
iii) （制御対象の）全プロセスと全PESとの統合化

［注］上記の項目ii)およびiii)は、国際規格65A(Secretariat)123で網羅されており、これに関連し、かつ完備であるために、項目i)がここに追加されている。

13.2.5 ソフトウェア／ハードウェア統合化計画書で特定されたツールおよび設備は、最も早い実施時期までに当然獲得されているべきである。

13.2.6 ソフトウェア／ハードウェア統合化の期間中で統合化済みシステムに対して行われたいかなる修正あるいは変更に関しても影響調査を実施して、影響を受けるすべてのモジュールを特定し、そして必要な再検証活動を実施すること。

13.2.7 テスト・ケースおよびそのテスト結果を記録すること。なお、後続の分析を容易にするために機械可読な形式が望ましい。

13.2.8 ソフトウェア／ハードウェア統合化報告書はテストの結果と、ソフトウェア／ハードウェア統合化計画書で規定した目標および評価基準が満たされたか否かを述べたものとして作成されること。もし故障が発見された場合には、その故障理由が記録されていること。

13.2.9 ソフトウェア／ハードウェア統合化報告書は監査可能な様式であること。

１４．ソフトウェア妥当性検査

１４．１ 目　標

14.1.1 本条の目標は、統合化済みシステムが意図したインテグリティ水準で、ソフトウェア要求仕様書に準拠していることを確かなものにするために、統合化済みシステムをテストすることである。

１４．２ 要　件

14.2.1 テストが妥当性検査の主要な活動であること。

14.2.2 シミュレーションおよびモデリングは、妥当性検査工程を補完するのに利用してもよい。

14.2.3 ソフトウェア妥当性検査計画が確定され、適切な文書に詳細に記述されていること。

14.2.4 ソフトウェア妥当性検査計画が開発、実行され、そして、その実行結果はインテグリティ水準として要求される度合に応じて、独立部隊によって評価されること。

14.2.5 そのインテグリティ水準で評価者の同意が要求されている場合には、ソフトウェア妥当性検査計画の適用範囲および内容に関して、評価者あるいは評価者の代理である部隊が同意していること。また、この同意は、テストの期間中に、評価者が同席するかに関する声明の発表であること。

14.2.6 ソフトウェア妥当性検査計画では、各々の要求された機能に関し、静的および動的なケースを含めて、下記に示す事項が含まれていること。

i)　要求された入力信号：その順序およびその値
ii)　予想される出力信号：その順序およびその値
iii)　受け入れ基準

14.2.7 ソフトウェア妥当性検査計画書には、選択された妥当性検査戦略を正当化する事由の要約が含まれていること。正当事由には、下記に示す考慮事項が含まれていること。

i) 要求されたインテグリティ水準
ii) マニュアル技法あるいは自動技法、またはその両方
iii) 静的技法あるいは動的技法、またはその両方
iv) 解析技法あるいは統計技法、またはその両方

14.2.8 妥当性検査で利用される装置は適切に校正されており、そして、利用されるいかなるツール（ハードウェアあるいはソフトウェア）も、その目的にふさわしいことが示されていること。

14.2.9 ソフトウェアは正常運転中に投入される入力信号、予想される出現頻度、およびシステム・アクションを必要とする望ましくない条件をシミュレートして実行されること。

14.2.10 妥当性検査の結果は、監査可能な様式で、ソフトウェア妥当性検査報告書に文書化されていること。

14.2.11 ソフトウェア妥当性検査報告書として、ソフトウェアが妥当性検査に合格したか否か、また故障が発見された場合にはその故障理由を述べたものが作成されること。

14.2.12 ソフトウェア妥当性検査報告書では、下記に示す事項のすべてが特定されていること。

i) 利用したソフトウェアおよびハードウェア
ii) 利用した装置
iii) 装置の校正
iv) 利用したモデル
v) 発見した相違（逸脱）（Discrepancy）
vi) 実行した訂正アクション

14.2.13 供給者および／または開発者は、システム開発者が国際規格65A(Secretariat)123の要件を満たすことを可能にするために、ソフトウェア妥当性検査報告書およびすべての当面関係のある文書を利用可能にしておくこと。

１５．評　価

１５．１目　標

15.1.1　本条の目標は、ソフトウェアが規定されたインテグリティ水準を保持し、意図したアプリケーションに適合するようにライフサイクル工程および成果物がなっているかを評価することである。

１５．２要　件

15.2.1　インテグリティ水準として要求される度合に応じて、ソフトウェアの評価は設計チームとは独立した適任の評価者によって遂行されること。

15.2.2　評価者は、ソフトウェアが意図した目的に適合しているか否かを評価すること。

15.2.3　ソフトウェアは、当該システムに関係する広範な安全性問題に関して評価されること。

15.2.4　評価者は、評価の結果を詳細に述べた報告書を作成すること。

15.2.5　ソフトウェア評価報告書には、ソフトウェアによって達成されたインテグリティ水準に関する陳述と、ソフトウェアが意図したアプリケーションに適合しているか否かに関する評価者の意見（Opinion）が含まれていること。

15.2.6　ソフトウェアが意図した目的に適合していない、あるいは、ソフトウェアが要求されたインテグリティ水準を達成していなかった場合には、評価者は、ソフトウェア評価報告書で故障原因および必要とする訂正アクションについての意見を表明すること。

15.2.7　そのインテグリティ水準で評価者の同意が要求されている場合には、評価者あるいは評価者の代理人がｿﾌﾄｳｪｱ妥当性検査計画の適用範囲および内容に関して同意していること。また、この同意は、テストの期間中に、評価者が同席するかに関する声明の発表であること（14.2.5項を参照）。

１６．品質保証

１６．１ 目　標

16.1.1 本条の目標は、ソフトウェアが要求されたインテグリティ水準を達成することを確かなものにするのに必要な技術的および管理的な諸活動すべてを特定し、監視し、そして管理することである。

16.1.2 本条の別の目標は、そのインテグリティ水準で要求される上述の目標を遂行した証拠を提供することである。

１６．２ 要　件

16.2.1 供給者および／または開発者は、本国際規格の要件を支援する目的で、ISO 9000規格シリーズ、あるいは同等規格で規定したとうりの品質保証システムを少なくとも使用すること。

16.2.2 供給者および／または開発者は、これらの規格の関係する部分、即ち、ソフトウェア製品用のISO 9000-3規格あるいは同等規格を履行すること。

16.2.3 供給者および／または開発者は、本国際規格の16.2.1項の要件を実現する目的で、品質計画を用意し、文書化すること。

16.2.4 品質計画は、プロジェクト期間中ずっと更新されること。

16.2.5 あるフェーズ中で実行されるすべてのアクションは、そのフェーズを開始する前に、品質計画書で完全に規定されていること。

16.2.6 品質計画は、その履行に関係するすべての部隊でレビューされ、同意されていること。

16.2.7 下記の項目が品質計画書で規定されているか、あるいは引用されていること。

i) 下記の点を含んだ品質／安全性、要件／目標
  ・達成の手段
  ・達成度合を評価する手段
ii) 下記の点を含んだ各々のフェーズの定義
  ・開始基準および終了基準
  ・各々のフェーズの入力および出力
  ・各々のフェーズの主要な品質活動
  ・各々の活動に責任を有する組織単位
iii) システム統合化手順
iv) テストおよび妥当性検査計画
v) 品質に影響を与えると予想される特別な項目
vi) 保守計画
vii) 開発計画
viii) レビューおよび評価活動
ix) 利用されるコーディング規格
x) 以前の妥当性検査テストの評価
xi) 構成管理

16.2.8 構成活動には、開発の各々のフェーズおよびすべてのフェーズ中でのすべてのソフトウェア品目およびそれらの文書を特定・識別する手順が含まれていること。特に、各々の個々のソフトウェア品目はユニークな識別子を保持し、ソフトウェア品目の各々の版は識別されていること。

16.2.9 引き渡たされたソフトウェア製品に関しては、そのソフトウェア製品の追跡を容易にする手順があること。

16.2.10 各フェーズ用のソフトウェア検証計画の十分性および結果を精査すること。

１７．ソフトウェア保守

１７．１ 目　標

17.1.1 本条の目標は、要求されたインテグリティ水準を達成していることを確かなものにする目的で、ソフトウェアを訂正、拡張、あるいは適応化することである。

１７．２ 要　件

17.2.1 ソフトウェア保守用の手順に関する制御サイクルが確定され、適切な文書に記録されていること。これらの手順には、下記に示す事項が含まれていること。

i) 誤り報告、誤りログ、保守日誌、変更許可、ソフトウェア／システム構成
ii) テスト、検証、妥当性検査、評価

17.2.2 ソフトウェア保守記録簿は、各々の保守活動ごとに確定されていること。この記録簿には、下記に示す事項が含まれていること。

i) 修正あるいは変更要求
ii) ハードウェア、ソフトウェア、人間との間での相互対話と、環境との可能性のある相互対話とを含めて、全体のシステムに対する保守活動の影響分析
iii) 変更修正あるいは変更の詳細仕様
iv) インテグリティ水準として要求される度合に応じて、修正あるいは変更に対する再妥当性検査および回帰テスト

17.2.3 保守は、当初のシステム開発と同じ水準の専門知識、自動ツール、そして、計画および管理で実行されること。

17.2.4 保守性は、設計時に、特に、本国際規格の第１１条の要件に従って、ソフトウェア・システムに作り込まれていること。

17.2.5 ソフトウェア用の保守制御手順は、設計時に定式化し、インテグリティに対するいかなる負の影響をも避けるために使用されること。制御手順は、内部的あるいは外部的な1段以上の制御水準の階層構造で編成してもよい。手順は、厳密に規定され、定期および不定期に監査されること。

17.2.6 ソフトウェア保守日誌は、確定され、維持されること。

17.2.7 ソフトウェア保守日誌には、下記に示す事項が含まれていること。

i) すべての変更要求および承認
ii) 変更の進捗状況
iii) 変更の結末情報
iv) 再妥当性検査および回帰テスト用のデータを含めた各構成要素用のテスト・ケース
v) ソフトウェア構成の履歴
vi) 正常運転および正常条件からの逸脱

17.2.8 保守制御手順が品質計画書で正当化されていない場合には、図5に示した保守制御手順が使用され、関連する文書が作成されること。

17.2.9 供給者は、変則（Anomaly）分析を行って、変則報告書を作成すること。

１８．アプリケーション分野固有の規格を派生するための手引

１８．１ 目　標

18.1.1 本条の目標は、他の技術委員会およびワーキング・グループが本国際規格から産業固有の規格を派生するための手引を与えることである。

１８．２ 要　件

18.2.1 本国際規格から産業固有の規格を派生する際には、他の技術委員会およびワーキング・グループは本国際規格のある側面を変更してもよい。

18.2.2 産業固有の規格での要件は、本国際規格の各々の条の目標すべてに向けられていること。

18.2.3 18.2.1項が適用された場合、産業固有の規格には、下記に示す事項を詳細に規定した節を含んでいること。

i) 産業固有の規格中の条と本国際規格中の条との間の相互関係を示したマトリックス

ii) 本国際規格と派生された産業固有の規格との間の違い

iii) 本国際規格と派生された産業固有の規格との間の同等である正当事由

18.2.4 本国際規格中に含まれる表中の母集団（Population）は、特定の産業分野の要件に合わせるために修正してもよい。表中の母集団に関する正当事由は、アプリケーション分野固有の規格で与えられていること。

18.2.5 本国際規格では、技法とその正しい適用とを一括して使用することを要求している。これらの技法は、表で使用が要求され、その技法の詳細は主題目録にある。しかし、本国際規格で勧告した技法／手段を使用したからといって、要求されたインテグリティ水準が達成されることを保証（guarantee）してはいない。

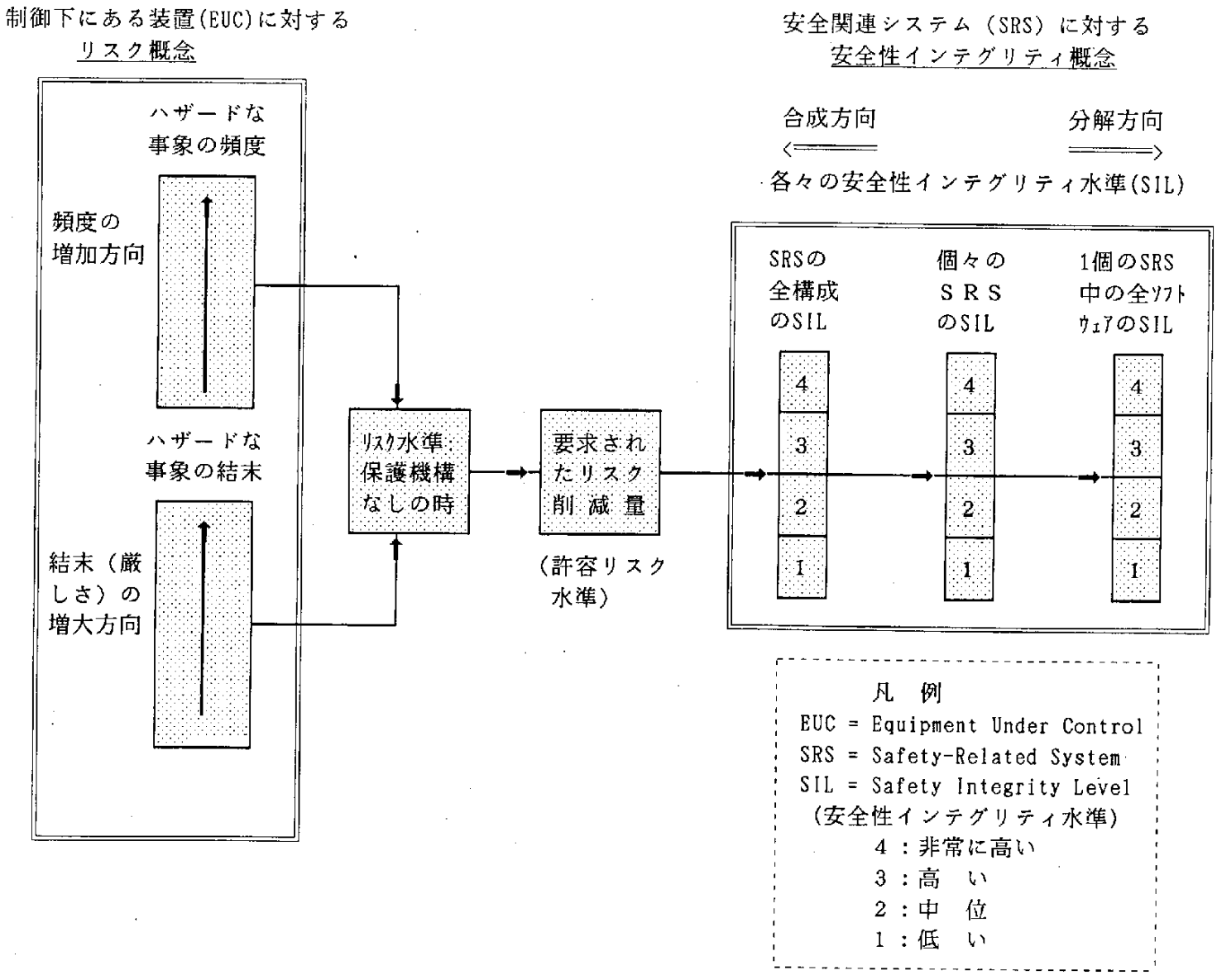


図1　安全関連システム（SRS）の安全性インテグリティ水準

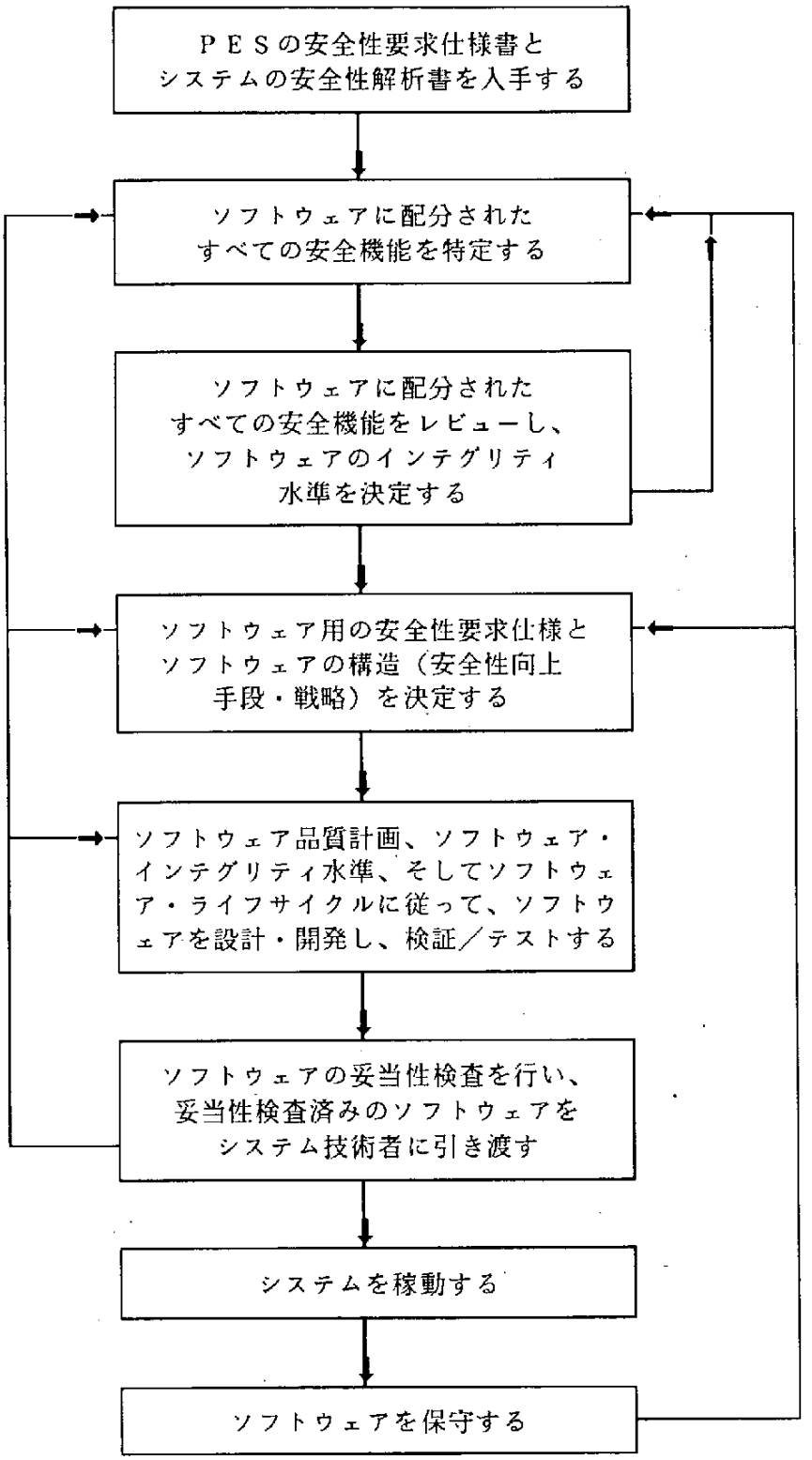


図２　ソフトウェア安全性の順路図

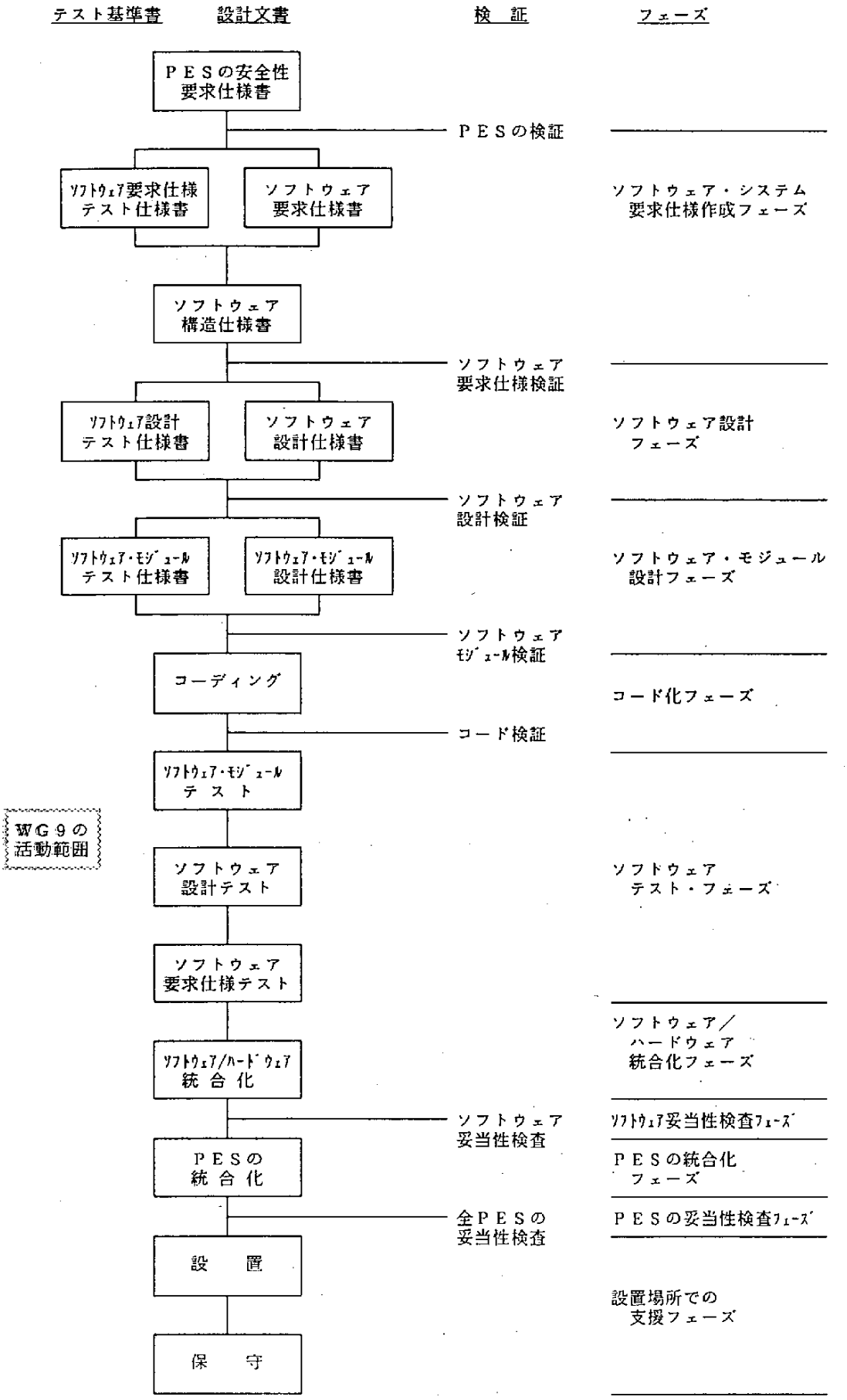


図３　　開発ライフサイクル　１

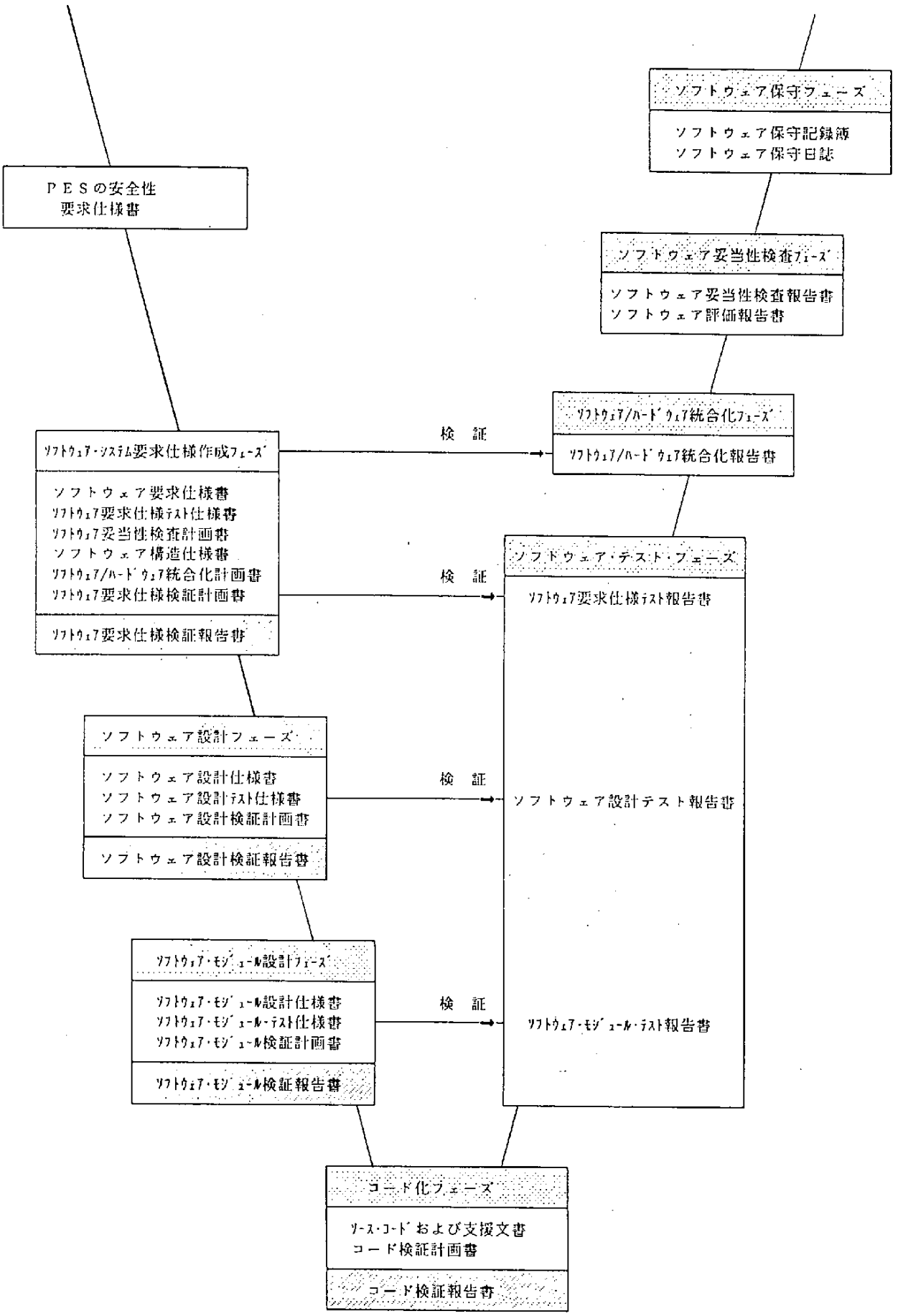
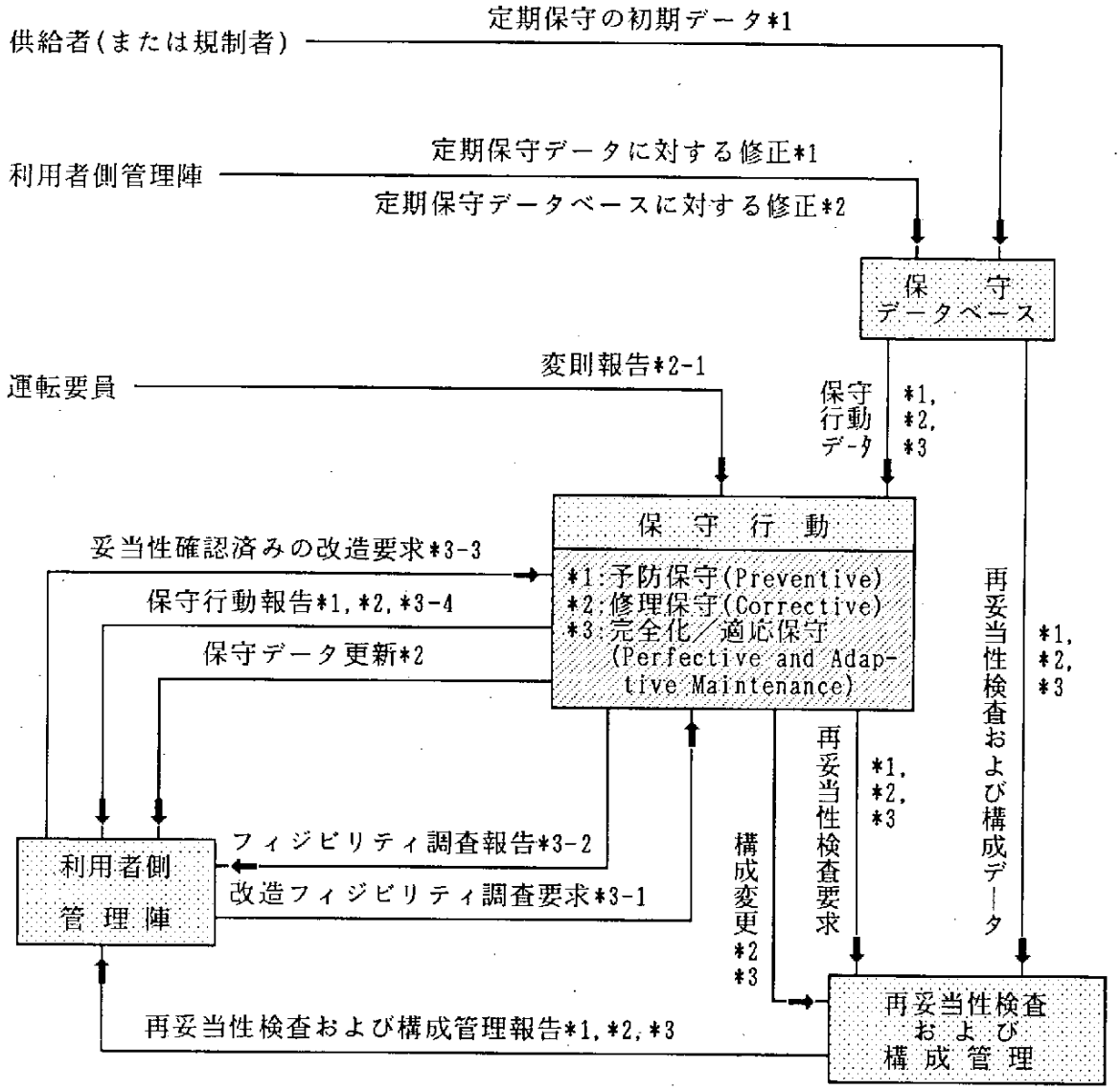


図４　開発ライフサイクル　２



[注] 修正：Modification　　改造：Modification　　変更：Change

図5　保守サイクル

技法および手段を選定するためのガイドライン

本国際規格の第６条から第１７条では、その適合性を達成する方法を図解した自条に関する表（Clause Table）を保持している。その表は、最も低水準の表と、当該条に関する表中の記入項目を展開した詳細表（Detailed Table）である。例えば、第６条に関する表中の記入項目“ハザード解析”は詳細表Ｄ．４で展開されている。

表には、表中の技法あるいは手段に対して、各々の安全性インテグリティ水準（IL: Integrity Level）１から４までを満たすのに、その技法あるいは手段を選定すべきかの勧告がある。これらの勧告の略語は下記の通りである。

“ＨＲ”　この略語は、当該技法あるいは手段に対して、この安全性インテグリティ水準を満たすために選定することを強く勧告する（Highly Recommended）ことを示している。もしこの技法あるいは手段が使用されなかった場合には、それを使用しない理論的根拠に関して品質計画書に詳細に記述し、評価者の同意を得ておくべきである。

“Ｒ”　この略語は、当該技法あるいは手段に対して、この安全性インテグリティ水準を満たすために選定することを、ＨＲ勧告よりも低い勧告で勧告し（Recommended）、かつ、その技法あるいは手段を技法群の一括を形成するのに組み合わせの構成要素として使用してもよいことを示している。

“－”　この略語は、当該技法あるいは手段に対して、その使用が勧告されないことを示している。

“ＮＲ”　この略語は、当該技法あるいは手段に対して、この安全性インテグリティ水準を満たすために選定しないことを確信を持って勧告する（Not Recommended）ことを示している。もしこの技法あるいは手段が使用された場合には、それを使用する理論的根拠に関して品質計画書に詳細に記述し、評価者の同意を当然得ておくべきである。

表に付記されている［注］が他の要件を満たすことの原因とならない限りは、技法あるいは手段の組み合せに関しては品質計画書で述べられていること。

条に関する表

＊技法あるいは手段に関する本条のインテグリティ水準での比較

第6条：ソフトウェア安全性インテグリティ水準

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. ハザード解析	D.4	－	R	R	HR	HR

第7条：要員とその責務

技法／手段	IL0	IL1	IL2	IL3	IL4
1. 資格、訓練	R	R	HR	HR	HR
2. 経　験	R	R	HR	HR	HR
3. 独立の評価者	－	R	R	R	HR

［注］1.すべての手段は利用されるライフサイクルに応じて当該インテグリティ水準を満たすために使用されること。

第８条：ライフサイクル問題と文書

技法／手段	IL0	IL1	IL2	IL3	IL4
1. ソフトウェア要求仕様書	HR	HR	HR	HR	HR
2. ソフトウェア設計仕様書	R	R	HR	HR	HR
3. ソフトウェア・モジュール設計仕様書	R	R	R	HR	HR
4. ソース・コードおよび支援文書	HR	HR	HR	HR	HR
5. ソフトウェア・テスト報告書	R	R	HR	HR	HR
6. ｿﾌﾄｳｪｱ/ﾊｰﾄﾞｳｪｱ統合化報告書	R	R	HR	HR	HR
7. ソフトウェア妥当性検査報告書	HR	HR	HR	HR	HR
8. ソフトウェア評価報告書	R	R	HR	HR	HR
9. ソフトウェア保守記録簿	HR	HR	HR	HR	HR

［注］1.すべての手段は利用されるライフサイクルに応じて当該インテグリティ水準を満たすために使用されること。

第９条：ソフトウェア要求仕様

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 形式的方法（例えば、CCS, CSP, HOL, LOTOS, OBJ, 時間論理 (Temporal Logic), VDM, Z)	B.30	－	－	R	R	HR
2. 準形式的方法	D.8	R	R	R	HR	HR
3. 構造化方法論（例えば、JSD, MASCOT, SADT, SSADM, Yourdon)	B.60	R	HR	HR	HR	HR

［注］1.ソフトウェア要求仕様書では、その問題を自然言語と、そのアプリケーションを表すのに必要なんらかの数学的記法とで記述したものが必ず要求されるはずである。

2.この表は、仕様を明確で、厳密に規定するための追加の要件を示している。要求されるインテグリティ水準を満足させるために、１個あるいはそれ以上の技法を選定すること。

第１０条：ソフトウェア構造

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 防衛化プログラミング	B.15	–	–	R	HR	HR
2. フォールト検出・診断	B.27	–	–	R	HR	HR
3. 誤り訂正符号	B.20	R	R	R	HR	HR
4. 故障ｱｻｰｼｮﾝ化ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ	B.25	–	R	R	R	HR
5. 安全バッグ技法	B.54	–	–	R	R	R
6. 多様化プログラミング	B.17	–	R	R	HR	HR
7. リカバリ・ブロック(RB)	B.50	–	R	R	R	R
8. 後退リカバリ	B.5	–	R	R	R	R
9. 前進リカバリ	B.32	–	R	R	R	R
10.再試行ﾌｫｰﾙﾄ･ﾘｶﾊﾞﾘ機構	B.53	R	R	R	R	HR
11.記録された実行済みケース	B.39	–	–	R	R	HR
12.優美劣化(Graceful Degradation)	B.33	R	R	R	HR	HR
13.人工知能 - フォールト訂正	B.1	–	–	NR	NR	NR
14.動的再構成	B.18	–	–	NR	NR	NR

［注］1.記入項目13および14を除いて、要求されるインテグリティ水準を満足させるために、１個あるいはそれ以上の技法を選定すること。

第１１条：設計および開発

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 形式的方法（例えば、CCS, CSP, HOL, LOTOS, OBJ, 時間論理 (Temporal Logic), VDM, Z）	B.30	－	－	R	R	HR
2. 準形式的方法	D.8	R	R	HR	HR	HR
3. 構造化方法論（例えば、JSD, MASCOT, SADT, SSADM, Yourdon）	B.60	R	HR	HR	HR	HR
4. モジュール・アプローチ	D.10	HR	HR	HR	HR	HR
5. 設計規格、コーディング規格	D.1	R	R	HR	HR	HR
6. 解析容易なプログラム	B.2	R	R	HR	HR	HR
7. 強い型付けをもつﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語	B.57	R	HR	HR	HR	HR
8. 構造化プログラミング	B.61	R	HR	HR	HR	HR
9. プログラミング言語	D.5	R	HR	HR	HR	HR
10. 言語サブセット	B.38	－	－	－	HR	HR
11. 検定済みツール	B.7	R	R	HR	HR	HR
12. 検定済みトランスレータ	B.7	R	R	HR	HR	HR
13. 枯れたトランスレータ	B.65	HR	HR	HR	HR	HR
14. 信用のおける／検証済みモジュールおよび構成要素のライブラリ	B.40	R	R	HR	HR	HR
15. 機能テスト、ﾌﾞﾗｯｸ・ﾎﾞｯｸｽ・ﾃｽﾄ	D.3	HR	HR	HR	HR	HR
16. 性能テスト	D.7	－	R	R	HR	HR
17. インタフェース・テスト	B.37	R	R	R	HR	HR
18. データ記録および解析	B.13	HR	HR	HR	HR	HR

［注］1. 様々な技法間の相互関係を十分に考慮して、要求されるインテグリティ水準を満足させるために、適切な技法のセットを選定すること。

第12条：検　証

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 形式的証明	B.31	－	－	R	R	HR
2. 確率論的テスト	B.47	－	－	R	R	HR
3. 静的解析	D.9	R	R	HR	HR	HR
4. 動的解析、動的テスト	D.2	R	R	HR	HR	HR
5. メトリクス	B.42	R	R	R	R	R

［注］1.要求されるインテグリティ水準を満足させるために、1個あるいはそれ以上の技法を選定すること。

独立性の度合	IL0	IL1	IL2	IL3	IL4
1. 独立会社	－	R	R	HR	HR
2. 独立部門	R	R	HR	HR	HR
3. 独立者	HR	HR	HR	R	R

第13条：ソフトウェア／ハードウェアの統合化

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 機能テスト、ﾌﾞﾗｯｸ・ﾎﾞｯｸｽ・ﾃｽﾄ	D.3	HR	HR	HR	HR	HR
2. 性能テスト	D.7	－	R	R	HR	HR

［注］1.要求されるインテグリティ水準を満足させるために、1個あるいはそれ以上の技法を選定すること。

第14条：ソフトウェア妥当性検査

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 確率論的テスト	B.47	－	－	R	R	HR
2. モデリング	D.6	－	R	R	HR	HR
3. 機能テスト、ﾌﾞﾗｯｸ・ﾎﾞｯｸｽ・ﾃｽﾄ	D.3	HR	HR	HR	HR	HR

［注］1.要求されるインテグリティ水準を満足させるために、1個あるいはそれ以上の技法を選定すること。

独立性の度合	IL0	IL1	IL2	IL3	IL4
1. 独立会社	－	R	HR	HR	HR
2. 独立部門	R	R	R	HR	HR
3. 独立者	R	R	R	NR	NR

第15条：評　価

評価対象の条	条番	IL0	IL1	IL2	IL3	IL4
1. ｿﾌﾄｳｪｱ安全性ｲﾝﾃｸﾞﾘﾃｨ水準	6	R	R	HR	HR	HR
2. 要員とその責務	7	－	R	HR	HR	HR
3. ライフサイクル問題と文書	8	－	R	HR	HR	HR
4. ソフトウェア要求仕様	9	－	HR	HR	HR	HR
5. ソフトウェア構造	10	－	R	HR	HR	HR
6. 設計および開発	11	－	R	HR	HR	HR
7. 検　証	12	－	R	HR	HR	HR
8. ｿﾌﾄｳｪｱ/ﾊｰﾄﾞｳｪｱの統合化	13	－	R	R	HR	HR
9. ソフトウェア妥当性検査	14	－	HR	HR	HR	HR
10.品質保証	16	－	HR	HR	HR	HR
11.ソフトウェア保守	17	－	HR	HR	HR	HR

評価技法	参照	IL0	IL1	IL2	IL3	IL4
1. チェックリスト	B.8	R	R	R	R	R
2. 決定表、真理値表	B.14	R	R	R	R	R
3. メトリクス	B.42	R	R	R	R	R
4. 原因-結果ダイアグラム(CCD)	B.6	R	R	R	R	R
5. 事象の木解析(ETA)	B.23	—	R	R	R	R
6. 故障の木解析(FTA)	B.28	R	R	R	HR	HR
7. FMECA	B.26	R	R	R	HR	HR
8. HAZOP	B.34	R	R	R	HR	HR
9. 共通原因故障解析(CCFA)	B.10	—	—	R	HR	HR
10.マルコフ・モデル	B.41	—	R	R	R	HR
11.信頼性ブロック図	B.51	—	R	R	R	R
12.モンテカルロ・シミュレーション	B.44	R	R	R	R	R

[注] 1.要求されるインテグリティ水準を満足させるために、1個あるいはそれ以上の技法を選定すること。

2.評価で要求される独立性の水準に関しては、ソフトウェア構造仕様書で規定される評価要件で変更してもよい。そのような評価要件はアプリケーション分野固有の規格で規定されるはずである。

独立性の度合	IL0	IL1	IL2	IL3	IL4
1. 独立会社	—	R	HR	HR	HR
2. 独立部門	R	R	R	HR	HR
3. 独立者	R	R	R	NR	NR

第16条：品質保証

技法／手段	条番参照	IL0	IL1	IL2	IL3	IL4
1. ISO 9001 あるいは等価な規格に関する認定	16	R	R	R	R	R
2. 会社の品質システム	16	HR	HR	HR	HR	HR
3. ソフトウェア構成管理	B.56	HR	HR	HR	HR	HR

第17条：ソフトウェア保守

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 影響分析	B.35	R	R	HR	HR	HR
2. 変更済みモジュールの再検証	B.35	R	HR	HR	HR	HR
3. 影響を受けるﾓｼﾞｭｰﾙの再検証	B.35	−	R	HR	HR	HR
4. 全システムの再妥当性検査	B.35	−	−	R	HR	HR
5. ソフトウェア構成管理	B.56	HR	HR	HR	HR	HR
6. データ記録および解析	B.13	HR	HR	HR	HR	HR

［注］1.要求されるインテグリティ水準を満足させるために、1個あるいはそれ以上の技法を選定すること。

権限付与の度合	IL0	IL1	IL2	IL3	IL4
1. 独立会社	−	R	HR	HR	HR
2. 独立部門	R	R	R	HR	HR
3. 独立者	R	R	R	NR	NR

詳 細 表

D.1 設計規格およびコーディング規格

第１１条で参照

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. コーディング規格が存在	B.16	HR	HR	HR	HR	HR
2. コーディング文体ガイド	B.16	HR	HR	HR	HR	HR
3. 動的オブジェクトがない	B.16	−	R	HR	HR	HR
4. 動的変数がない	B.16	−	−	R	HR	HR
5. 割り込みの制限された使用	B.16	−	R	R	HR	HR
6. ポインタの制限された使用	B.16	−	−	R	HR	HR
7. 再帰の制限された使用	B.16	−	−	R	HR	HR
8. 無条件ジャンプがない	B.16	−	R	HR	HR	HR

［注］1.これらの技法に関する情報については、主題目録のB.16“設計規格およびコーディング規格”を参照のこと。

D.2 動的解析および動的テスト

第１２条で参照

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 限界値分析法のﾃｽﾄ･ｹｰｽでの実施	B.4	R	R	HR	HR	HR
2. 誤り推定法のﾃｽﾄ･ｹｰｽでの実施	B.21	R	R	R	R	R
3. 誤り蒔く法のﾃｽﾄ･ｹｰｽでの実施	B.22	−	−	R	R	R
4. 性能モデリング	B.45	−	R	R	R	HR
5. 同値分割法のﾃｽﾄ･ｹｰｽでの実施	B.19	R	R	R	R	HR
6. 構造に基づくテスト	B.58	R	R	R	HR	HR

［注］1.テスト・ケースの分析はサブシステム水準に対して行われ、仕様書か、仕様書およびコードかのいずれか一方または両方に基づいていること。

D.3 機能テストおよびブラック・ボックス・テスト

第11条、第13条、第14条で参照

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 原因-結果ダイアグラム(CCD)のテスト・ケースでの実施	B.6	NR	—	—	R	R
2. ﾌﾟﾛﾄﾀｲﾋﾟﾝｸﾞ/ｱﾆﾒｰｼｮﾝ	B.49	NR	—	—	R	R
3. 限界値分析法のﾃｽﾄ･ｹｰｽでの実施	B.4	R	R	HR	HR	HR
4. 同値分割法のﾃｽﾄ･ｹｰｽでの実施	B.19	R	R	HR	HR	HR
5. プロセス・シミュレーション	B.48	R	R	R	R	R

[注] 1. テスト・ケースの分析はソフトウェア・システム水準に対して行われ、仕様書のみに基づいていること。

2. プロセス・シミュレーションの完備性は、インテグリティ水準、複雑度、そしてアプリケーションの度合に依存するはずである。

D.4 ハザード解析

第6条で参照

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 原因-結果ダイアグラム(CCD)	B.6	R	R	R	R	R
2. 事象の木解析(ETA)	B.23	—	R	R	R	R
3. 故障の木解析(FTA)	B.28	R	R	R	HR	HR
4. FMECA	B.26	R	R	R	HR	HR
5. HAZOP	B.34	R	R	R	HR	HR
6. モンテカルロ・シミュレーション	B.44	R	R	R	R	R

[注] 1. 予備ハザード解析は、ソフトウェアを最も適切な安全性インテグリティ水準に類別するために、既に行われているべきである。

D.5 プログラミング言語

第１１条で参照

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. ＡＤＡ	B.62	R	ＨＲ	ＨＲ	R	R
2. MODULA-2	B.62	R	ＨＲ	ＨＲ	R	R
3. PASCAL	B.62	R	ＨＲ	ＨＲ	R	R
4. Fortran 77	B.62	R	R	R	R	R
5. “Ｃ”	B.62	R	－	－	ＮＲ	ＮＲ
6. PL/M	B.62	R	R	R	ＮＲ	ＮＲ
7. BASIC	B.62	R	－	ＮＲ	ＮＲ	ＮＲ
8. アセンブラ	B.62	R	R	R	－	－
9. ラダー・ダイアグラム	B.62	R	R	R	R	R
10.機能ブロック	B.62	R	R	R	R	R
11.ステートメント・リスト	B.62	R	R	R	R	R

［注］1.インテグリティ水準３および４で、言語１、２、３、および４のサブセットが使用される場合には、その勧告を“ＨＲ”に変更すること。

2.あるアプリケーションの場合、言語５、６、８、９、１０および１１が唯一の利用可能な言語かもしれない。インテグリティ水準３および４で、“ＨＲ”と勧告された言語が利用できない場合、“Ｒ”と勧告された言語を“ＨＲ”に引き上げるには、その言語のサブセットが当然あり、かつ、厳格なコーディング規格のセットが当然あるべきであることを強く勧告する。

3.プログラミング言語の適切性の評価に関する情報については、主題目録のB.62“適切なプログラミング言語”を参照のこと。

D.6 モデリング

第１４条で参照

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. データ・フロー・ダイアグラム	B.12	R	R	R	R	R
2. 有限状態マシン	B.29	－	－	HR	HR	HR
3. 形式的方法	B.30	－	－	R	R	HR
4. 性能モデリング	B.45	－	R	R	R	HR
5. 時間ペトリ・ネット	B.64	－	－	HR	HR	HR
6. ﾌﾟﾛﾄﾀｲﾋﾟﾝｸﾞ/ｱﾆﾒｰｼｮﾝ	B.49	R	R	R	R	R
7. (プログラム)構造図	B.59	R	R	R	R	HR

D.7 性能テスト

第１１条、第１３条で参照

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 過負荷テスト	B.3	－	R	R	HR	HR
2. 応答時間制約、メモリ制約	B.52	HR	HR	HR	HR	HR
3. 性能要件	B.46	HR	HR	HR	HR	HR

D.8 準形式的方法

第９条、第１１条で参照

技法／手段	条番参照	IL0	IL1	IL2	IL3	IL4
1. ﾛｼﾞｯｸ/機能ﾌﾞﾛｯｸ・ﾀﾞｲｱｸﾞﾗﾑ	3	R	R	R	HR	HR
2. シーケンス・ダイアグラム	3	R	R	R	HR	HR
3. データ・フロー・ダイアグラム	B.12	R	R	R	R	R
4. 有限状態マシン／状態遷移図	B.29	－	R	R	HR	HR
5. 時間ペトリ・ネット	B.64	－	R	R	HR	HR
6. 決定表、真理値表	B.14	R	R	R	HR	HR

[注] 1. ロジック／機能ブロック・ダイアグラムおよびシーケンス・ダイアグラムについては、文書65A(Secretariat)90: “プログラマブル・コントローラ、第3分冊 - プログラミング言語”で記述されている。

D.9 静的解析

第12条で参照

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. 限界値分析法	B.4	R	R	R	HR	HR
2. チェック・リスト	B.8	R	R	R	R	R
3. 制御フロー分析	B.9	R	R	HR	HR	HR
4. データ・フロー分析	B.11	R	R	HR	HR	HR
5. 誤り推定法	B.21	R	R	R	R	R
6. Faganインスペクション	B.24	—	—	R	R	HR
7. 想定外回路分析 (SCA: Sneak Circuit Analysis)	B.55	—	—	—	R	R
8. 記号実行	B.63	R	R	HR	HR	HR
9. ウォークスルー／設計レビュー	B.66	HR	HR	HR	HR	HR

D.10 モジュール・アプローチ

第11条で参照

技法／手段	参照	IL0	IL1	IL2	IL3	IL4
1. モジュール・サイズの制限	B.43	HR	HR	HR	HR	HR
2. 情報隠蔽化／カプセル化	B.36	R	R	HR	HR	HR
3. パラメータ個数の制限	B.43	R	R	R	R	R
4. サブルーチンおよび関数では 1入口／1出口	B.43	R	HR	HR	HR	HR
5.完全に規定されたインタフェース	B.43	HR	HR	HR	HR	HR

[注] 1. これらの技法に関する情報については、主題目録のB.43“モジュール・アプローチ”およびB.36“情報隠蔽化／カプセル化”を参照のこと。

Ⅱ. 規格（案）の解説

1. 序　文

[1.0.1] 本国際規格は、ＩＥＣ国際規格65A(Secretariat)123：“Functional Safety of Programmable Electronic Systems - Generic Aspects”（プログラマブル電子システムの機能別安全性 - 汎用的な側面）と一対となった規格である。本国際規格は“ソフトウェア”規格と、一方、ＩＥＣ国際規格65A(Secretariat)123は“システム”規格と略称される（以後では、この名称を必要に応じて使用する。）。ソフトウェア規格を適用する前に、システム規格を熟知しておく必要がある。

システム規格の目次構成を次に示しておく。

序　文
1. 適用範囲
2. 引用規格
3. 定義と用語の説明
4. 考慮対象のシステム
　4.1 全　般
　4.2 プログラマブル電子システム（ＰＥＳ）
　4.3 システムの構成
　4.4 安全関連システム
　4.5 システム構成の分類
5. 安全性ライフサイクル
　5.1 全　般
　5.2 ハザード解析およびリスク評価
　5.3 安全性要求仕様
　5.4 安全関連システムの特定・指名
　5.5 設計および構築
　5.6 安全性の妥当性検査
　5.7 運転および保守
　5.8 システムの改造
　5.9 退　役

5.10 後付け

6. 一般安全原則

6.1 全　般

6.2 システム安全原則

7. 評価の普遍的な枠組み

7.1 全　般

7.2 主要なステップ

8. 要員の能力

9. 安全性計画

付録A（規範文書）. リスクと安全性インテグリティ

付録B（規範文書）. 手引での根拠となる考慮事項

付録C（参考文書）. 組込み用アプリケーション・ソフトウェアを図解するための異なるＰＥＳの例

付録D（参考文書）. システム・ライフサイクル・モデル

付録E（参考文書）. 安全性ライフサイクル・モデルの諸側面を図解するための事例

付録F（参考文書）. （今後の事例のための予備）

付録G（参考文書）. 全般的な文書化要件

付録H（規範文書）. 本規格で使用されている略語

付録I（参考文書）. 参考文献

なお、現時点で入手している1989年10月発行版のシステム規格案はかなり不備である。

本国際規格の想定読者、即ち、利用者は誰なのか。本国際規格はソフトウェア工学の規格として制定されているので、規格中では明確に述べられていないが、主たる想定読者は、安全関連ソフトウェアの開発活動を計画し、その開発活動を監査する人達と思われる。即ち、開発側の設計・構築・評価者のための規格である。しかし、ソフトウェア・システムの調達者が本規格を活用することにより、恩恵を受ける。

調達者側の規格の例として英国防省（MOD: Ministry of Defence）のINT DEF STAN 00-55: “The Procurement of Safety Critical Software in Defence Equipment”（防衛装置で利用されるセーフティ・クリティカル・ソフトウェアの調達）を参考として示しておく。この規格の目次構成は次に示すようになってい

る。

まえがき
第1部　総　則
0．序　文
1．適用範囲
2．警　告
3．関連文書
4．定　義
5．競争、利用、改造および支援を可能とする要求事項
第2部　安全性管理
6．安全性に対する責任
7．MOD安全性保証担当局
8．ハザード解析および安全性リスク評価
9．入　札
10．支　援
11．品質保証
12．リスク分析
13．要員の質
14．設計チーム
15．V&Vチーム
16．独立安全性監査担当者
17．下請負契約協定
18．安全性計画
19．安全性レビュー
20．設計実施基準
21．安全性記録簿
22．文　書
23．納入物件に対する要求事項
24．構成管理
25．検定合格および役務への受け入れ
26．製　造
27．供　用
28．廃棄処分

第３部　ソフトウェア工学に関する実施基準

２９．仕様定義

３０．設　計

３１．コーディング

３２．形式的論証

３３．動的テスト

３４．既存ソフトウェアの利用

３５．妥当性検査

３６．ツールおよび支援ソフトウェア

付録Ａ．定　義

付録Ｂ．納入物件

付録Ｃ．構成システムに対する要求事項

付録Ｄ．セーフティ・クリティカル・ソフトウェアの検定事項

索　引

本国際規格での必要条件となっているソフトウェアの品質管理／品質保証規格であるISO 9000-3(1991)：“Quality Management and Quality Assurance Standards, Part 3 - Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenance of Software”（品質管理および品質保証の規格集、第３分冊 - ISO 9001のソフトウェアの開発、供給、保守への適用のための指針）（ここでは、この規格をソフトウェア品質保証規格と略称する）は２者間契約に基づいてソフトウェア製品を設計・製造する場合の、すべての品質システム要素に関する供給者に対する要求事項を規定している。

この規格の目次構成は次に示すようになっている。

序　文

１．適用範囲

２．引用規格

３．用語の意味

４．品質システム － フレームワーク

４．１　経営者の責任

４．２　品質システム

４．３　内部品質システム監査

４．４　是正処置

5．品質システム － ライフサイクルでの活動

5．1 一　般

5．2 契約レビュー

5．3 購入者要求仕様

5．4 開発計画立案

5．5 品質計画立案

5．6 設計と製造

5．7 テスト及び妥当性確認

5．8 検　収

5．9 複製、引渡し及び据付け

5．10 保　守

6．品質システム － 支援活動

6．1 構成管理

6．2 文書管理

6．3 品質記録

6．4 測　定

6．5 規則及び取決め

6．6 ツール及び技法

6．7 購　買

6．8 支給ソフトウェア製品

6．9 訓　練

附属書A（参考）ISO 9000-3とISO 9001との間の参照表

附属書B（参考）ISO 9001とISO 9000-3との間の参照表

システム規格とソフトウェア規格はともにプロセス規格である。規格の種類としては、次に示す3種の規格がある。

①製品規格

②プロセス規格

③サービス規格

製品規格は、目的との合致性を確立するために、製品または製品のグループが満たさなければならない要求事項について規定している。プロセス規格は、目的との合致性を確立するために、プロセスが満たさなければならない要求事項について規定している。そして、サービス規格は、目的との合致性を確立するために、

サービスが満たさなければならない要求事項について規定している。

一般に、製品規格は厳密に規定できるが、プロセス規格は方針を述べた文の集りとなり、厳密さに欠けるきらいがある。本国際規格もプロセス規格であるので、厳密さに欠けている。

システム規格およびソフトウェア規格は汎用規格、即ち、基本規格である。基本規格は、広範囲に渡る規定、または、ある特定の分野に対する全般的な規定からなる。基本規格は、直接適用する規格としての機能をもつこともあるし、また、他の規格の基盤としての機能をもつこともある。

[1.0.2] 本国際規格では、プログラマブル電子システム（PES）のソフトウェアに関する安全性インテグリティを達成するのに必要な要件を規定している。ソフトウェアの安全性インテグリティが、本国際規格での中心主題である。即ち、ソフトウェアの安全性インテグリティ水準が本規格の出発点となっている。

[1.0.3] 本国際規格は、アプリケーション・ソフトウェアだけでなく、オペレーティング・システムおよびファームウェアにも適用されることに注目する必要がある。

[1.0.4] 本国際規格は、ソフトウェアの仕様フォールトや設計フォールトが原因であるＰＥＳの危険な故障の発生確率をいかにして低くするかを規定している。

[1.0.5] 本国際規格は、安全性インテグリティの４つの水準に対して、使うべき技法／手段を規定している。これが本国際規格の最大の特徴である。

ＩＬの基本概念を鉄道分野に適用すれば、ＡＴＣ（Automatic Train Control；自動列車制御装置）とＡＴＳ（Automatic Train Stop；自動列車停止装置）がＩＬ４（最高水準）、ＣＴＣ（Centralized Traffic Control；列車集中制御装置）がＩＬ３、ＣＯＭＴＲＡＣ（Computer Aided Traffic Control System；列車運行管理システム）がＩＬ２、そして、各駅の旅客サービス・誘導システムがＩＬ１（最低水準）に該当すると思われる。ＩＬ０はＭＩＳ（Management Information System；経営情報システム）に代表されるビジネス・ソフトウェアが該当し、非安全関連ソフトウェアと称される水準で、ISO 9000-3[ISO 1990]あるいは同等規格の履行を勧告している。ＳＲＳの開発では、ISO 9000-3の適用は最低限と見な

している。

形式的方法に関しては、いくつかある有効な技法の一つと位置付けている。

[1.0.6] システム規格でソフトウェアに要求される安全機能およびその安全性インテグリティ水準を規定している。ソフトウェア規格では、ソフトウェア安全性インテグリティ水準が決定されているという前提で規定内容が決められている（[6.1.1]の解説を参照）。

[1.0.7] この規定に関しては、システム規格を参照すること（解説図1参照）。

i) ここでは、ハザード解析およびリスク評価が実施される。

iv) システム構造とは、事故の未然防止や事故が発生した際にその影響を軽減するためにどんな技法／手段（およびシステム安全原則）をとるかのことである。

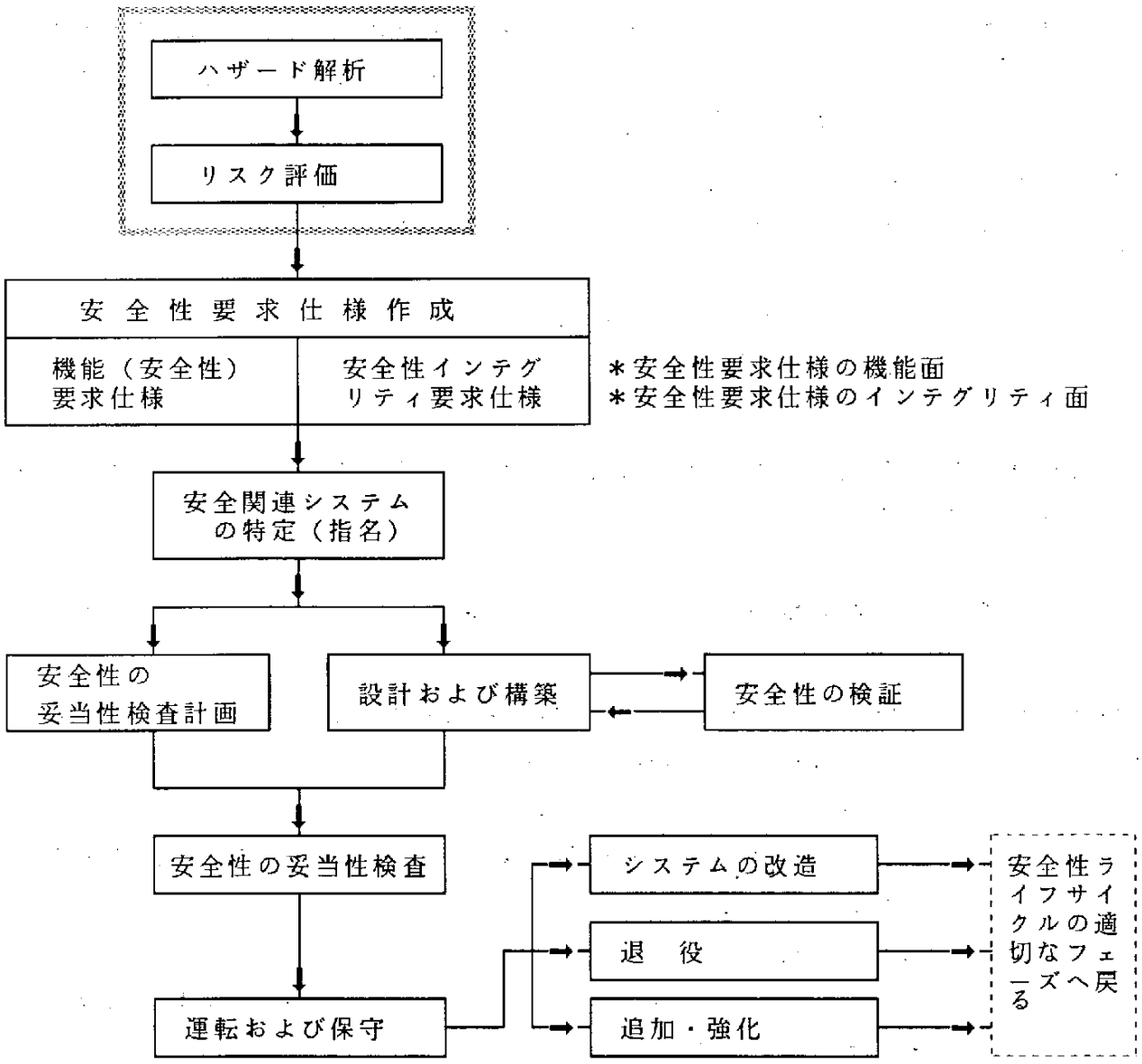
[1.0.8] リスクは、ハザードな事象の結末（即ち、厳しさ(Severity)）と頻度との両方からなる。

ここでは、構成要素という用語はプログラマブルではない（NP: Non-Programmable）（ＰＥＳではない）従来型システムを指していると思われる。ＮＰシステムは、例えば、電気機械装置、空気圧／水圧／油圧式装置などである。

●リスクと安全性インテグリティの関係は?

リスクの概念が（安全関連システムの）制御下にある装置（EUC: Equipment Under Control）に対して適用され、一方、安全性インテグリティの概念が安全関連システム（「安全関連システムの全構成」および「安全関連の個々のシステム」）に対して適用される（この２つの概念の分離が、システム規格およびソフトウェア規格でのキー・ポイントである。）（図1参照）。また、ソフトウェア・インテグリティの概念は安全関連の個々のシステム中に存在するソフトウェアに対して適用される。

許容リスク水準は、通常、問題としているアプリケーション領域での利害関係者によって設定される。即ち、それらは、社会的基準に基づいて決定される。安全性インテグリティは、安全関連システムを工学用語で規定することを可能とす



解説図1　システム／ソフトウェアの安全性ライフサイクルのモデル

る工学的概念である。安全性インテグリティ水準は、許容リスク水準を適切に満たす必要があるが、問題としているアプリケーション領域で独自に設定されることになる。しかし、実際問題として、許容リスク水準を達成するにはどの位の安全性インテグリティ水準が要求されるかが問題となろう。

システム規格および規格ソフトウェアでは、リスクと安全性インテグリティとが分離されて、安全性インテグリティのみを対象としている。

[1.0.9] 運用信頼度とは、使用または運用状態での品目の信頼度である。運用信頼度は一定ではなく、使用・保守の条件で代わってくる。

安全関連ソフトウェアでは、高信頼度は必要条件ではあるが、それだけでは十分条件を満足させない。

信頼性と安全性は別の概念であることをはっきり理解することが要諦である。

英語Ensureについて、ANSI/ASQC A3-1987から引用して、ここで正確に定義しておく。類似語として、AssureとInsureがある。

- Assure: To remove doubt, uncertainty, or worry.
  （疑念、不確実、心配の種を除くこと；ここでは“保証する”と訳すことにする。）
- Ensure: To make sure, certain, or safe.
  （大丈夫かどうか確認すること；ここでは“確かなものにする”と訳すことにする。）
- Insure: To make arrangements for indemnification.
  （保障・賠償の対策を取ること。）

[1.0.10]信頼性と安全性では目標が異なるので、信頼性要件と安全性要件を明確に分けて記述する必要がある。

関連する用語としてアベイラビリティ（Availability）がある。アベイラビリティとは、修理系が規定の時点で機能を維持している確率、あるいは、ある期間中に機能を維持する時間の割合である。修理系とは、運用開始後、保守によって故障の修復が可能な系である。

安全が第一であるシステム以外のシステムでは、一般にアベイラビリティの向上が最大の目標である。

[1.0.11]安全性要件では、システムがやっていけないこととか、そのような状況・状態になってはならないことをも規定すること。

[1.0.12]例えば、拳銃の場合、弾丸が決して発射されない拳銃が最も安全な拳銃ではあるが、それでは、その拳銃は全然役に立たない。これは、一種のパラドックスである。高信頼度の拳銃は誤って発射するおそれがより高いので、ある意味で不安全である。拳銃では、誤っての暴発を防止するために、一般にインタロックが採用されている。

[1.0.13]2台のコンピュータが常用冗長で、安全が第一であるシステムの場合、2台のコンピュータの内の1台のコンピュータが故障した時には、フェール・セーフ状態に移行し、直ちにコンピュータの稼動を停止する必要がある。

[1.0.14]ディペンダビリティという概念が要諦である。

[1.0.15]ソフトウェアを開発するのに利用するツール類（コンパイラ、静的解析ツールなど）が安全関連であると特定・識別された場合には、そのツール類にも本国際規格が適用される。

[1.0.16]この指摘が最も重要である。ではどうするか。フォールトがないことを完全に保証することはできないが、本規格で推奨している技法／手段を使用することによって、フォールトを少なくすることは可能である。即ち、システムに対する信頼を向上させることはできる。

英語Guaranteeについて、ここで正確に定義しておく。類似語として、Warrantyがある。GuaranteeとWarrantyは通常は同義語であるが、区別して使われる場合もある。

- Guarantee: 全体の性能に関する保証
- Warranty: 個々の機器・製品の保証

[1.0.17]高インテグリティ・ソフトウェアとは、次に示す1つ以上のクリティカル分野に属するソフトウェアである。

①セイフティ・クリティカル：その故障あるいは設計フォールトが人命に対するリスクの原因となる可能性のある機能を担うソフトウェアである。

②セキュリティ・クリティカル：その故障あるいは設計フォールトが権限のない人による情報の開示、変更、損失、あるいは破壊の原因となる可能性のある機能を担うソフトウェアである。

③性能クリティカル：その故障あるいは設計フォールトによって、システムが重大な目標を達成できなくなる原因となる可能性のある機能を担うソフトウェアである。

④コスト・クリティカル：その故障あるいは設計フォールトが多大な金銭的な損失をもたらす原因となる可能性のある機能を担うソフトウェアである。

⑤環境クリティカル：その故障あるいは設計フォールトが環境あるいは財産に多大な被害をもたらす原因となる可能性のある機能を担うソフトウェアである。

[1.0.18]本規格では、要求されたソフトウェア安全性インテグリティ水準ごとに、それに応じた（ふさわしい）諸原理・原則を適用することを要求している。

1.1 本国際規格の適用方法

[1.1.1] i)　●安全性要求仕様の内容について

安全性要求仕様は、次の2種類の仕様からなっている。

- 機能要求仕様（Functional Requirements Specification）
- 安全性インテグリティ要求仕様（Safety Integrity Requirements Specification）

機能要求仕様では、安全関連の制御システムが遂行しなければならない安全機能のすべてを規定する。例えば、熱シャットダウン用の機能別安全性要件（Functional safety requirement）は、下記の通りかもしれない。

if TEMP > HIGH SET **then** OPEN VENT and CLOSE FEED

しかし、機能要求仕様では、規定された安全機能が完全に果たされるかの確からしさの度合（Degree of certainty）に関する要件については一切規定しない。

一方、安全性インテグリティ要求仕様では、安全機能を確実に果たすのに必要な安全性インテグリティの水準を達成するための要件を確定する。

安全性インテグリティ要求仕様では、次に示す3つのシステム安全原則（System Safety Element）に関する要件を規定する。

- システム構成（System Configuration）
- 安全関連ハードウェアの信頼度（Safety-Related Hardware Reliability）
- 系統的インテグリティ（Systematic Integrity）

---

ii) 最も重要な作業はソフトウェアに要求されるインテグリティ水準を決定することである。

iii) 基本安全性戦略とは、事故の未然防止や事故が発生した際にその影響を軽減するためにどんな技法／手段をとるかのことである。

[1.1.2] 検証、評価、そして品質保証は付帯活動で、すべてのフェーズで実施する必要がある。

[1.1.3] 採用するソフトウェア開発ライフサイクルを決定し、それに必要な文書一式を決める必要がある。

[1.1.4] ソフトウェア開発に参画する要員の能力を要件として規定している。そのため、この要件に合致しないとソフトウェアの開発作業に従事できなくなる。

[1.1.5] 要求されたインテグリティ水準にふさわしい技法／手段が選定される必要がある。この技法／手段の選定が開発の成功のキー・ポイントである。

2. 適用範囲

2.1　本国際規格（ソフトウェア規格）は安全関連ソフトウェアのみに適用され、システム規格は安全関連システム全体に適用される。

2.2　“環境の保護”とは、製品、プロセス、およびサービスの影響および作用から生じる、容認できない損害から環境を守ることをいう。

“製品の保護”とは、使用、輸送、または、保管中の製品を、気候条件、または、その他の不利な条件から守ることをいう。

2.3　i)　本国際規格は、汎用基盤、即ち、基本規格である。

v)　ＰＥは特定のセンサ(Sensor)またはアクチュエータ(Actuator)のみだけに専用に利用されるＰＥＳの構成要素ではなく、ソフトウェアの制御下にあって異なる時には異なる機能を果たすＰＥＳの構成要素である。ＰＥの中にはＣＰＵが存在している。

2.4　（[1.1.2]の解説を参照）

2.5　（[1.1.3]の解説を参照）

3. 引用規格

ここで引用している規格以外に、第1条の解説で紹介した規格と、以下に示す規格をも参照すると有用である。

ANSI/IEEE-ANS-7-4.3.2-1982: Application Criteria for Programmable Digital Computer Systems in Safety Systems of Nuclear Power Generating Stations, American Nuclear Society, 1982.

Guideline for the Categorization of Software in Ontario Hydro's Nuclear Facilities with respect to Nuclear Safety, Revision 0, Nuclear Safety Department, Canada, 1991.

DLP880: (DRAFT)Proposed Standard for Software for Computers in the Safety Systems of Nuclear Power Stations(based IEC Standard 880), Ontario, Canada, 1991.

IEC/SC45A/WG-A3(Secretary)42: (DRAFT)Software for Computers Important to Safety for Nuclear Power Plant, 1991.

NPR-STD-6300: Management of Scientific, Engineering and Plant Software, Office of New Production Reactors, U.S. Department of Energy, 1991.

P1228: (DRAFT)Standard for Software Safety Plans, IEEE Working Group P1228, 1991.

RTCA/DO-178A: Software Considerations in Airborne Systems and Equipment Certification, Radio Technical Commission for Aeronautics, 1985.

Standard for Software Engineering of Safety Critical Software, Rev.0, Ontario Hydro, Canada, 1990.

4. 定　　義

以下では、安全関連の重要概念・用語を補完する意味で、本国際規格で定義されている用語を含めて、アルファベット順に定義・紹介する。なお、◎印は本国際規格でも定義されている用語である。

○事故（Accident）

(1) 死、損傷、環境破壊、または物質破壊の原因となる不測の出来事／事象または一連の出来事／事象[MOD 1989, MOD 1991]

(2) 結果として、死、損傷、健康・環境被害、または機器もしくは財産の損失をもたらす予定外の出来事／事象（event）または一連の出来事／事象[IEEE 1990]

○事故条件（Accident condition）

(1) 放射性物質の放出が適切な設計で定められた受け入れ可能な運転限界範囲内に保たれている状態からの逸脱（deviation）[IAEA 1988b]

○クリティカル属性（Critical attribute）

(1) その属性が制御不能になった場合に、システムの開発または機能に多大な影響を及ぼす属性[EWICS 1988]

○ハザード（Hazard）

(1) 事故に至る可能性のある条件[MOD 1989, MOD 1991]

(2) ある特定の環境条件が揃えば、事故に至る可能性のある条件（即ち、状態）の集り[Leveson 1990]

◎プログラマブル電子システム（PES: Programmable Electronic System）

(1) 制御、安全保護、または監視の目的で、センサ（sensor）および／またはアクチュエータ（actuator）が結合した1台またはそれ以上の中央処理装置（CPU: Central Processing Unit）に基づくシステム；用語ＰＥＳには、一体となった複数のシステム、マイクロプロセッサ、プログラマブル・コントローラ（PC: Programmable Controller）、プログラマブル・ロジック・コントローラ（PLC: Programmable Logic Controller）、そして、その他のコンピュータに基づく機器が含まれる[IEC 1989]。

○品質保証（QA: Quality Assurance）

(1) 製品またはサービスが所与の品質要件を満たしていることの妥当な信頼感を与えるために必要な計画的および体系的活動のすべて[ISO 1984, JIS Z 9900用語]

◎信頼性（Reliability）

(1) アイテムが与えられた条件で規定の期間中、要求された機能を果たすことのできる性質[JIS信頼性用語]

(2) [ソフトウェア信頼性] ソフトウェアが、規定された条件下で規定の期間中、システムの故障（Failure）を起こさない確率。この確率は、入力およびシステムの利用とからなる関数であると同時にソフトウェア内に存在する欠陥（フォールト; Fault）からなる関数でもある。システムに対する入力が欠陥に遭遇するか否かによって、その存在の有無が決まる[ANSI 1983]。

○信頼性プログラム（Reliability program）

(1) 信頼性目標値の設定およびそれを実現するための技術的、管理的な手順の計画体系[JIS信頼性用語]

◎リスク（Risk）

(1) 事故の厳しさとその尤度の尺度[MOD 1989, MOD 1991]

(2) システム・ハザードが事故の原因となる尤度（likelihood）と、その事故の厳しさ（severity）とを組み合わせた尺度[IEEE 1990]

(3) ①ハザードが発生する尤度、②そのハザードが事故に至る尤度、そして、③そのような事故に伴う最悪の可能性のある起こり得る損失とからなる関数[Leveson 1990]

◎安全性（Safety）

(1) 人間の死傷または資財の損失もしくは損傷を与えるような状態がないこと。信頼性では任務遂行のための機能上の故障を対象にするが、安全性では人間・資財に損失・損傷を与える危険な状態を対象とする[JIS信頼性用語]。

(2) 機器の損害もしくは破壊、または人間の損傷もしくは死の原因となる可能性のある諸条件が発生しないこと[MIL 1984]。

(3) はなはだしい放射性ハザードから現場要員、一般市民、そして環境を保護するための適切な運転条件の達成、事故の予防、または事故の結末の軽減

[IAEA 1988a]

(4) その故障によって発生する危険から生命および手足、環境そして財産を保護する、または危険そのものを除去すること[EWICS 1988]。

(5) 人命に危害を与える受け入れられないリスクがないこと[EOQC 1989]。

(6) 危害または損害のリスクが受け入れ可能な水準内に保たれている状態[ISO 1989]

○セーフティ・クリティカル機構（SCF: Safety Critical Features)

(1) システムに関連する受け入れられないハザードを除去したり、あるいは、これらのハザードに起因するリスクを軽減して、これらのハザードを受け入れられる水準にまでにするためにシステムが保持しなければならない機構[MOD 1989]

(2) 防衛装置から受け入れられない安全性リスクを除去し、防衛装置からその他のリスクを軽減して、これらのリスクを許容水準にまでにするために必要な機構[MOD 1991]

○セーフティ・クリティカル・ソフトウェア（SCS: Safety Critical Software)

(1) 安全性インテグリティとして最高水準が要求される機能あるは構成要素を実現するのに使用されるファームウェアを含むソフトウェア[MOD 1991]

(2) システム中で使用されているソフトウェアで、受け入れられないリスクの原因となる可能性のあるソフトウェア；ＳＣＳには、①そのソフトウェアの運転または運転故障がハザードな状態に至る可能性のあるソフトウェア、②ハザードな状態から回復することを目的とするソフトウェア、そして、③事故の厳しさを軽減することを目的とするソフトウェアが含まれる[IEEE 1990]。

○セーフティ・クリティカル・システム（SCS's: Safety Critical Systems)

(1) 人命または財産を危険にさらす可能性のある環境下で利用されるシステム[EWICS 1988]

◎安全性インテグリティ（SI: Safety Integrity)

(1) コンピュータだけでなく、システム全体に依存して、制御システムが安全に作動する（これにはフォールトが発生した時に安全に停止することが含まれる）能力[EIA 1990]

(2) セーフティ・クリティカル・システム、機能、あるいは構成要素が規定さ

れた利用方法の範囲内で規定されたすべての条件下で、要求されたセーフティ・クリティカル機構を達成する尤度[MOD 1991]

◎安全関連ソフトウェア（SRS: Safety-Related Software）

(0) セーフティ・クリティカル・ソフトウェア（SCS）と同義語または広義語

○安全関連システム（SRS's: Safety-Related Systems）

(0) セーフティ・クリティカル・システム（SCS's）よりも広義の用語

(1) 安全関連システムは、他のものの力を借りず、独力で、主として、①予期されたハザードな事象が発生するのを防ぎ、②もし仮に、（予期されない）ハザードな事象が発生した際には、その影響を軽減する目的で使用されるシステムである。そして、安全関連システムには、運転員など安全関連の要員もおそらく含まれるであろう[IEC 1989]。

(2) 原子力分野では、①は制御系、そして②は安全（保護）系に分類している。

○安全系（Safety system）

(1) ①原子炉の安全な運転停止を保証する、②炉心からの残留熱の除去を保証する、または③想定された運転状況の発生、即ち、事故条件発生の結末に限界を設けるという条件付きで、安全が最優先であるシステム[IAEA 1988 b]

○ソフトウェア安全性（Software safety）

(1) ソフトウェアがシステムの文脈（system context）下で実行された時に受け入れることのできないレベルのリスクを発生させることがないことを確かなものにすること。システム安全技術者は安全性問題をシステムとして解決することを強調するためにソフトウェア安全性よりもソフトウェア・システム安全性という用語を好んで使用している[Leveson 1991]。

(2) ソフトウェアの設計の際に、システムの安全性を向上させるために積極的な処置を講じ、そして、システムの安全性を低下させる可能性のある誤りに対して、その誤りによるリスクを許容可能な水準になるまで除去あるいは制御していることを保証し、検証する目的でソフトウェアに対してシステム安全性工学の技法を適用すること[AFISC 1985]。

○ソフトウェア安全性プログラム（Software safety program）

(1) システムの安全性を保証する系統的なアプローチ[IEEE 1990]

○ソフトウェア・システム安全性（SSS: Software Systems Safety）

(1) 「ソフトウェア・システムの設計・構築・使用・保守」および「セーフティ・クリティカルなハードウェア・システムとの統合」においてのシステムの安全性の最適化である[MIL 1984]。即ち、ＳＳＳの定義の意図する所は、ソフトウェアがトータル・システムの環境下で、受け入れることのできないレベルのリスクを発生、またはハザード状態の原因になることなしに、機能することを保証することである。

○システム安全性（System safety）

(1) 科学的、管理的、工学的な諸原則を適用して、システムのライフサイクルのすべてのフェーズにおいて、軍事作戦有効性（Operational effectiveness）、時間、そしてコストに関する諸制約条件を満足させて、システムの安全性を最適にすること[MIL 1984]。

○システム安全性プログラム（System safety program）

(1) システムのライフサイクルのすべてのフェーズにおいて、タイムリーでコスト有効度の高いやり方でシステム安全性要件を満足させ、軍事作戦有効性を向上させるために必要なシステム安全性面での管理的および工学的な作業および活動のすべて[MIL 1984]

以下は、本国際規格で定義されている用語の補足説明である。

◎誤り（Error）

フォールトは技術的に認識された状態(顕在化した状態)になると“誤り”と呼ばれる。この認識過程を誤り検出という。フォールトが誤りとして認識されるまでの期間を誤り潜在期間（Error Latency）という。

◎フォールト（Fault）の［注］

故障、誤り、およびフォールトの３者の階層関係は再帰的である。ある水準での故障は、その直上位水準においては根本原因のフォールトと見なされる。

◎安全性（Safety）の［注］

ランダム・ハードウェア故障の範疇での危険モードになる故障率の値は指数モデルに基づいて計算される。

故障には、危険でない故障(Nfundang)と危険な故障(Nfdang)がある。信頼性の故障割合（Reliability Failure Ratio）Rfでは両者を含む。ここで、Nは全構成品目数である。

$$Rf = \frac{Nfundang + Nfdang}{N}$$

安全性の故障割合（Safety Failure Ratio）Srは

$$Sr = \frac{Nfdang}{N}$$

である。

◎安全関連ソフトウェア（SRS: Safety-Related Software）

●ＳＲＳ（安全関連ソフトウェア）の故障について

故障とは、正常動作をしないことである。ＳＲＳの正常動作とは、制御対象のシステムが危険な故障を発生させた際、即ち、一般的にはシステムを安全側に停止（Fail-safe）させるべき時に確実に安全側に停止させ、システム停止をしなくてもよい時には確実に停止させないことである。ＳＲＳの場合には、正常動作率よりもむしろ故障の内容を分けて考察する必要がある。

①不安全故障（Unsafe failure）

不安全故障とは、ＳＲＳが制御対象システムに対する安全側停止ができない状態になってしまう故障をいう。この故障が発生している時に、制御対象システムに対して安全側に停止させる事態に至ると安全側停止の失敗を起こし、事態は深刻である。このような故障状態は点検によってしか発見できないであろう。

②安全故障（Safe failure）

安全故障とは、ＳＲＳが制御対象システムに対する安全側停止が必要でない時にシステムを安全側に停止させてしまう故障をいう。この故障が発生している時には、システム停止が現実に行われているので、この故障の発見はさほど困難ではないであろう。しかし、制御対象システムのアベイラビリティ（Availability）が低下し、再起動に長時間（例えば10時間）を要するシステムの場合は大きな損害を与える。

5. 目標および適合性

5.1　これ以降の各条は、次に示す形式からなっている。

X.1 目　標

X.2 要　件

5.2　そのＳＲＳに対して規定された当該インテグリティ水準に応じた水準で各条の要件を満足させて、その条の目標を満たすことによって、本国際規格に適合させることになる。

5.3　要求されるインテグリティ水準に応じた技法／手段を使用することによって、その水準での要件を満足させる。

5.4　技法／手段の選定には、本国際規格にある技法／手段に関する諸表を利用する。

5.5　表中でＨＲ（強く勧告）である技法／手段は必ず使用すること。その技法／手段を使用しない場合には、その理由を明確にすること。

5.6　表中にない技法／手段を使用する場合には、その技法／手段の有効性および適切性について評価すること。

5.7　本規格で要求した文書類を査読し、立会いテストを行って、各条の要件と表中の各々の技法／手段に対する準拠性を評価すること。

5.8　本国際規格で勧告した技法／手段を使用したからといって、要求されたインテグリティ水準が達成されることを、本国際規格では保証（guarantee）してはいない。本国際規格では、現状の技術水準を反映して、その技法／手段を使用することを勧告はしているが、実施の深さに関しては一切規定していない。本国際規格はプロセス規格であるので、実施の深さに関しては、リスクおよびインテグリティ水準を勘案して開発者自体（および調達者）で決める必要がある。

6. ソフトウェア安全性インテグリティ水準

6.1 目　標

6.1.1　ソフトウェアに対する要求されるインテグリティ水準をシステム規格で規定するのか（システム水準で決めるのか）、ソフトウェア規格（本国際規格）で規定するのかがはっきりしない（[1.0.6]を参照）。ここでは、ソフトウェアに対する要求されるインテグリティ水準はソフトウェア規格（本国際規格）で規定されると仮定する。即ち、トップダウンで言えば、ソフトウェア全体のインテグリティ水準はＰＥＳの安全性要求仕様書で与えられ、そのインテグリティ水準をソフトウェア・システムを構成する個々のソフトウェア・モジュールに配分する。逆にボトムアップで言えば、個々のソフトウェア・モジュールで達成されたインテグリティ水準を統合することによってソフトウェア全体に課せられたインテグリティ水準を達成する。

6.2 要　件

6.2.1　原則として、ＰＥＳの安全性要求仕様書が作成されていることが前提となっている（[1.1.1]の解説を参照）。

6.2.2　ここでは、ソフトウェア安全性インテグリティ水準は、ソフトウェアに関係するリスクの水準に基づいて決定すると規定している。

6.2.3　ここに列挙されている考慮すべきリスクは、ＥＵＣ（制御下にある装置）に関係するリスクであって、ソフトウェアに関係するリスクではないと思われる。ソフトウェア開発に関係するリスクを考慮する必要がある。

6.2.4　ソフトウェア安全性インテグリティは定量的尺度ではなく、定性的尺度で規定される。安全性インテグリティ水準０は非安全関連ソフトウェアである。

6.2.5　ソフトウェア安全性インテグリティ水準は、第９条のソフトウェア要求仕様書で規定される。

6.2.6　同等規格とは、しばしば、整合規格（Harmonized Standard）と同じ概念を表すものとして使用することがある。整合規格は、異なる標準化機関が承認した、同じ主題についての規格であって、製品、プロセス、およびサービスの互換性、または、これらの規格に従うことによって得られたテスト結果もしくは情報の相互理解を確保するものである。この定義では、整合規格は、①表現形式における差異があってもいいし、さらに、②例えば、説明的な注記、規格の要件をいかにして満たすかということに関する手引、代替案の優先順位、多様性のような内容における差異があってもよいことになっている。

7. 要員とその責務

7.1 目　標

7.1.1　ＳＲＳの開発作業に従事する人の能力を確かなものにする（ensure）ことを目標とし設定している。

Responsibilityは実施責任をいい、Accountabilityは結果責任をいう。ここでは、結果責任を問うていないようである。

7.2 要　件

7.2.1　ＳＲＳの開発作業に従事する人には一定の資格が要求されている。要求される一定の資格については、特定の産業ごとに決めることを要求している。同時に開発従事者に対する必要な厳格な訓練も要求している。

7.2.2　要求されている安全性インテグリティ水準に応じた能力および資格が要求されている。しかし、ＩＬ１では具体的にどんな能力および資格が要求され、そして、ＩＬ４ではどんな能力および資格が要求されるかを決めることが今後の課題である。

7.2.3　英国での安全関連システム技術者の教育カリキュラム案を次に示しておく。

モジュールＡ：概　論
- 問題の特性
- 法制度および経済性に関する検討事項
- 用語の定義
- 典型的な安全関連システム・プロジェクト
- 検定規格
- システムに関する検討事項
- 問題点
- 問題解決の諸原則

モジュールＢ：ハザード解析およびリスク解析

Ｂ.１　ハザード解析

B．2　リスク解析と要求事項

モジュールC：技法および技術

C．1　全般的なソフトウェア／システム工学に関する検討事項

C．2　構造（アーキテクチャ）に関する検討事項

C．3　基礎離散数学

C．4　形式的仕様

C．5　形式的アプローチ：諸原則およびプログラム検証

C．6　テスト

C．7　妥当性検査

C．8　フォールト・トレランス

C．9　心理学 - HCI

C．10　プログラム作成

モジュールD：管　理

D．1　原理および技法

D．2　安全関連システムの管理

モジュールE：評　価

E．1　基礎確率論および基礎統計学

E．2　ソフトウェア信頼性

モジュールF：安全システムの構築

F．1　技法類の統合化

F．2　事　例

モジュールG：諸原則および諸問題 - 繰り返し

・問題点

・社会規範に関する検討事項

・事例研究

・将来動向

7．2．4　要求されるインテグリティ水準の度合に応じて、任命する評価者の独立性の度合（独立会社、独立部門、独立者）を決めること。

7．2．5　開発者は、評価者の評価作業に全面的に協力すること。

7．2．6　開発者は、評価者の評価作業に使用されるプロジェクトに関係する文書類を整備しておくこと。

8. ライフサイクル問題と文書

8.1 目　標

8.1.1　ソフトウェアの開発を体系化することを要求している。

8.1.2　第三者がレビューおよび監査ができるように、ソフトウェアのライフサイクルでのすべての活動に関して文書に記録することを要求している。。

8.2 要　件

8.2.1　本国際規格では特定の開発ライフサイクル・モデルを採用することを示唆していないことに注目する必要がある。開発対象のアプリケーションの特質に合った開発ライフサイクル・モデルを選定する必要がある。

8.2.2　品質保証手順は、すべてのライフサイクルで実行される。

8.2.3　フェーズとは、明確に定義できる作業の一部分である。フェーズという用語はある特定のライフサイクル・モデルを暗に示すものではなく、また、ソフトウェア製品の開発におけるある期間を暗に指すものではない[ISO 9000-3用語]。フェーズという用語では、時間的順序は問題にしておらず、その作業がどこかの時点で実施されることを意味している。

8.2.4　各フェーズの最後では、必ず検証活動が実施されること。

8.2.5　すべての文書は、変更容易なように作成されていること。

8.2.6　各文書には、識別番号が付けられていること。

8.2.7　文書の記録形式に関しては特に規定してないが、変更を容易にし、かつ関連する文書間での一貫性を維持することを可能にするためと、構成管理を確実に行うために、電子化すること。

8.2.8　図3および図4で示されている開発ライフサイクルは典型的なウォータフォール・モデルである。この他にプロトタイピングを主たる手段とす

るライフサイクル・モデルとか、スパイラル・モデルなどがある。

ソフトウェア開発を成功させる上で、ソフトウェア要求仕様書が最も重要な文書であるので、以下にその目次構成の例を示しておく。

(1) 安全関連ソフトウェア用のソフトウェア要求仕様書の目次構成の例

1. 全　般
   1.1 ソフトウェアの目的
   1.2 ソフトウェアの機能
   1.3 他のソフトウェア・システムとの整合性
   1.4 課せられた制約条件
      1.4.1 性能制約条件
      1.4.2 設計制約条件
      1.4.3 品質特性
      1.4.4 外部インタフェース要件
      1.4.5 その他の要件
   1.5 ソフトウェアの構造
   1.6 管理面の詳細要件
   1.7 文書面の詳細要件
   1.8 信頼性および安全性
2. 外部要件
   2.1 機能要件
   2.2 物理的環境
3. プロジェクト要件
   3.1 プロジェクトの目的および機能
   3.2 他のソフトウェア／システムとの整合性
   3.3 技術の選定
   3.4 主要な構成要素
   3.5 文書面の詳細要件
   3.6 安全性および信頼性
   3.7 維　持
4. ソフトウェア受け入れテスト仕様
5. 使用される定義
6. 引用資料

(2) クリティカル・ソフトウェア用のANSI/IEEEが制定したソフトウェア要求仕様書の規格であるANSI/IEEE Std 830-1984に準拠したソフトウェア要求仕様書の目次構成の例

1. 序　論
  1.1 本書の目的
  1.2 本書の範囲
  1.3 用語定義と略語
  1.4 関連文書
  1.5 概　説
    1.5.1 本書の構成
    1.5.2 本書の対象者
2. 一般的事項
  2.1 製品の物理的諸元
  2.2 製品の機能
  2.3 使用者の特性
  2.4 全般的な制約条件
  2.5 前提条件と依存関係
3. 個別要件
  3.1 機能要件
  3.2 外部インタフェース要件
    3.2.1 ヒューマン・インタフェース
    3.2.2 ハードウェア・インタフェース
    3.2.3 ソフトウェア・インタフェース
    3.2.4 通信インタフェース
    3.2.5 物理的環境インタフェース
  3.3 性能要件
    3.3.1 入出力の負荷要件
    3.3.2 データベースの負荷要件
    3.3.3 応答の時間要件
    3.3.4 資源の使用要件
  3.4 設計要件
    3.4.1 適用標準
    3.4.2 ハードウェア制約条件

3．5 品質要件

3.5.1 アベイラビリティ

3.5.2 信頼性

3.5.3 保守性

3.5.4 セキュリティ

3.5.5 使用容易性

3．6 その他の要件

3.6.1 データベース

3.6.2 運　用

3.6.3 システム・ジェネレーション

4． 関連支援情報

(3) ミッション・クリティカル・ソフトウェア用の米国防省のソフトウェア要求仕様書の規格であるDI-MCCR-80025A(1986)の目次構成の例

1. SCOPE

1.1 Identification

1.2 CSCI Overview

1.3 Document Overview

2. APPLICABLE DOCUMENTS

2.1 Government Documents

2.2 Non-Government Documents

3. ENGINEERING REQUIREMENTS

3.1 CSCI External Interface Requirements

3.2 CSCI Capability Requirements

3.2.X (Capability name and project-unique identifier)

3.3 CSCI Internal Interfaces

3.4 CSCI Data Element Requirements

3.5 Adaptation Requirements

3.5.1 Installation-dependent data

3.5.2 Operational parameters

3.6 Sizing and Timing Requirements

3.7 Safety Requirements

3.8 Security Requirements

3.9 Design Constraints

3.10 Software Quality Factors

3.11 Human Performance/Human Engineering Requirements

3.12 Requirements Traceability

4. QUALIFICATION REQUIREMENTS

4.1 Qualification Methods

4.2 Special Qualification Requirements

5. PREPARATION FOR DELIVERY

6. NOTES

7. APPENDIXES

8.2.9 各フェーズ毎に個別の検証計画書あるいは検証報告書を作成してもいいし、各フェーズとの対応が明確になっていれば1つにまとめた単一の計画書あるいは報告書で対処してもいいといっている。

9. ソフトウェア要求仕様

9.1 目　標

9.1.1　本条の目標は、ＰＥＳ（プログラマブル電子機器）に求められる安全性のインテグリティ水準を満たすために必要となるソフトウェア要求仕様の全体像を定めることである。

9.2 要　件

9.2.1　ソフトウェア要求仕様はＰＥＳに対する安全性要求とその品質保証から発生する要求などから決められることを示している。

9.2.2　ソフトウェア要求仕様書には、成果物として提出される必要のある項目を網羅する必要があること、及びそれらの項目には機能や信頼性、安全性が当然含まれることを示している

9.2.3　インテグリティ水準を満たすために必要な、ソフトヴェア要求仕様書の記述、構成について示している。

i)　記述内容は明確で、正確で、明白で、立証可能で、テスト可能で、保守可能であり、かつ技術的にも可能な要求であることが必要である。

ii)　構成については、国際標準SC65A(Secretariat)123に規定されるＰＥＳ安全要求仕様書全体との関係が明確である必要性が示されている。

9.2.4　システム全体の設計、運用、保守に関係するすべての要員に理解可能なようにソフトウェア要求仕様書の表現や記述が工夫されなければならないことを示している。

9.2.5　ソフトウェア要求仕様書に必要な事項を下記に示している。

i)　安全機能とそれに必要なインテグリティ水準

ii)　メモリやディスク容量及び応答時間性能

iii)　ＰＥＳシステム-運用オペレータに関するマン-マシン・インタフェース

iv)　その他の機能及び特性

9.2.6　ソフトウェア要求仕様書には、制御対象の機器の内／外のインタフェース、あるいは直接に結合され／される可能性があるシステム間のインタフェースがすべて洗いだされ、記述されていなければならないことを示している。

9.2.7　対象となるＰＥＳのすべての想定される運用モードがソフトウェア要求仕様書に詳しく述べられていることが必要であることを示している。

9.2.8　対象となるＰＥＳのすべての動作モード、特に故障時に想定される動作モードは、ソフトウェア要求仕様書に詳しく述べられていることが必要であることを示している。

9.2.9　ハードウェア及びソフトウェアに関する制約条件はすべてソフトウェア要求仕様書に記述されていなければならないことを示している。

9.2.10　ＰＥＳの動作をモニタする方法にはソフトウェア的な手法とハードウェアによるものがあるが、ソフトウェア要求仕様書にはこれらに対する要求が記述されていることが必要であることを示している。

9.2.11　安全機能を維持するために実施する定期テストの実施内容、実施時期などの要求事項がソフトウェア要求仕様書に記述されていることが必要であることを示している。

9.2.12　システム運用中でも実施可能なテスト項目を洗いだし、要求事項としてソフトウェア要求仕様書に記述することが必要であることを示している。

9.2.13　ＰＥＳのインテグリティ水準の要求を満たすのに必要な機能がソフトウェアとして必要な場合には、これらの機能はソフトウェア要求仕様書の中に明示されていることが必要であることを示している。

9.2.14　ＰＥＳの非安全関連機能がソフトウェアに含まれている場合、これらの

機能はソフトウェア要求仕様書の中で非安全関連機能として明示されていることが必要であることを示している。

9.2.15 ソフトウェア要求仕様テスト仕様書がソフトウェア要求仕様書に基づいて作成され、ソフトウェア要求仕様の作成者と顧客の間で合意されることが必要であることを示している。

１０．ソフトウェア構造

１０．１ 目　標

10.1.1 本条の目標は、安全性インテグリティ水準の確保の観点から、ソフトウェア要求仕様書の要求事項を達成するために用い得るソフトウェア構成を選定し、推薦することにある。

10.1.2 本条のもう一つの目標として、ＰＥＳ構成によってソフトウェア上に発生する要求事項を整理することを挙げている。

10.1.3 本条のもう一つの目標として、安全性のためのハードウェア-ソフトウェア間の相互関係に関する重要事項の特定化と評価を挙げている。

１０．２ 要　件

10.2.1 ソフトウェア構成の提案はソフトウェア供給者によってなされ、ソフトウェア構成仕様書に詳述されることを求めている。

10.2.2 ソフトウェア構成仕様書には、必要なインテグリティ水準を達成しつつソフトウェアの機能要求仕様を満たすような、ソフトウェアのフィージビリティ（実現可能性）についての記述が必要であることを示している。

10.2.3 10.2.2項を適用したことによってＰＥＳ要求仕様書に生ずるいかなる変更点も、ＰＥＳ開発者との間で合意し、ソフトウェア構成仕様書に明記されなければならないことを示している。

10.2.4 ソフトウェア構成仕様書には、ハードウェア-ソフトウェア間のあらゆる相互干渉に関係する事項が特定され、評価され、述べられていなければならないことを示している。

10.2.5 ソフトウェア構成仕様書においては、すべてのソフトウェア構成要素が特定されることが求められ、それぞれの構成要素について特定することが必要な項目を示している。

i)　各構成要素の新規性、既存性、あるいは知的財産性
ii)　各構成要素の検証性、検証条件
iii)　各構成要素と安全関連要素の兼ね合い
iv)　各構成要素に必要なソフトウェアのインテグリティ水準

10.2.6　ソフトウェアが安全関連機能と関連しない機能の双方を備えている場合で、両機能を完全に別々に／独立に扱えなければ、両機能とも安全関連ソフトウェアとして取り扱うことが示されている。

10.2.7　ソフトウェア構成を設計する場合に、開発アプリケーションにおける安全関連部分を最少化するよう設計しなければならないことが示されている。

10.2.8　開発ソフトウェアに含まれる機能の安全性インテグリティ水準が異なるような場合に、それらの安全関連機能相互間に充分な独立性が示されない限り、ソフトウェア全体が最も高度のインテグリティ水準に属するものとして取り扱われることが示されている。

従って、安全要求仕様のレベルが異なる機能はなるべく切り離し、クリーン・インタフェースを実現し、機能間の独立性を確保してソフトウェアの開発コストを下げることが必要となる。

10.2.9　ソフトウェア構成仕様書には、必要なインテグリティ水準を達成するのに必要となる、ソフトウェアの開発、組み込み、品質保証、検証、及び保守に関する方針が示されることの必要性が示されている。

10.2.10　ソフトウェア構成仕様書に、冗長性や多様性などの耐故障性及び故障予防に関連するソフトウェアの設計方針が含まれていることの必要性が示されている。

10.2.11　ソフトウェア構成仕様書に、ソフトウェア設計時における故障予防性と耐故障性の兼ね合いの図り方の方針が含まれていなければならないことを示している。

10.2.12 ソフトウェア構成仕様書に、所要のインテグリティ水準においてソフトウェア要求仕様書で定められたインテグリティ水準を達成するためソフトウェア・ライフサイクルの各フェーズで必要となるソフトウェア検証の技法／手段が明らかになっていなければならないことを示している。

10.2.13 ソフトウェア構成仕様書に、要求されているインテグリティ水準を満たしつつソフトウェア要求仕様を満足するために使えるソフトウェア検証技法／手段を明示することが求められている。

１１．設計および開発

１１．１ 目　標

11.1.1 設計工程が開始される前段で作成されるソフトウェア要求仕様に基づき、規定された安全性インテグリティ水準を満足できるソフトウェアを製作することを目標とし、設計・開発やツールについて規定する。

11.1.2 副目標：設計：本条の副目標として、要求されるインテグリティ水準の達成と、解析や検証、そして保守の容易なソフトウェアを設計し、開発することをあげている。

11.1.3 副目標：ソフトウェア・エンジニアリング環境：さらに本条の副目標として、要求される安全性インテグリティ水準を達成するため、ソフトウェアのライフサイクルの各段階における支援ツール（言語およびコンパイラも含む）について適切な選択が行われることを意図している。

１１．２ 要　件

11.2.1 ソフトウェアの設計工程以前にソフトウェア要求仕様やソフトウェア構造仕様が作成されていることを規定している。ソフトウェア要求仕様は、ＰＥＳの安全性要求仕様書から導かれる要件を満たすものとして作成される（詳細は第９条参照）。ソフトウェア構造仕様書はソフトウェア要求仕様書で規定された要件を達成するためのものとして作成される（詳細は第１０条参照）。

11.2.2 検証など作業の実施の度合いや、解析の深さを、安全性インテグリティ水準に応じて設定することが本基準の特徴の一つでもある。このことは、安全性インテグリティ水準の異なるソフトウェアが混在するような場合には、水準の低いソフトウェアの誤りにより高い水準のソフトウェアが影響を受ける懸念があることを意味している。従って、ソフトウェア要求仕様書に，安全関連機能とそうでない機能が含まれる場合，もしくは異なるインテグリティ水準を有する複数個の安全関連機能が混在することが記載されている場合には，これらの機能間の独立性をどの様にして達成するかが重要になる。本条では、このことについてソフトウェア設

計仕様書への記載を述べている。

11.2.3 生産されるソフトウェアは，サイズそして複雑度を最小にすべきとしている。この要件は潜在的なソフトウェア・フォールト発生の要因を少なくし、検証しやすいソフトウェア製作に効果があり重要な事項である。

11.2.4 要求される安全性インテグリテイ水準に応じて、選択される設計手法が具備すべき要件について定めている。内容は、複雑性を制御する抽象性やモジュラリテイなどの機構的要件から始り、機能的要件、人間の理解のし易さ、そして検証のし易さまでを求めている。

11.2.5 設計手法に要求される特性として保守性をあげ、モジュラリテイ、情報隠蔽そして情報のカプセル化が含まれことを求めている。モジュール化手法には、サイズの制限や、情報隠蔽・カプセル化、パラメータ個数制限、入力口と出力口の制限、インタフェースの明確な規定が必要である（詳細表１０参照）。情報隠蔽・カプセル化は、信頼性や保守性向上を目的とした技術で、ある定められた手続きに従わない限りデータのアクセスを不能とすることにより、データ破壊などの障害を防止する技法である。

11.2.6 ここでは、8.2.8項に掲げられている文書の中で、設計・開発段階で記録すべき文書を示している。

11.2.7 ソフトウェアに組み込まれる自己監視について、ソフトウェア設計の段階で含めることを述べている。ソフトウェアのオンライン稼働中の診断技術はソフトウェアの安全性保証に重要な役割を持つ。ソフトウェア・ライフサイクルの各フェーズにおけるＶ＆Ｖを重視している本規格においてこの項の持つ意味は大きい。また、自己監視により障害検出がなされた場合の対応についても規定している。

11.2.8 ソフトウェアの設計においては、モジュール化を必須の要件としており、各モジュールに設計仕様書とテスト仕様書を用意することとしている。この設計書の作成にあたっては、付録Ｂ.43に記載された推奨規則を満たすことが望まれる。

11.2.9 標準のもしくは既に開発済みのソフトウェアを一部に利用する際の要求事項について記している。内容は、要求されるソフトウェア・インテグリテイ水準に相応しい実績があるかどうかの識別を求めることと、もし不足の場合には、新規のものと同様に扱うということである。本項のねらいは、既存のソフトウェアの使用を制限するところにあるのではない（次項参照）。

11.2.10 既存の検証済みソフトウェア利用の推奨である。ソフトウェア・インテグリテイ水準の高いプログラムに要求される開発作業の量は大きなものがある。ソフトウェアの再利用はこの面でも積極的に行う意義がある。

11.2.11 個々のソフトウェア・モジュールに対し、読み易さ，理解の容易さ，テストの容易さを求めている。当然の事項でもあるが、上記のソフトウェア再利用と共に考えると意義のある事項である。

11.2.12 要求された安全性インテグリテイ水準を達成するための適切なツール・セット選定についての条項である。このツール・セットは国家もしくは国際規格による検定に合格しているか同等の水準にあることがまず必要で、ソフトウェアのライフサイクルの全フェーズに利用できることが理想である。しかし、まだその段階にはなく、わずかコンパイラ（トランスレータ）としてＡｄａやPascalがその方向を目指しているに過ぎない。

11.2.13 自動検査ツールや統合開発ツールが使えるなら使用すべきである.

11.2.14 11.2.12項に関する条件を述べたものである。付録Ｂ.７を参照のこと。

11.2.15 採用されるプログラミング言語の要件について、次の11.2.16項と共に規定している。

11.2.16 選択される言語に、３つの要件を挙げている。この要件からは、オールマイテイな言語の存在よりも、設計手法などと調和した言語の選択が妥当としているように思われる。“条に関する表の第１１条”では、インテグリティ・レベル４に対しプログラミング言語や検定済みトランスレータの利用を強く要求している。しかし、詳細表のＤ.７ではプログミング言語としてレベル４に対しハイリコメンドできるものはなし、として

いる。ＡｄａやPascalもFortran 77並みのリコメンドに留まっている。現段階で「これなら」というものはないということか。

11.2.17 代替言語を利用する際の条件を述べている（この場合には代替となる言語の使用についての事由をソフトウェア構造仕様書に詳述して残さねばならない）。本規格でも、産業部門の特化した応用には、理想的な上記要件を満足できないケースがあり得ることを認めていると思われる。

11.2.18 コーディング規格を定め、すべてのソフトウェア開発に用いることとしている。本規格が利用されるような分野では、インテグリティ水準０であっても、コーディング規格を持つこととしている.

11.2.19 上項に記したコーディング規格は評価者によってレビューされていること。

11.2.20 コーディング規格での条件について述べている。

11.2.21 本項以下25項までソフトウェア統合計画に関するものである。ソフトウェア統合化計画は、個々のモジュールを組み合わせ統合化するためのものであるが、本規格では本条以外に記載はなく、どの段階で作成されるものかは曖昧である（ソフトウェア／ハードウェア統合化計画書とは異なるものである）。ソフトウェア統合化計画の策定は、本来ソフトウェア・システム要求仕様書作成フェーズという初期の段階で行われ、その後の設計フェーズや開発フェーズにおける成果を統合化計画に反映させかつ詳細化するべきと考える。

11.2.22 本項ではソフトウェア統合化計画書に文書化され記載されるべき事項があげられている。本来この事項は、設計開発の段階ではなくもっと前段で採り上げられるべきであろう。

11.2.23 ソフトウェア統合化計画の報告書について述べている。

11.2.24 ソフトウェア統合化報告書の様式についての記載である。11.2.22項の内容に基づいた記載の際に留意すること。

11.2.25 ソフトウェア統合化の期間中に生ずる不具合や、改良事項による修正作業が、これまでの作業によって築き上げてきた安全性インテグリティ水準を崩壊させることがあってはならない。従って、修正作業や変更が及ぼす影響調査と再検証作業は極めて重要である。

11.2.26 テスト・ケースおよびそのテスト結果の記録は、できる限り機械可読な形式とする。

12. 検　証

12.1 目　標

12.1.1　本条の目標について述べている。インテグリテイ水準として要求される度合いに応じたテスト、評価を目指している。作業は、各フェーズの成果物に対し、そのフェーズに入る前に提供された各種文書および規格をより所とし、正当性や一貫性の有無のテスト、評価を行うことである。

12.2 要　件

12.2.1　検証計画がソフトウェアのライフサイクルの各フェーズにおいて開発と同時に確定されることと、適切かつ適当な文書に詳述されることを定めている。ここでいう検証は各フェーズで行われ、かつ各フェーズの作業は検証報告書をもって完了となる（8.2.4項参照）。

12.2.2　検証計画は検証計画書に記載される。その計画書には当該フェーズに対する検証工程で利用されるすべての評価基準、技法およびツールについて記載される。

12.2.3　また、検証計画書には、検証作業として行われる諸活動について記述する。

12.2.4　検証計画における各種事項の定めである。

12.2.5　上記事項による検証計画に基づき、個々の開発フェーズでは、機能要件、信頼性要件、性能要件、そして安全性要件が満たされたことを証明する。

12.2.6　検証作業は、インテグリテイ水準の度合いに応じ、独立部隊により遂行されることとしている。インテグリテイ水準4の場合、この独立部署として独立会社、独立部門が強く推奨されるものの、独立者による作業は推奨されないとされている。

12.2.7　個々の検証結果は、監査可能な様式で保持されること。

12.2.8 それぞれの検証活動の後の検証報告書について述べている。検証に合格しないものがあれば、その故障理由について述べることとなる。報告は次の事項についてなされる。

i) ソフトウェア要求仕様書、ソフトウェア設計仕様書、あるいはソフトウェア・モジュール設計仕様書に適合していない品目
ii) 設計規格に適合していない品目
iii) 品質保証手順に適合していない品目
iv) 当該問題に完全には適応していないモジュール、データ、構造およびアルゴリズム

12.2.9 ここでは、8.2.8項に従って、以下に示す条項を実行すること。10項から16項は、本項を受けて各フェーズにおける検証作業について述べている。

12.2.10 ソフトウェア要求仕様書の検証：ソフトウェア要求仕様書が制定され、ソフトウェア設計フェーズに入る前に、次の項目に向けた検証を行う（第9条参照のこと）。

i) 安全性要件、機能要件、信頼性要件、性能要件、およびその他の要件を規定したＰＥＳの安全性要求仕様書（IEC/65A 123に従ったもの）と、安全性に関する品質計画書とから作成した諸要件を満たしているかという観点で、ソフトウェア要求仕様書の十分性について検証する。
ii) ソフトウェア要求仕様書に対するテストとしてのソフトウェア要求仕様テスト仕様書の十分性。

検証では、上記文書間の非両立性についてもチェックすること。

12.2.11 ソフトウェア設計検証：ソフトウェア設計仕様書が確定された後、次のフェーズ（ソフトウェア・モジュール設計）が開始されるまでに行う検証作業について述べている。

i) ソフトウェア設計仕様書の十分性（ソフトウェア要求仕様書とソフトウェア構造仕様書を満たしているか。）
ii) ソフトウェア設計仕様書に対するテスト・ケース・セットとし

てソフトウェア・テスト仕様書が十分であるか。

iii) ソフトウェア設計仕様書が、ソフトウェア要求仕様書に対し一貫性、完備性の点で十分であるか。

さらに、検証では上記文書間の十分性と共に非両立性についてもチェックすること。

12.2.12 ソフトウェア・モジュール検証：ソフトウェア・モジュール設計仕様書が制定された後、次のフェーズ（コード化フェーズ）が開始されるまでに行う検証作業について述べている。

i) ソフトウェア・モジュール・テスト仕様書の十分性（ソフトウェア設計仕様書を満たしているか。）

ii) ソフトウェア・モジュール仕様書に対するテスト・ケース・セットとしてソフトウェア・モジュール・テスト仕様書が十分であるか。

iii) ソフトウェア設計仕様書のソフトウェア・モジュールへの分解と、下記の事項に対するソフトウェア・モジュール設計仕様書の記載について

- ・要求された性能の実現可能性について
- ・以後の検証でのテスト容易性
- ・開発チームおよび検証チームに対する可読容易性
- ・以後の発展を可能とする保守容易性

さらに、検証では下記文書間の十分性と共に非両立性についてもチェックすること。

iv) ソフトウェア・モジュール仕様書とソフトウェア設計仕様書

v) ソフトウェア・モジュール仕様書とソフトウェア・モジュール・テスト仕様書

vi) ソフトウェア・モジュール・テスト仕様書とソフトウェア設計テスト仕様書

12.2.13 コード検証：インテグリテイ水準として要求される度合いに応じて、ソース・コードは、ソフトウェア・モジュール設計仕様書、コーディング

規格および品質保証手順に適合していることを保証するために検証されること。

12.2.14 ソフトウェア・モジュール・テスト：各々のモジュールは、自モジュールに対してテストすべきことを規定したソフトウェア・モジュール・テスト仕様書を保持していること。これらのテストでは、各々のモジュールが意図した機能を実行し、意図しない機能を実行していないことを証明すること。

12.2.15 ソフトウェア統合化テスト：全モジュールをソフトウェア・システムとして統合化する一環として、ソフトウェアは、ソフトウェア設計テスト仕様書に則ってテストされること。これらのテストでは、すべてのモジュールが意図した機能を実行し、意図しない機能を実行していないように正しく相互対話していることを証明すること。

12.2.16 ソフトウェア要求仕様テスト：ソフトウェア・システムがソフトウェア設計仕様書を満たした後、ソフトウェア・システムは、ソフトウェア要求仕様テスト仕様書に則ってテストされること。これらのテストでは、ソフトウェア要求仕様書中の要件すべてが正しく実行され、ソフトウェア・システムが意図しない機能を実効していないことを証明すること。

12.2.17 テスト・ケースおよびそのテスト結果を記録すること。なお、後続の分析を容易とするために機械可読な形式が望ましい。

１３．ソフトウェア／ハードウェアの統合化

１３．１ 目　標

13.1.1　ＰＥＳにおいては、ソフトウェア／ハードウェアの統合化というフェーズは不可欠である。とりわけＳＲＳをのせる安全関連システムにおいては、その要求されたインテグリティ水準を損なうことなく実施することが要求される。

13.1.2　（同　上）

１３．２ 要　件

13.2.1　ソフトウェア／ハードウェア統合化計画は要求仕様作成フェーズにて作成し、設計フェーズ、開発フェーズにて詳細化することを要求している。

13.2.2　ソフトウェア／ハードウェア統合化計画書に記載すべき事項

i)　システムを統合化する手順を考慮して分割し、その結果を示す。分割の内容は13.2.4に示されている。

ii)　テスト・ケースおよびテスト・データは要求仕様書にのみ基づき作成されること。

iii)　テスト要項

iv)　テストの前提となるテスト環境

v)　テスト（統合化）の完了を判定する評価基準

13.2.3　他ＰＥＳとの接続や制御対象プロセスとの確認などは、開発者の工場内では最終確認できないことが多く、そのため、最終確認を行う環境を規定しておくことが要求される（フィールドテスト）。

13.2.4　13.2.2項参照

13.2.5　使用するツールなどはテスト開始前に準備され、正当性が確認されていることが要求されている。

13.2.6　テストは故障を発見することが目的で実施される。故障に対する修正／変更作業後のテストにおいても同様な立場で、すべての影響範囲についてテストすることが要求されている。

13.2.7　テスト・ケースおよびテスト結果はとりわけ膨大な文章、データとなることから文書管理が重要となる。

13.2.8　ソフトウェア／ハードウェア統合化計画書の評価基準が満たされない場合は、統合化の作業は完了せず中断となる。修正変更作業の結果、影響範囲によっては統合化のステップをさかのぼって再度統合化の作業を実施することとなる。

１４．ソフトウェア妥当性検査

１４．１ 目　標

14.1.1 統合化済みシステムをテストすることでソフトウェアの妥当性を検査し、ソフトウェア要求仕様書に準拠していることを確かなものにすることを要求している。

本規格では文書類についての妥当性検査は「評価」として分離している。

１４．２ 要　件

14.2.1 ソフトウェアの品質保証を最終的にはＰＥＳをテストすることで実施することを求めている。

14.2.2 シミュレーションやモデリングによって実環境では生成し得ないテスト環境やテスト・ケースの生成を行うことを推奨している。

14.2.3 ソフトウェア妥当性検査計画はソフトウェア要求仕様書にのみ基づき要求仕様書作成フューズにて作成することが要求されている。

14.2.4 ソフトウェア妥当性検査の実行結果は、第三者（独立部隊）によって評価されることが要求されている。その独立の度合いについては、付表でインテグリティ水準別に示されている。

14.2.5 ソフトウェア妥当性検査計画の内容についても、インテグリティ水準により第三者の評価者による評価が必要なこと、テストへの立会いも必要となることが示されている。

14.2.6 （特になし）

14.2.7 妥当性検査に用いる技法およびツールの正当性を示すことが要求されている。

14.2.8 （同　上）

14.2.9 テストは通常予想される実環境をシミュレートして実行されると共に、ハードウェアの故障などの異常時についてもシミュレートして行うことが要求されている。

14.2.10 ソフトウェア妥当性検査報告はユーザの検収および評価が容易に行われるように文書化されることが要求されている。

14.2.11 妥当性検査にて故障が発見された場合は変更／追加作業が実施されるが、報告書にその故障内容、理由を明記することを求めている。

14.2.12 ソフウェア妥当性検査報告書には、検査を実施した環境、前提をすべて特定しておくことが要求されている。

14.2.13 （特になし）

１５．評　価

１５．１ 目　標

15.1.1 評価の対象は、ソフトウェア成果物そのものと、ライフサイクル工程である。また、評価内容には、要求機能の実現度合いだけではなく、当該ソフトウェアにあらかじめ設定されたインテグリティ水準を満たしているか、も含まれている。評価の時期については、明確には述べられていないが、各段階ごとに並行して実施されると考えるべきであろう。いずれにせよ、ソフトウェアの開発作業とは、かなり独立した作業であるから、開発工程と必ずしも密接にリンクしていなくともよいと考えられる。いわゆるシステム監査と同様な位置付けとなる。

１５．２ 要　件

15.2.1 評価者の、ソフトウェアの開発・保守を行う組織からの独立性の度合は、インテグリティ水準によって決定される（7.2.4項参照）。

独立性の度合は、「独立会社」、「独立部門」、「独立者」の３ケースに分類される（各定義は第４条参照）。

但し、第１５条に対する表の注記２にもあるように、独立性の水準（？）に関しては、各アプリケーション独自の規格などによって、変更可能である。

15.2.2 評価内容は、各アプリケーションの要求機能の実現性と、インテグリティ水準の保持である。

15.2.3 ソフトウェアは、その特定のアプリケーションの中だけではなく、そのソフトウェアが機能するシステム全体の安全性問題に関して評価する必要がある。

15.2.4 ライフサイクルのソフトウェア妥当性検査フェーズまでに報告書が作成される。

15.2.5　ソフトウェア評価報告書に記載すべき事項として、インテグリティ水準の達成度合がはっきりと記載しなければならないが、アプリケーションへの適合度については評価者の「意見」でも良い。

評価のポイントはインテグリティ水準の達成の確認にあると考えるべきである。

15.2.6　達成されていない事項の原因および解決策について、評価者はあくまで「意見」を報告するだけである。

15.2.7　14.2.5項とのからみで、どのインテグリティ水準から、ソフトウェア妥当性検査への評価者の同意が必要とすべきか？。

１６．品質保証

１６．１目　標

16.1.1　ソフトウェアの品質の保証とは、ソフトウェアが定められたインテグリティ水準を満たしていることを保証することである（？）。

このためには、供給者または（かつ）開発者は、インテグリティ水準を達成するために取られる技術的または管理上の手段が確実に履行されていることを監視・管理しなければならない。

16.1.2　また16.1.2項の監視・管理によって、必要な事項が履行されたことを証明する証拠を残すことも重要な目標である。

１５．２要　件

16.2.1　ISO 9000は、本国際規格の品質保証のベースとして位置付けられる（すべてのインテグリティレベルにおいて）。他の等価な規格（例えばISO 9000の内容を取り込んだ国内規格など）でも可としている。

16.2.2　準拠すべき規格は、明確にソフトウェア（あるいは同等な製品）を対象としているものでなければならない。

16.2.3　品質計画は16.2.1項でベースとした国際規格に基づいて文書化されなければならない。

16.2.4　品質計画は、常に最新の国際規格に適合するよう改定しなければならない。

16.2.5　あるライフサイクルのフェーズを実行する場合には、あらかじめ品質計画が明確に定められていなければならない。

フェーズの途中で国際規格が改定された場合はどうするのか？。

16.2.6 品質計画は、関連するすべての組織で、その履行内容が理解され、実施可能でなければ意味がない。

16.2.7 品質計画書の記載内容としては、以下の条項も参照すること。

5.5
7.2.3
8.2.1
8.2.8
9.2.1
12.2.10

16.2.8 品質保証のためには、保証すべき対象、手段および結果が特定（識別）できなければならない。

16.2.9 追跡可能であることは、ライフサイクル中のどの段階においても、不具合の原因の追求と対応の検討および改善結果の確認において有用である。

追跡不可能の場合には、単に要求機能の実現は可能かもしれないが、品質の保証は困難である。

16.2.10 各フェーズでの検証の実施が、ソフトウェアの品質保証の中で重要な役割を占めている。

17. ソフトウェア保守

17.1 目　標

ソフトウェア保守にどのような活動が含まれているかについて、ソフトウェア・ライフサイクル全体から体系的に定義されたものはない。ソフトウェアの保守はハードウェアなど他の分野の保守と比較し、様々な可能性や問題を持っている。ソフトウェア保守や運用における実際的な経験から、保守は開発活動以上に、インテグリティ水準を低下させることが知られている。

ソフトウェア工学において一般的に用いられているソフトウェア保守活動には、他の分野の保守のように当初設計されたインテグリテイ水準を維持するための行為としてではなく、ソフトウェアを修正したり改造する活動が含まれている。

[17.1.1]で書かれていることは、セーフティ・クリティカル・システムにおけるソフトウェア保守の目標を、要求されたインテグリティ水準の達成を保証することに置いている。これは非常に重要なことである。改造や修正などの行為が先にあり、その修正・改造工事のインテグリテイ水準を達成するのではない。改造や修正という保守の目的自身も、インテグリテイ水準を維持し、保証するためのものに限って行うことである。

起こり得る保守のケースとして、次の2つのケースがある。

(1) セーフティ・クリティカル・システムの要求インテグリテイ水準の変更

システム全体のインテグリテイ水準を維持するため、ソフトウェアに与えられ仕様（安全性機能面）の変更、あるいはソフトウェアに配分されたインテグリティ水準の変更が生じる。これら許可された変更に対し、開発と同等以上の技術・管理のもとで保守が行われる。

(2) ソフトウェアのインテグリティ水準を維持・向上する保守

ソフトウェアのインテグリティ水準を保ち、保証するために行う保守である。当初開発されたインテグリテイ水準や安全性機能が、要求を満たしていないことが明らかになった時、あるいは疑いがある時に行われる。

後者は、運用中に発見された直接ソフトウェアが原因となる障害や兆候の処置に対応している。前者はシステムとしての問題、例えば当初予測出来なかった誤

操作に、何らかの予防手段をソフトウェアを手段として追加するような処置に対応している。

[17.1.1]の中でソフトウェアの保守行為として次の3つの分類を用いている。

(1) 訂正保守

訂正保守は、仕様に定義された通りにソフトウェアが作られていない場合、それを正しくするための保守活動である。

(2) 適応化保守

適応化保守は、ソフトウェアに与えられた仕様の前提となる環境が変化した時、その変化に適用するための保守活動である。例えば接続されているセンサーが新しいものに置き代わった時、それに対応してソフトウェアを保守することに相当する。

(3) 拡張保守

拡張保守は、ソフトウェアの機能を拡張する保守であり、どこまでを保守に含めるか意見が別れる。これは、既存の部分を大量に流用して、新しいソフトウェアを作る開発行為との境界の問題である。

ここでは、先に述べた通りインテグリティ水準を維持するための拡張保守に限り保守として取り扱う。

本規格では、先に示したように保守を行う目的についてインテグリテイ水準の維持を上げており、そのための行為として3つの種類の保守を含むとしている。

保守として、ある程度の設計変更を行うのは、必ずしもソフトウェアに限ったことではない。例えばハードウェア装置を保守する場合にも2種類の行動が存在する。一つは構成要素の部品が摩耗や経年変化により劣化し、当初の機能や性能が満たされなくなることを防ぐための保守である。このための保守活動として、一定時間ごとに部品を交換したり調整する予防保守と、何か悪い兆候が見つかった時に行う修理保守（事後保守）とが良く知られている。

もう一つの保守行動は、構成管理と変更管理の配下で行われる稼働後の設計変更（設計改善）である。この行為は保守とは別に分類されることもあるが、保守

のライフサイクルで行われる活動と言う意味で保守である。

ソフトウェアの保守は、主に後者に対応する。ソフトウェアは柔軟性として改造が容易である特性と、複雑で非可視性のため改造の影響が定かでないと言う相反する２つの特性を持っている。改造が容易であることとは、改造が簡単に行えることであり、これは簡単にインテグリティ水準を低下させるリスクでもある。

この特性は、ソフトウェア保守に対し管理面からの制御が重要であることを示している。ソフトウェア保守において、プログラムを１行修正するだけで、システムのインテグリティ水準を壊滅状態にすることも可能である。

１７．２ 要　件

本国際規格は審議中のものであり、保守の要件については十分に審議がされていないようである。保守の要件について[17.2.1]から[17.2.13]まで色々な要件が上げられているが、体系的な記述になっておらず、今後の審議を待つ必要がある。現在記述されている要件をまとめると次のように分類される。

(1) 保守の手順に関する要件：[17.2.1],[17.2.5],[17.2.8]
(2) 保守の行動記録に関する要件：[17.2.2],[17.2.6],[17.2.7]
(3) 保守の技術面からの要件：[17.2.3],[17.2.4],[17.2.9]

保守行動は独立の行動ではなく、他の行動（運用や変更の許認可など）とインタフェースを持ち一連の行動として捕らえる必要がある。特に利用者側の管理活動、直接的な運用行動、ソフトウェア保守（変更）に対する品質保証、あるいは構成管理などである。これら関係部署との役割分担を含め、何らかの保守に関する行動制御手順を決め文書化する必要がある。この関連を図５保守サイクルとして示してある。これらの関連は産業分野によって異なる部分でもあり、図５をそのまま、すべての産業分野で適用させるものではない。

★保守行動の制御手順（管理手順）に関する要件

[17.2.1]
保守のための手順に関する制御（管理）サイクルを確定し、保守マニュアルなど公式の文書として文書化することが要求されている。その規定文書に規定され

る最低限の項目として、以下のものを上げている。これらのものには、手順だけでなく、その技術的な活動の水準と、活動の責任について記述される必要がある。

(1) 誤り報告
運用局面で発生したすべての誤りに対す報告、処置の手順。
(2) 誤りログ
誤り記録に関する手順。
(3) 保守日誌
保守日誌の記載手順。詳細については後述。
(4) 変更許可
変更に対するインテグリティ水準確保のための審査と許可の手順。
(5) ソフトウェア／システム構成
上位の構成要素への影響に対するインテグリティ水準確保のための手順。
(6) テスト
保守に対して行うべきテスト手順とテストの水準。
(7) 検　証
テストとは別に行われる検証の手順とその水準。
(8) 妥当性検査
変更に対する妥当性検査の手順と水準。
(9) 評　価
変更、テスト、検証、妥当性検査の有効性を評価する手順。

保守を行う者は以上の項目について、何らかの形で体系的な保守の安全性マニュアル類として文書化することを要求している。

[17.2.5]
保守のための制御（管理）手順を定義するライフサイクル上の時期を設計時としている。これは[17.2.4]で規定される、保守のために必要なソフトウェア自身の特性を、設計時に考慮することと対になっている。保守のための制御手順はインテグリティ水準をいかなる場合においても、低下させないように行うことを求めている。

制御手順が確実に実施されるようにするため、独立した役割と責任が要求されている。組織独立の程度についてはインテグリティ水準により、内部（内部独立組織／独立者）／外部の条件を満たす必要がある。詳細について付録の表で独立

の程度をガイドしている。

また設定された手順が厳密に規定され運用されること。かつ手順が厳密に規定され、かつ運用されていることを、監査により確かめることが要求されている。

[17.2.8]

保守のための制御手順は品質保証のための品質計画書（あるいはその体系下の保守計画書）で規定するが、そのガイドとして保守サイクルを図5に示している。この図は保守に関する関連項目のガイドラインとして位置付けられるものであり、図に示された項目について手順などの規定が必要である。

保守をとりまく様々な活動は構成管理や運用などを含めた体系的な活動である。よって相互の関係や手順を明確にし、図5と同程度の定義が必要である。しかし、この関係は産業分野（原子力、鉄道など）によって異なるものであり、この規格で詳細に規定することは出来ない。そこで図5と同程度のものを定義することを要求している。

★保守の記録に関する要件

保守に対する記録は、先の制御手順が厳密に運用された証拠である。

[17.2.2]

ソフトウェア保守記録簿に関する規定を、保守の活動毎に作成し運用することを規定している。その中に最低限含まれる活動として以下の項目を示している。

(1) 修正あるいは変更要求

保守の要件を審議し、決定する活動に対し、その活動の記録の規定。

(2) 範囲を広げた影響分析

保守が与える影響範囲の影響分析に関する記録の規定。

(3) 変更の詳細仕様

変更に関する詳細仕様の作成、レビュー、承認、検証などの記録。

(4) インテグリティ水準と必要な妥当性検査、回帰テスト

保守活動におけるインテグリティ水準の設定、変更、妥当性の確認などに関連する活動の記録。

[17.2.6]

保守日誌の記載、承認、指摘事項の処置などが実施されることを規定している。保守日誌とは、活動記録より詳細でかつ報告単位が短い。

[17.2.7]

保守日誌として記載すべき項目を指定している。

(1) すべての変更要求および承認
(2) 変更の進捗状況
(3) 変更の結末状況
(4) 再妥当性検査及び回帰テスト用のデータを含めた各構成要素用のテスト・ケース
(5) ソフトウェア構成の履歴書
(6) 正常運転および正常条件からの逸脱

保守日誌の記載内容について、ここに記載されている項目だけで十分か否かは、対象とする分野によって変わってくる。ここで、記載が義務付けられているものは、記載が必要な一例と捕らえるべきものである。

★保守に対する技術的な要件

[17.2.3]

保守は開発時と同じ水準の専門知識を有した作業者、開発環境、計画、管理で実施することを要求している。

[17.2.4]

ソフトウェアの保守を確実・容易にするための特性（一般的には保守容易性として、テスト容易性、理解性、変更容易性の3つの特性）を設計時に考慮して開発する（開発に対する要件）。

形式的方法、構造化方法論、モジューラア・プローチなど第11条の[技法／手段]で取り上げられているものは、いずれも保守性を向上させるものである。

[17.2.9]

変則を行った、あるいは起こった場合にその分析と報告を義務付けている。変則とは運転中における基準とは異なった応答や行為に相当する。変則の中には事故に至る可能性のあるものが含まれる。

★保守に関連する表の解釈

[技法／手段]と[権限付与の度合]の２つの表が付録として上げられている。

[技法／手段]

(1) 影響分析：保守や変更における影響分析の技法、方法
(2) 変更済みモジュールの再検証：直接変更が行われた部分の再検証
(3) 影響を受けるモジュールの再検証：変更モジュールや機能と何らかのインタフェースを持つモジュールの再検証
(4) 全システムの再検証：影響の有無を問わず、全体の再検証
(5) ソフトウェア構成管理：すべての局面で必要
(6) データ記録と解析：すべての局面で必要

[権限付与の度合]

(1) 独立会社：検証の独立性としてまったく別の会社を要求している。
(2) 独立部門：ＩＬ３以上における意味は、独立会社との併設を意味する。
(3) 独 立 者：I.L.3以上の意味は、独立者ではなく、独立部門とすることを要求している。

１８．アプリケーション分野固有の規格を派生するための手引

国際規格では、規格を適用させるための手順としてテーラリングとカスタマイズが用いられる。カスタマイズとは、規格に規定された要素の選択、つまり該当する項目を選択し、該当しない項目を削除し、規格を実際に適用させることにである。一方、テーラリングはカスタマイズより上位の考え方であり、素材として本規格を用い、別の規格（ここでは産業分野別の規格に相当）を派生させることを意味している。

本国際規格は、テーラリングについてのガイドとして、この条文を示している。

１８．１ 目　標

本国際規格はメタ規格であり、産業固有の規格に引用され、テーラリングされることを前提として作られている。つまり、本規格により規定される項目は、すべてのセーフティ・クリティカル・システムのソフトウェアに対して、直接適用を義務付けるのではなく、セーフティ・クリティカル・システムを含む産業固有の規格の中に、記載されるべき項目について規定してある。

ここで言う技術委員会とは、例えばISO/IEC JTC1 SCxxに対応し、ワーキング・グループとはSC配下のWGyyなどに相当する。

１８．２ 要　件

18.2.1　産業固有の規格を作成するにあたり、本規格の変更を認めている。産業固有の規格（国際規格や国内規格など）に限定される。

18.2.2　産業固有の規格において、側面を変更しても良いが、本規格の各条における目標が達成されることを要求している。

18.2.3　本国際規格を基にテーラリングしたことの正当性を示すため、次の項目について文書化を要求している。

(1) 項目の対応
本規格の条項との対応表を作成すること。

(2) 差がある場合

その違いを明示すること。

(3) 同等性

本規格との違いがあっても、本規格における各条の目標を達成していることの事由を明記すること。

18.2.4 本規格の付録としてインテグリティ水準に合わせた表を示しているが、その表中の項目について修正を認めている。修正部分の正当性については、アプリケーション分野固有の規格の中で与えて良い。

18.2.5 注意書きである。安全性の確証はプロダクトとプロセスの両方から行われる。プロセスが本規格に準拠していてもそれだけで安全性を確証できるものではない。

第３部　特定分野のソフトウェア安全性規格との比較評価

中　目　次

Ⅰ．航空宇宙分野

Ⅱ．鉄道分野

Ⅲ．原子力分野

I. 航空宇宙分野

細　目　次

1. 航空宇宙用ソフトウェアの開発とその安全性について
　1.1 航空宇宙機器に求められる信頼性要求の動向
　1.2 本年度の報告概要
2. ＳＭＡＰの概要
3. RTCA/DO-178Aにおけるソフトウェアの安全対策
　参考文献

I. 航空宇宙分野

1. 航空宇宙用ソフトウェアの開発とその安全性について

1.1 航空宇宙機器に求められる信頼性要求の動向

航空機や衛星／ロケットなどの宇宙機システムを安全、確実に運用するために、その開発に当たっては、個々の構成要素部品に極めて高い信頼性が求められている。

昨年度の報告書においては、航空宇宙分野におけるソフトウェア開発の考え方を紹介し、その具体的基準例として民間航空機用の基準であるRTCA/DO-178Aの特徴、及びＥＳＡのソフトウェア工学基準について報告した。また、将来動向として、ＮＡＳＡにおける人工知能技術、エキスパート・システムの扱い方、及び宇宙開発事業団におけるソフトウェア要求仕様書の制定経過を述べた。

宇宙機システム用のハードウェアの安全性／信頼性要求基準としては、宇宙開発事業団が日本の基準として、ＮＡＳＡやＭＩＬ規格を参考に昭和63年にまとめた。また平成４年度には、それまでの検討結果を基にソフトウェア標準仕様書として、宇宙用ソフトウェアが満たすべき要件を安全性／信頼性の観点から制定した。

ＮＡＳＡにおいては、これまでもソフトウェア作成基準としてＳＭＡＰ（Software Management and Assurance Program）を用いており、宇宙ステーション計画でもＳＭＡＰを基本的には用いることを勧告している。このため、我が国の宇宙関連者もここ数年宇宙ステーション計画の進展とともに多くの技術者がその訓練を受講し、相互啓発のための研究会を宇宙開発事業団を中心に組織している。

1.2 本年度の報告概要

本年度の報告では、最近注目されて来始めたこのＳＭＡＰの概要と昨年報告した民間航空機用の基準であるRTCA/DO-178Aを当ワーキング・グループでこれまで議論してきたソフトウェア安全性の全般の基準文書であるISO/IEC JTC1/SC7-N917（以下N917と記す）と比べながら紹介する。

2. ＳＭＡＰの概要

ここでは、ＮＡＳＡのＳＭＡＰにより作成された“Software Assurance Guide book, SMAP-GB-A201, 1989”の概要を紹介する。同文書は宇宙ステーションを始めＮＡＳＡ関連の宇宙機システムのソフトウェア開発に広く適用されているソフトウェア管理手法を記述したものである。

同文書では、ソフトウェア保証を

“ソフトウェアの各プロセス及び各々の要求、標準、手順に沿っているかどうかを保証する計画的／体系的な一連の活動”

と定義している。

ソフトウェア開発は本質的に技術的、プログラム的リスクに満ちており、ソフトウェア保証活動によってこれらのリスクを低減することを目指す。ＳＭＡＰに記述されたガイドの目的はソフトウェアの開発側と受入れ側の双方の要員にソフトウェア保証の手引を与えることである。

同文書の骨子を図１に示す。以下では、各節の概要を述べる。

(1) プロジェクト・ソフトウェア保証活動の確立

すべてのソフトウェア開発には程度の違いはあるが、何等かの保証活動（プログラム・デバッグに始まる）を含んでいる。

ソフトウェアの保証内容は、対象ソフトウェアの安全性、ミッション・クリティカリティ、スケジュール等々でそれぞれ異なり、課題毎にテーラリングされるものである。

ソフトウェア保証活動は、プロジェクト構造を考慮した保証計画の立案、完成評価基準の作成、及びその実施からなる。

(2) ソフトウェア品質保証（ＳＱＡ）

ソフトウェア品質保証（SQA: Software Quality Assurance）とは、ソフトウェア標準、開発プロセス、及び開発手順の妥当性を評価するためのシステマティックな手続きである。

ＳＱＡでは、標準及び手順が文書作成、設計、コーディングの各作業について定められ、成果物の監視や評価を通じて、これらの標準／手順が遵守されていることを確認する。

ソフトウェア品質保証作業の基本的手法は監査活動からなり、管理手順の遵守、ドキュメントの維持、作業状況報告の記録などに対する監査がなされる。

第１章　概　要

・概念及び定義

・ソフトウェア保証の目標

第２章　プロジェクト・ソフトウェア保証

・概念及び定義

・プロジェクト毎のソフトウェア保証のテーラリング

・ソフトウェア保証計画作成

・プロジェクト構造の考慮

・完成評価基準

・ソフトウェア保証プラン

第３章　ソフトウェア品質保証

・概念及び定義

・標準及び手順の制定

・ソフトウェア品質保証作業

・他の保証作業とソフトウェア品質保証の関係

・ライフサイクルにおけるソフトウェア品質保証

第４章　ソフトウェア品質工学

・ソフトウェア品質とは

・ソフトウェア品質工学プログラム

第５章　Ｖ＆Ｖ

・Ｖ＆Ｖ活動内容

・ライフサイクルの中でのＶ＆Ｖ

・独立機関によるＶ＆Ｖ

第６章　不具合報告と修正

・不具合報告／修正活動の概要

・不具合の相関関係

第７章　ソフトウェアとシステムの安全性

・ソフトウェアにおける課題

・ソフトウェア安全性の向上方法

・安全性確保のプログラム

第８章　セキュリティ

・セキュリティの自動生成

・セキュリティ保証活動

図１　ＳＭＡＰの骨子

(3) ソフトウェア品質工学（ＳＱＥ）

ソフトウェア品質工学（SQE: Software Quality Engineering）とは、ソフトウェアを品質評価し、アセスし、改良する方法である。

ソフトウェアの品質を定める要素には、

“信頼性、保守性、移植性、相互運用性、試験容易性、操作性、再利用性、追跡可能性、維持性、効率性”

などがある。

(4) Ｖ＆Ｖ

Ｖ＆Ｖ（Verification and Validation）とは、ソフトウェアが機能要求などの要求を満たし、正しい結論を生成することを保証することで、検証（Verification）あるいは確認（Validation）などと区別されるが、Ｖ＆Ｖの２つのＶの違いに意味は殆どない。

Ｖ＆Ｖの主な活動はインスペクションなどのレビューとテストからなる。

Ｖ＆Ｖ活動は、ソフトウェア開発の各フェーズでの活動として展開される。

Ｖ＆ＶではＩＶ＆Ｖ、即ち独立機関によるＶ＆Ｖ作業が重要となる。

Ｖ＆Ｖの手法には種々あるが、誤りを発見するのに特に有効な手法としてフォーマル・インスペクションは重要である。

(5) 不具合報告と訂正

不具合とは、問題、相違、例外、障害、異常などと呼ばれ、ソフトウェアが要求を満たしていない事象を指す。

不具合と調整作業には、発見と報告、追跡／管理、影響評価／調整作業などのプロセスがある。

(6) ソフトウェアとシステムの安全性

ソフトウェアの不具合は、要求定義段階での不具合、コーディング、タイミングのミスなどから発生する。

ソフトウェアの安全性を向上させるには、開発工程の改良と高品位なソフトウェア開発が重要である。

ソフトウェアの安全性は、要求分析、設計分析、コード分析、安全性テスト、などにより分析される。

(7) セキュリティ

セキュリティによって、ソフトウェアの紛失、不正確、改変、使用確保、誤りなどを防止する。

セキュリティ保護のレベルはそのソフトウェア及び取り扱うデータのセンシティビティによって決まる。

セキュリティ保証活動の項目には

・セキュリティの評価
・セキュリティ要求の作成と確立
・ソフトウェア審査にセキュリティ要求が含まれること
・構成管理にセキュリティが含まれること
・物理的セキュリティの確保

などが挙げられる。

3. RTCA/DO-178Aにおけるソフトウェアの安全対策

RTCA/DO-178A文書は米国の半官半民組織RTCA（Radio Technical Commission for Aeronautics）によってまとめられた航空機搭載用ソフトウェアの開発基準であり、同文書には強制力がないが、米国航空局（FAA: US Federal Aviation Administration）が認定しており、ＦＡＡの認定を受ける際の事実上の基準として用いられている。

昨年度の報告書では、同文書の興味深い特徴として、同文書がソフトウェア残存エラー確率を認識している点、及びクリティカリティの分類の考え方を紹介した。特にクリティカリティの分類は、検討中の基準文書ISO/IEC JTC1/SC7-N917においては安全性インテグリティ水準との概念に類似するものであり、興味深いことを述べた。

ここでは、基準文書ISO/IEC JTC1/SC7-N917との違いの概要を以下に述べる。

(1) 全般的な違い

・ISO/IEC JTC1/SC7-N917では安全性を確保するために採るべき手法について詳しく勧告を作成しているのに対して、RTCA/DO-178Aでは具体的な手法の勧告よりも、クリティカリティ・レベルに応じて必要となる各種文書の作成及び検証作業の勧告が中心である。

・RTCA/DO-178Aでは、規格作成に関与した個人が特定できるのに対して、ISO/IEC JTC1/SC7-N917では個人の特定は困難である。
・RTCA/DO-178Aでは、規格の適用範囲を航空機分野に限定し、基準の対象となる範囲、基準実施のプロセスが明確である。

(2) 導入部
・導入部分については、RTCA/DO-178Aでは1節及び2.1節において航空機アビオニクスに限定した論議に終始／限定しているのに対して、ISO/IEC JTC1/SC7-N917では安全関連ソフトウェア全般及びその開発手法に言及している。
・インテグリティの論議とRTCA/DO-178A-5.2のクリティカリティの議論は極めて関連深い。

(3) 基準のカバーする範囲
・RTCA/DO-178Aでは、航空機アビオニクス関連ソフトウェアを対象にしている。
・RTCA/DO-178Aでは、ソフトウェア残存エラー確率の扱いについて特記している。

(4) 他の規格の引用
・ISO/IEC JTC1/SC7-N917ではISO及びIECの安全性やソフトウェアに関連する文書を多数引用しているが、RTCA/DO-178Aには特段関連する記述が見当たらない。

(5) ソフトウェアの検証手法
・RTCA/DO-178Aではどのような手法を用いるべきかについて明示しておらず、ソフトウェアのクリティカリティ・レベルに応じて用意しなければならない文書類がソフトウェア開発の各段階での要求として定められているのみである。

(6) ソフトウェアのクリティカリティ・レベルと安全性インテグリティ水準
・RTCA/DO-178Aではソフトウェアのクリティカリティ・レベルが3段階で規定されているが、ISO/IEC JTC1/SC7-N917では5段階のインテグリティ水準と言う概念で安全性を規定している。但し、インテグリティ水準の具体的判断基準や概要は個々の適用分野に委ねるものと考えられ、基準中に明記されていない。

(7) ソフトウェア開発要員の規定
・ISO/IEC JTC1/SC7-N917ではソフトウェア開発要員の資質、資格、訓練などについての基準を定めているが、RTCA/DO-178Aには該当する記述がない。
・RTCA/DO-178Aで対象とする生産活動が航空機製造に関わるものであり、開発要

員の資質は暗黙の前提となっている。

(8) その他の項目

その他、以下の項目については双方の規定に対応する箇所があり、細目の違いは見られるものの、大筋は似ている。

①用語の定義
②ソフトウェアのライフサイクルと必要文書
③ソフトウェア要求仕様書
④ソフトウェア・アーキテクチュア仕様書
⑤設計及び開発工程
⑥検証工程
⑦ソフトウェア統合
⑧ソフトウェア機能検証
⑨評　価
⑩品質管理／保証
⑪ソフトウェア維持
⑫各分野への基準適用ガイド

参考文献

[1] Software Engineering Standards, ESA PSS-05-0 Issue 1, Jan. 1987.

[2] Software Considerations in Airborne Systems and Equipment Certification, RTCA/DO-178A, Mar. 1985.

[3] 宇宙開発事業団委託業務成果報告書：「品質管理技術の調査・検討ＳＱＡプログラム文書の作成要領等の検討(その２)」，日本航空宇宙工業会,1991.3.

[4] 宇宙開発信頼性ハンドブック，NASDA, CR18603B.

[5] 日本情報処理開発協会資料

[6] Bilardo, V.J., "A Generic Trade Study Methodology for the System Analysis of Regenerative Life Support Systems," SAE-911320, July 1991.

[7] ソフトウェア保証ガイドブック，SMAP研究会，1993.

Ⅱ. 鉄道分野

細　目　次

1．欧州鉄道における標準化の動向

2．欧州鉄道における安全標準

2．1　ドイツ

2.1.1 ＶＤＥ８３１の記述

2.1.2 ドイツ国鉄によるＭＵ８００４

2．2　フランス

2．3　イタリア

2．4　イギリス

3．我が国の鉄道分野におけるソフトウェアの安全性

4．鉄道信号システムに見るインテグリテイ水準

Ⅱ. 鉄道分野

1. 欧州鉄道における標準化の動向

鉄道の安全性技術には長い歴史と技術的経験が蓄積されている。従って、列車の安全性の確保に対する基本的制御概念は、各国とも同一であるが、用いられている制御システムや制御の仕組みはそれぞれ異なった様相を呈している。保安制御に計算機を導入したのは今から十年ほど昔に過ぎない。しかし、国が異なれば同一の制御装置であっても異なった制御手法が用いられるのが常であった。従って、この間に生み出された制御装置の種類は、国の数、鉄道事業体の数をも上回るほどになっている。この傾向は、同じ安全関連システムでありながら、国際機関の統一された規格の基に開発が行われている原子力産業や、航空機産業等と趣を異にしている。

この様な特徴も、一国の中に閉じた経営が主体の場合は、さしたる不都合ではなかった。しかし、欧州では市場統合を控え、この多様性が大きな矛盾として考えられ、相互調和を求めた検討が盛んに行われつつある。ここでは、何をもって相互の制御システムを受入れるか、また、安全性の考え方はどうあるべきかが議論されている。この中で、各国の安全性規格の現状を調査すると共に今後の統一的方向が検討されている。

2. 欧州鉄道における安全標準

欧州各国は鉄道信号の標準にそれぞれ独自の位置付けをもって対応してきた。ある国では政府が装置に一般的原理や、基礎的な要件のみを定め、鉄道管理者が自分の処理方法や詳細スペックを作るに任せている。また、別のケースでは、政府が詳細な条件を定め、鉄道管理者はこれに合致する詳細スペックを作る。

将来は、法的な要求としてＥＣによって採択され、スペックや設計や検証、認証のためのある範囲での技法に対して、共通に受入れられるような、安全関連プログラマブル電子システムの一般的スペックがあるべきとする考え方が現在重要視されている。このベースになるものが ISO/IEC 65A/WG9, WG10の標準であると考えられる。英国鉄道工業協会は、ドメイン特化した規格として RIA23 (1991) の標準創案を作っている。これはＩＥＣのＷＧ９と結合もしくは、それを拡張しようとするものである。

以下では、欧州各国の鉄道に関連する安全標準について概観する。

2.1 ドイツ

ドイツのＶＤＥ（Verband Deutscher Electrotechniker;ドイツ電気技術組合）は「鉄道信号用電気部品標準VDE0831」を作っており、これはまた同時に、DIN Norm 5783にもなっている。この標準は、鉄道信号用電気部品の設備や使用に関するものであるが、従来の鉄道信号用電気部品の拡張・変更・使用に対応するものではない。

ＶＤＥ標準（VDE0831）は、プログラマブル電子部品を念頭において特に制定されたものではない。同標準は鉄道信号用語の拡張した定義や電気的パラメータのための定量的データ、バイタルな信号システムを設計する際に用いるべき考え方の記述もなされている。

ドイツ国鉄（ＤＢ）では、これを補足するものとして、「勧告MU(nich)8004」を発行してきた。これは鉄道管理者によって使うことを保証するために、電子システムが故障した場合における確認の仕方と安全機能とを規定している。

2.1.1 ＶＤＥ８３１の記述

この標準は、ドイツにおける信号工業や電気信号装置に対する技術状況や安全標準を記述するのに必要である。最新のバージョンは１９９０年８月に出版されており、プログラマブル電子システムの必要要件をも含んでいる。その構成は次の通りである。

(1) 展　望
　鉄道信号のための電気装置の概念と動作に関する標準としている。
(2) 用語の定義
(3) 防護手段
(4) 鉄道信号用固定電気設備のための一般的要件
(5) 電気装置の必要条件
(6) 信号工学の要件
　・フェイルセーフ回路、計画、回路結線図の原理
　・フェイルセーフ回路の効能と信号工学の安全性の要件
　・品質保証処理や計画、工事の際の配慮
　・故障モードと環境

・規定や要求事項から見た、信号装置使用時のチェック事項（安全の証明）
・ユニットの単一、多重故障に対するフェイルセーフの確保

2.1.2 ドイツ国鉄によるＭＵ８００４

信号装置が故障の場合でも、その時の動作と安全性から見た機能の保証は、鉄道運営者の義務である。それゆえＤＢ（ドイツ国鉄）は、安全を証明するため基準MU8004を公表してきた。

(0) 参考文献
(1) 信号通信材料の技術的認定
(2) 必要要件に対する仕様、技術的解法
(3) バイタル回路と電子リレー技術による装置の一般的ガイドライン
(4) バイタル回路と電子装置の一般的ガイドライン
　電子装置の安全性の証明の構造も定義しており、次の節が含まれている。
　1)準備
　2)
　3) 故障の影響
　4) 環境の影響
　5)安全に関する使用方の規制
　6)テキスト
(5) バイタル回路とリレー装置の一般的ガイドライン
(6) 基本と規則
(7)
(8) 技術文書の一般的ガイドライン
(9) 各種ガイドラインに関する情報

動作特性を明らかにすることは、動作仕様の必要条件とそのための標準的な技術解法が、故障の無い状況のもとで満足され、正しく守られてきたことを示すのに役立つ。

コンポーネント故障の影響の章は、安全に関する記述の中心部である。そこでは、装置の故障が安全側だけに導かれねば成らないとしている。

環境によるトラブルの影響の章では、電磁適合性と温度や機械的影響が扱われている。装置の工事に先立って技術的認証を得ることが必要である。認証プロセ

スでは、そのシステムに関する安全文書の検査と、装置内の個々の安全性に関係ある部品の使用に関する認証プロセスを含んでいる。

信号装置や設備の安全性は、ひとえに安全な部品を使用することに基づいている。そういう部品とは、ある状態でのある変化が、1つないし複数の特性に受入れがたい変動を引き起こすことがないという、絶対的性質をもっている部品をいう。それには十分なテストを受けねばならない。

MU8004の重要部分は、様々な部品の故障リストである。安全を証明する際にはどの様な故障を考慮すべきかを示している。設計によって安全性が付与されるような部品の場合には、十分なテストが実行された後であれば、ある種の（危険な）故障が起こる可能性はない。

2.2 フランス

フランス標準化協会は、フェイルセーフデータ処理のための3つの標準を策定している。

(1) NF71011

NF71011は、安全システムのクラス、専門用語、ハードウェアとソフトウェア間の結合と制約、ソフトウェアプロジェクトの一般的方法と組織についての一般的記述である。形跡を辿れること（トレーサビリティ）や、認証計画と共に、品質、検証、認証のチームを別にする必要性についても記述している。

(2) NF71012

NF71012は、次のような現実の様々な段階を提供している。

- 公式の使用
- リスクの予備解析
- トレーサビリティ
- ソフトウェアリスクの解析と影響度評価
- モジュールの組み合わせかたとデータフローチャートを付けた、予備的概念
- 高級言語によるコーディング
- ソフトウェア検査計画
- ドキュメンテーションの書き方のスペック

2.3 イタリア

イタリア国鉄（ＦＳ）は、標準IS402を制定している。これは保安・信号システム用電子装置の供給のための技術仕様で、次の項目が掲載されている。

- 一　般
  イタリア国鉄の権利に関する一般的情報、受け入れられる国際標準、請負者が従うべき一般的規則
- 契約書類
- テストの定義と組織
- 事前チェック
- 動作チェック
- 電気・電磁テスト
- 気候環境テスト
- 機械的テスト
- 包装の規則

ソフトウェアの設計に関しては、EWICS TC7、ＥＳＡ、IEEE標準730&931、鉄道関係ではUIC738R等が推奨される。

2.4 イギリス

RIA23が制定されている。この仕様はIEC 65A-941の「産業用安全関連システムで利用されるコンピュータ用ソフトウェア」に基づいている。ＩＥＣの仕様では、以下について特に強調しつつ、設計や評価について述べている。

- ソフトウェアとハードウェアの相互依存性
- ソフトウェアの開発と必要要件の検査と故障許容設計
- システムの認証
- プログラム作成者の権限
- リスクのレベルとソフトウェア設計の戦略

さらに応用できるあらゆる技術の意味と利点、欠点をも述べている。

RIA23の仕様は、IEC 65Aの処理を鉄道信号に応用したものであり、３つの根本となる概念、すなわちインテグリティレベル、ライフサイクルモデル、個人と組織の役割と責任に基づくものである。

(1) インテグリティレベル

インテグリティの定義に関し、例えば連動装置はレベル４で最高という説明がなされる。インテグリティのレベルのレベル選択理由は、ハザード解析報告書に記述され、評価者により認定されねばならない。

(2) ライフサイクルモデル

この仕様は各々のライフサイクルの階層関係をシステムのライフサイクルと連携しつつ明確にする。各サイフサイクルは、以下の事項を含む。

- 必要条件の定義
- 開　発
- 検　証
- 認　定
- 保　全
- 廃　棄

システムライフサイクルには、以下を含まねばならない。

- 必要条件の定義と解析
- 全体システムの設計仕様
- ハードウェアの開発
- ソフトウェアの開発
- システムの受け取り
- モニタリング
- 保全と改良

また、システムのソフトウェアとデータの分離として次の事柄を述べている。仕様では、特定のグラフィカルな領域の制御に対しては、データをもとに適合させるようなシステムとすることにより、システムソフトウェアの標準化ができる。すなわち、「データ駆動型システム」とすることが望ましいとしている。さらに、データの計画、作成、検証のための要件も示している。

さらに、役割と責任に関しては仕様で、安全当局、設計当局、検証者、事前評価者を挙げ、安全責任者は、システムが実用に供することができるか否かを認定することとしている。この人物は，次のことを行わねばならない。

・設計当局、検証員、事前評価者の間の正確な独立の度合いを確認すること。
・事前評価者の参照事項を定義すること。

そして、事前評価者は、次のことを行わねばならない。

・安全計画を評価すること。
・安全計画のもととなるあらゆる記録を監査すること。
・システム要求仕様と安全レベルを評価すること。
・システムスペックを評価すること。
・システム検査計画を評価すること。

設計当局は、RIA23の仕様の要求が適用され、適合性が証明されていることを保証するための責任をシステムに与える機関によって指名されねばならない。
検証者は検証計画を作り実行しなければならない。設計当局によって行われた活動の監査と評価も行う。

3. 我が国の鉄道分野におけるソフトウェアの安全性

欧州各国のソフトウェアの安全性に対する取り組みで見られるよう、各国の鉄道によりそれぞれ多様な形態が存在する。我が国の鉄道の場合は、さらに独特な形態を採りつつ今日に至っている。我が国の電子式（計算機式）信号保安装置の開発は、1976年頃から当時の国鉄において、開始された。開発は国鉄の技術研究所と信号関連製作会社の共同で行われたが、ソフトウェア、ハードウェアのアーキテクチャや安全性に関する基本的な方針は、技術研究所が責任を持って遂行した。このため、国鉄はユーザーでもあると同時に開発者という両面の性格を持っていた。
保安制御に最初に組み込むシステムの開発には、安全性の確保を目的にハードウェア、ソフトウェア両面の検討がなされた。ハードウェアにおいては、バス同期型３重系計算機という特殊なアーキテクチャを持つシステムが開発されたが、ソフトウェアについては、依拠し得る特別な手法が見出せず、安全性向上に有効

と考えられる様々な手法や概念を組合わせて構築することとされた。しかし、マルチタスクや処理中の割り込みを避けることや、処理のチェックポイント時におけるアクセプタンステストの義務付け等、今日有効とされている多くの技法が組込まれていた。

また、論理は従来のリレーによる結線論理の概念を踏襲することとし、徹底した検査により確実性を期した。ソフトウェアが開発され、室内試験の後、フィールドでの実入力による検証試験には、3年間をあてている。さらに、実用プロトタイプを用いた室内でのシミュレータによる動的総合試験や静的検証試験がユーザーとメーカーが一体となって行われた。

欧州の事例と大きく異なる点は、欧州の場合、ソフトウェアライフサイクルの各段階で何を行うかが規定され義務と責任が明確であるのに対し、我が国では実効を挙げることをより重視し、反面明確な規定なしに作業が進められている点であろう。技術研究所が内部資料として作成した「信号保安システムへのマイクロエレクトロニクス導入指針」は、一定の基準としての性格を持ち、今日でも安全性審査に用いられている。しかし、そこには義務付けや責任に対する規定はなく、あくまでも推奨事項があるに過ぎない。もちろん、安全性に対する取り組みは真剣かつ厳しいものが有り、検査の規模や内容の深さといったV&Vの作業実態は欧州と何等変わるものではなく、事業者が参加し、安全性を検証しようとする姿勢は特筆される。

国鉄がその後分割民営化され、当時の技術研究所は独立した財団法人鉄道総合技術研究所として生まれ変わった。この中で、事業者が仕様を提供し、製作者のみが装置を開発し納入するという状況も生まれつつある。さらに、電子化が進展し、既存の信号機器をソフトウェアに置き換えるだけではなく、新たな保安システムの構築が要求される時代にも入りつつある。この様な今日の状況は、従来のソフトウェアの作り方ではなく、新たな高安全ソフトウェアの開発に向けた取り組みを必要としているといえよう。

4. 鉄道信号システムに見るインテグリテイ水準

鉄道信号システムは多くのサブシステムを抱える。IEC/65A/WG9の標準のベースには、インテグリテイ水準に応じた安全性活動の義務付けがある。しかし、現実にはインテグリテイ水準（0～4まである）をどの様につけるかが議論となろう。欧州の鉄道事業者が提案したインテグリテイ水準に関する割り当ての一覧を表1に示す。同一名称の装置でも事業者が運営する鉄道に応じ、その役割や機能が微

妙に異なる場合が多々ある。この場合には、事業者により大きくレベルの設定が異なる。また、運営する鉄道の事情により、該当する設備が全く無い場合もある。このときには適用されない“ＮＡ”といった評価にしている。

表１　鉄道事業者（Ａ～Ｉ）によるインテグルティレベルの設定事例

鉄道事業者 機能	A	B	C	D	E	F	G	H	I
連動装置	4	4	4	4	4	4	4	4	4
信号、ポイント	4	4	4	4	4	4	4	4	4
列車検知	4	4	4	4	4	4	4	4	4
列車ﾌﾞﾚｰｷ・ｼｽﾃﾑ	2	2	4	4	2	4	4	2	4
自動閉塞	4	4	4	4	4	4	4	4	4
踏切制御	4	4	4	4	4	4	ＮＡ	4	4
踏切監視	4	4	2	ＮＡ	4	2/4	ＮＡ	4	ＮＡ
保守員防護	2/4	2	2	4	3	4	4	4	4
制御盤	4	2	4	4	3	4	ＮＡ	4	2
ＶＤＵ(*1)	4	ＮＡ	4	4	3	4	ＮＡ	4	4
ＶＤＵ(*2)	4	2	4	4	2	4	4	4	4
遠隔制御(*1)	4	4	4	4	3	4	4	4	ＮＡ
遠隔制御(*2)	2	2	2	2	2	2	2	2	2
ＡＴＰ(*3)	4	4	4	ＮＡ	ＮＡ	4	4	ＮＡ	4
ＡＴＰ(*4)	ＮＡ	ＮＡ	ＮＡ	3	2	ＮＡ	ＮＡ	2	ＮＡ
ＡＴＰ(*5)	ＮＡ	ＮＡ	ＮＡ	2	ＮＡ	ＮＡ	ＮＡ	2	ＮＡ
ＡＴＯ(*5)	ＮＡ	2	ＮＡ	ＮＡ	ＮＡ	ＮＡ	2	4	ＮＡ
列車番号	2	2	2	2	2	2	2	2	2
自動進路設定	2	2	ＮＡ	ＮＡ	2	2	2	2	2

[注]*1：安全な移動を承認する能力付
*2：運転整理と線区制御の目的だけ
*3：ＡＴＰが安全を確保する
*4：運転士が主体で、ＡＴＰはバックアップ
*5：列車運転の自動化だけ
ＮＡ：適用されない

Ⅲ. 原 子 力 分 野

細　目　次

1．ＩＥＣ８８０の概要

2．Ｎ９１７規格との比較

3．現在の検討状況

Ⅲ. 原子力分野

1. ＩＥＣ８８０の概要

原子力分野では、安全上重要な機器である原子炉安全保護装置（緊急時の原子炉停止、あるいは非常用炉心冷却装置の作動を自動的に行う装置）への、プログラマブル制御装置の適用が進行中である。

各国によって法的規制内容が異なること、あるいは各原子炉タイプによってシステム構成が異なることなどにより、種々の対応が取られてきたが、安全上重要な設備であることから、プログラマブル制御装置の適用に関する国際規格の必要性が高まり、IEC880「原子力発電所の安全系に用いられる計算機のソフトウェア」（Software for Computers in the Safety Systems of Nuclear Power Stations）が1986年に制定された。

現在、多くの国が（特に欧州）、IEC880に適合していることを、許認可上の要件としている。わが国においては、許認可上の手続きにおいて、IEC880への適合性を審査されることはないが、民間レベルでは、IEC880の要求事項も見据えた形での開発・設計を行っている。

以下に、IEC880の概要と、N917規格との関連をまとめる。

検討機関：IEC/TC45の原子力計装サブ委員会（SC45A）

(注)原子力分野での国際規格としては、安全設計などの指針については、ＩＡＥＡがまとめ、さらにＩＥＣにて個別項目ごとに具体的な基準や要求事項をまとめるのが一般的である。
但し、日本においては、米国より技術導入を行った経緯より、米国の規準に従うことが多い（ＮＲＣ(Nuclear Regulatory Commission；原子力規制委員会)のRegulatory Guide、およびIEEEの各種規準）

対　　象：原子力発電所の安全系に用いられる計算機のソフトウェア

(注)本基準では、明確には規定していないが、いわゆるコントローラやシーケンサといった、特定の機能を有するマイクロプロセッサベースの制御装置のソフトウェアを対象にしていると思われる。

これは、原子力分野における「安全系」が明確に規定されており、例えば汎用計算機を用いたプラントデータ管理システムや、ＥＳ（Expert System）のようなものは「非安全系」となるため、現実的な対象としては上記のようなものだけとなることによる。
但し、将来的には対象範囲が広がる可能性はある。

記載範囲：IEC880は、原子力発電所の安全系に対して新たな機能要求を確立するものではなく、計算機システムとそのソフトウェアの特殊性に係わる、以下に示すような内容の推奨事項をまとめたものである。

(1) ハードウェアとソフトウェアの間の高度な相互依存性を考慮し、ソフトウェアに影響を与えるハードウェアの基準の確立

(2) 高度な信頼性を要求されるソフトウェアの製作を確実なものとするための、ソフトウェア開発についての一般的アプローチ

(3) ソフトウェア検証と計算機システムの健全性確認（V&V: Verification and Validation）に係わる一般的なアプローチ

(4) ソフトウェアの保守、変更および管理に対する手順

(注)本基準はあくまでもソフトウェアの設計・開発を対象としたものであり、ハードウェア単独の、あるいはシステム全体に関する要求事項は含まれない。
従って、システム機能の重要度の定義に関する記述などは殆どないし、さらにいえば安全系に対する機能要求を特に追加するものでもない。

構　　成：基本原則を述べた本文および、詳細な要求／推奨を述べた添付資料からなる。

1. 目的と適用範囲
2. 用語の定義
3. プロジェクト体制
   3.1 一般事項
   3.2 ソフトウェアの品質保証計画
4. ソフトウェア要求
5. 安全系ソフトウェアの開発
   5.1 開発理念

5.2　言語、コンパイラその他のツール
5.3　詳細な推奨案
5.4　ドキュメント化
6.　検　証
6.1　ソフトウェア検証の工程
6.2　ソフトウェア検証
7.　ハードウェア／ソフトウェア統合
7.1　システム統合計画
7.2　システム統合とハードウェア／ソフトウェア開発の関係
7.3　システム構成管理
7.4　システム統合段階
7.5　統合されたシステムの検証
7.6　エラー解決の手順
7.7　統合システム検証レポート
8.　計算機システムの健全性確認
8.1　計算機システム健全性確認レポート
9.　保守と変更
9.1　変更の要請の手順
9.2　変更を行う手順
9.3　保　守
10.　運　用
10.1　計算機システム
10.2　操作員の訓練
10.3　定期的試験に対するソフトウェア要求
添付A.　システム開発ライフサイクルとソフトウェア要求の詳細
添付B.　安全系ソフトウェア開発（設計とコーディング）への詳細な推奨事項
添付C.　ソフトウェア設計仕様書の概要（記載項目）
添付D.　言語、コンパイラ、リンカなどの選定基準
添付E.　ソフトウェアの試験手法
添付F.　必要な図書のリスト

(注)添付Bに詳細な要求事項（アーキテクチャ、プログラミング上の制約など）が優先度とともに記載されている。しかし、総じて網羅的であり、要求と推奨とが混在している。

2. N917規格との比較

IEC880は、ソフトウェア開発のライフサイクルの定義と、各ステップでの検証の明確化という点で、N917規格と同様の内容になっていると考えられる。しかし、N917の大きな特徴である安全機能に応じた重要度分類（インテグリティレベル）の定義と、それに基づく要求事項の設定という概念は、IEC880には見られない。

これは、原子力発電所としてのシステム設計における安全機能の重要度分類や、各分類カテゴリー間の独立性の確保などが、長い年月をかけて既に確立していることによる。さらに、安全系へのプログラマブル制御装置の導入に際しても、従来の安全系の機能およびシステム構成をそのまま踏襲しており、N917でいうところのインテグリティレベルが、1つのシステム内で混在することもないため、特にシステム分類に関する内容を含んでいないと考えられる。

また、N917の第16条の「評価」のような、システム監査についての記述も、IEC880には記載されていない。しかし、原子力分野においては、各国の行政機関による設計審査（日本でいう安全審査）が行われており、現状においても、システム設計レベルからみた安全設計の確認が第3者によって実施されているため、該当する記述がないと考えられる。

一方、IEC880の対象としているソフトウェアは、N917でいうところのインテグリティレベル-4であると考えられるが、N917のガイドラインについての詳細表にあるインテグリティレベル-4に対する勧告のすべてが、IEC880の中でも同様に扱われている訳ではない。

N917の場合、各インテグリティレベル同士の相対的な関係より、詳細表の勧告事項の「HR」、「R」などの振り分けが決定されているのに対して、IEC880では「安全系」という1つの分類内の記載になっているため、推奨事項の重み付けがはっきりとは読み取れないことが1つの要因となっている。

また、IEC880では、Formal Method などについても記載がない。

このように、同じIEC内で検討された同様の規格で、相互に若干の相違があるが、今後、後述するIEC880の改訂作業などを通じて相互の協調が図られてゆくものと考える。

3. 現在の検討状況

IEC880は、記載内容を補足・強化するために、2回に分けて補遺を発行する計画がある。

補遺1には、以下の項目が記載され、ドラフトが作成されている。

(1) 共通要因故障に対する多様性による防護対策
(2) Formal Method - 仕様と設計
(3) 開発／検証用自動化ツール
(4) 既存のソフトウェアの利用

また、補遺2については、以下の項目を記載する予定である。

(1) 保守と保守性
(2) セキュリティ
(3) オンライン計算
(4) ソフトウェアアーキテクチャ
(5) 通信リンク
(6) 言　語
(7) ハードウェア自己診断のためのソフトウェア

前述のとおり、IEC880はソフトウェアのみを対象としているという限界があるため、システム全体に対する基準を作成することが検討されている。構成の案を以下に示す。

(1) 安全性の目標
(2) 計測制御系に対する要求事項
(3) システム構成への要求事項
　・システム設計とその最適化
　・システム設計に用いるツールと手法
　・保守に関する要求
　・Commissioning に関する要求
　・運用上の要求
(4) プロジェクト体制に対する要求
(5) 品質保証に対する要求

第４部　文献紹介

中　目　次

Ⅰ．ＩＥＣ暫定規格の概観：プログラマブル電子システムの機能別安全性

Ⅱ．信号保安システムに関する相互受入れ

Ⅲ．セーフティ・クリティカル・コンピュータ・ソフトウェア用の英国防規格の開発の根本的理由

Ⅳ．潜在的に危険な産業でのソフトウェア安全性検定

Ⅴ．参考文献一覧

I. IEC暫定規格の概観：プログラマブル電子システムの機能別安全性

細　目　次

1．はじめに
　1.1　全　般
　1.2　システム暫定規格
2．汎用規格と特定用途向け規格
3．範　囲
4．用語定義：安全性インテグリティと信頼度
5．一般安全原則：システム安全原則
6．安全関連システム（ＳＲＳ）
7．リスクと安全性インテグリティ
8．システムのインテグリティ水準とソフトウェアのインテグリティ水準
9．安全性ライフサイクル
　9.1　ハザード解析およびリスク評価
　9.2　安全性要求仕様
　9.3　ＳＲＳの特定・指名
　9.4　設計および構築
　9.5　安全性の妥当性検査
　9.6　運転および保守
　9.7　システムの改造と退役
10．今後の進め方
参考文献

ＩＥＣ暫定規格の概観：プログラマブル電子システムの機能別安全性

(An Overview of IEC Draft Standard: "Functional Safety of Programmable Electronic Systems")

著　者：R. Bell(Health and Safety Executive, UK)
S. Smith(B H-F(Triplex) Inc., USA)

出　典：COMPASS'90, June 1990, pp.151-163

1. はじめに

1.1 全　般

一般には、プログラマブル電子システム（PES: Programmable Electronic System）と呼ばれているコンピュータに基づくシステムが、安全関連アプリケーションに、ますます利用されるようになってきている。この傾向は、システムが提供する潜在的な利点により、間違いなく続く。しかし、その潜在的な安全面の利点は、もし適切な設計方法論および評価方法論が使用された場合のみ実現しうる。都合が悪いことに、ＰＥＳの特徴的な保護措置（feature）の大半を考慮に入れても、その安全性インテグリティ（Safety Integrity）の水準に関して、在来技術による複雑でないハードウェアに基づくシステムで入手できる信頼度水準（degree of confidence）と同じ水準にあるとは予測することはできない。

いくつかの機関が、この技術のあぶなげない開発を可能にするガイドライン類を発行、あるは開発している。英国では、主要な安全規制機関である労働省（Department of Employment）のＨＳＥ（Health and Safety Executive；衛生安全庁）が、安全関連アプリケーションに利用されるＰＥＳに対するガイドラインを開発した。（［訳注］ＨＳＥは核施設の認定機関で、核査察も担当している。）このガイドラインは、産業利益団体と一緒にかなりの調査および討論をした後で、他の国での進行中の同種の作業について考慮を払って開発された。このガイドラインでは、システム・アプローチを採用し、プログラマブル電子装置（PE: Programmable Electronics）と一体となったシステムに対する設計および評価の両方で系統的アプローチを採用してもらえるようにする方法論も規定している。

このガイドライン全体の表題は「安全関連アプリケーションに利用されるプログラマブル電子システム（PES）」で、1987年6月に発行された。次に示す2分冊が発行された[1]。

1)「総論的ガイド」

2)「汎用の技術ガイドライン」

このガイドラインは汎用基盤であり、アプリケーションをとわず、複数個のPESが合体したシステムの安全性インテグリティの水準を決定できるようにしている。

ドイツ連邦共和国では、予備的な暫定規格 DIN V VDE 0801 が最近発行された[2,3]。また、欧州共同体域内でも、執行機関命令（Machinery Directive）での該当要求事項に関連する、各国の国家規格を整合ヨーロッパ規格（Harmonised European Standards）に一致させる作業での重要な要素項目は、（PESを採用したシステム[4]をも含めて）安全関連制御システムにかかわっている。

米国では、ISA（Instrument Society of America；米計測器協会）が「安全性アプリケーションに利用されるプログラマブル電子システム（PES）」に関する規格の作成を大いに前進させた。また、米化学技術者協会の理事会であるCCPS（Centre for Chemical Process Safety；化学プロセス安全性センター）が、化学プロセス領域用のガイドラインを作成した[5]。

ガイドラインの開発では、国家レベルおよび国際レベルの両方の標準化機構で進行中の作業について考慮を払い、国内で開発されたガイドラインを国際レベルに発展させることを保証することが重要である。主要な目標は、国際標準化の達成でなければならない。IEC（International Electrotechnical Commission；国際電気標準会議）では、最近、次に示す表題の2つの暫定国際規格（DIS: Draft International Standard）を発行した。

1)プログラマブル電子システム（PES）の機能別安全性（Functional Safety）：汎用的な諸側面（Generic aspects）；第1分冊：共通の諸要件

この暫定規格（“システム”暫定規格）は、作業部会10（IEC/SC65A/WG10）が作成した[6]。

2)産業用安全関連システムの用途で利用されるコンピュータ用ソフトウェア

この暫定規格（“ソフトウェア”暫定規格）は、作業部会９（IEC/SC65A/WG9）が作成した[7]。

これらのＤＩＳの両方が、意見を求めるために、すべての国々の国内規格委員会（NSC: National Standards Committee）に送付された。この諮問（consultation）は、これらの２つの重要な分野での頑健な国際規格の作成において、必要で重要なステップと認識されている。この諮問期間は、1990年初めに終了した。この２つの国際規格の開発を分担する技術委員会（IEC/SC65A）は、1990年６月に、米ノース・カロライナ州Raleighで開催される会合で、今後の進め方について、公式に決定することになっている。

１．２　システム暫定規格

システム暫定規格[6]の範囲は広範で、すべてのアプリケーション領域に適用され、完成したあかつきには、他の特定用途向けの規格の拠り所となる基盤を形成することになる。これは、異なるアプリケーション領域を網羅する規格間での高レベルの互換性をもたらすはずである。そのような互換性は、重大な安全面や経済面での利点をもたらす。

本暫定規格を作成する作業部会（IEC/SC65A/WG10）では、本文書に対するさらなる考察が必要であることを受け入れ、その将来に対する潜在的な戦略面での重要さを考えると、今日までに開発された本提案書に対する諸意見を得ることが重要であったと痛感している。

本暫定規格は、ＰＥＳが安全機能を遂行するのに利用される場合に注意を注ぐ必要のある側面に関するガイドラインを規定している。ＰＥＳは、次のように定義されている（図１参照）。

「制御、安全保護、または監視の目的で、センサー（入力装置）および／またはアクチュエイター（actuator）（出力装置）が結合した１台またはそれ以上の中央処理装置（CPU: Central Processing Unit）に基づくシステム」

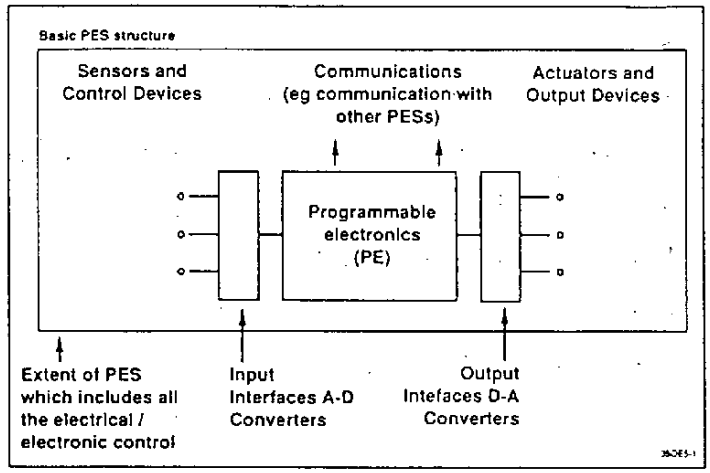


図１　ＰＥＳの構造と記法を図解したダイアグラム

本ＤＩＳは、次の目的を満たすように開発された。

・安全関連アプリケーションに利用されるＰＥＳの機能別安全性（Functional safety）の面に系統的に注意を注ぐことができるようにすること。
・急速に発展する技術を考慮に入れていること。
・安全性インテグリティ評価基準および評価技法の諸変化を反映して規格の将来の開発が可能であること。
・将来の特定用途向けの国際規格の開発が可能であること。

以下では、システム暫定規格の概観を与え、その意図した目的を説明し、重要な基本となる特徴的な保護措置を強調し、そして、今後の進め方について検討する。

2. 汎用規格と特定用途向け規格

過去において、様々なアプリケーション領域（例えば、製造、医療、プロセス、輸送など）では、安全関連システム（SRS: Safety-Related System）用の独自の規格を開発した。これらは、他のアプリケーション領域の規格とはほとんど共通する部分のない、特定の、そして時々は独自のやり方で発展した。プログラマブル電子デバイスの出現と、それらのシステムへの一体化（ＰＥＳ）により、ＳＲＳに対するガイドラインに関係するすべての領域で重大な問題が発生した。例えば、もし、各アプリケーション領域で、各アプリケーション領域用の規格を開発する際に独自のアプローチを開発したならば、重大な資源問題が起こるであろう。すべてのアプリケーション領域で利用可能な共通の、即ち、汎用のアプローチを開発することがより望ましい。特定用途向け規格を開発することを導く汎用国際規格化の概念は、標準化の分野では新しいことではない。この概念が、図２に示されている。

本暫定規格は、汎用基盤であり、１つの主要な目標は、他のアプリケーション領域で特定用途向け規格を開発する際に本規格を利用できるようにすることである。このアプローチを採用することにより、各特定用途向け規格が同じ基本原則に基づくことを可能にしている。これは、１つ以上のアプリケーション領域に属する人達、例えば、装置供給会社の人達に対し多くの利益をもたらすはずである。また、それは、汎用レベルで行われる将来の作業がすべてのアプリケーション領域のためになることを意味している。

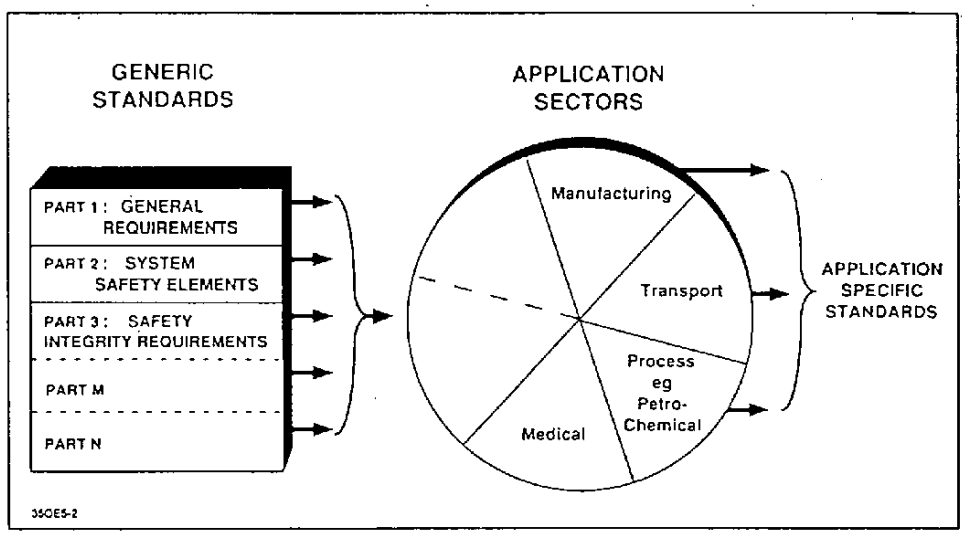


図２　汎用規格と特定用途向け規格

本暫定規格はシリーズの第１分冊であり、各分冊では、汎用的なやり方で特定の側面を扱っていることが主たる注意を引く顕著な特徴となっている。将来発行される各分冊では、次に示す項目を網羅することが提案されている。

・ハザード解析
・リスク評価
・システム安全原則（第２分冊の予定）
・安全性インテグリティの要求仕様（第３分冊の予定）
・システムのインテグリティ水準
・システムの妥当性検査
・後からの取り付け（retro-fitting）
・文書化

３．範　囲

本暫定規格は、プログラマブル電子装置（PE）あるいは他の複雑な電子デバイスと一体となったＳＲＳを主たる対象としている。しかし、本暫定規格では、これらのシステムが基盤としている技術（例えば、電気機械装置、固体電子装置、プログラマブル電子装置、空気圧／水圧／油圧式装置など）をとわず、ＳＲＳに適用可能な枠組みを規定している。各国の国内規格委員会（NSC）に対して、諮問期間中に、本暫定規格が、ＰＥＳのみに集中する方がいいのか、あるいは、（システムが基盤としている技術をとわず）すべての型のＳＲＳを網羅する方がいいのかについて、どちらを選ぶかを知らせることが要請された。

特に、本暫定規格は、次に示す要求事項を満たすように開発された。

・ＳＲＳで、プログラマブル電子装置（PE）あるいは他の複雑な電子デバイスと一体となった１つ、あるいはそれ以上の同種のシステムに適用できること。

・汎用基盤であり、アプリケーションをとわず、設計および評価の両方の目的に利用できること。本規格の対象範囲内のアプリケーション領域の例は、下記の通りである。

・プロセス産業（緊急時シャットダウン・システム、火災およびガス漏れ検知システム、ボイラー制御）
・製造業（工業用ロボット、工作機械）
・輸送業（鉄道信号、制動システム、エレベーター／リフト）
・医療（電子医療機器、例えばレントゲン写真）

・主として人間に対する安全に係わるが、環境問題にもまた適用できること。

・ＳＲＳの全構成（TC: Total Configuration）（即ち、ＳＲＳの全階層）に適用できること。

・要求された安全性インテグリティ水準が、考慮対象のＳＲＳにより、諸条件下で達成されていることを保証するために必要なすべての活動を網羅する安全性ライフサイクル・モデル（Safety Lifecycle Model）を使用すること。

・ＳＲＳの設計および評価に関係するすべての関連する側面に向けること。

・標準化機構での当該担当の技術委員会が特定用途向け規格を開発するのを容易にするように開発されていること。

・特定の勧告が入手できないような分野にＰＥＳが有する安全機能を適用する際のガイドラインを規定すること。

・ＳＲＳの安全性インテグリティの水準を規定するための“安全性インテグリティ水準”の概念を導入すること。

［注］本暫定規格では、特定のアプリケーションで要求される安全性インテグリティ水準を分類するためのガイドラインを規定していない。これは、該当アプリケーション領域の知識あるいは他の関連する経験に基づいて確定されなければならない。特定アプリケーション領域を取り扱うことに責任のある技術委員会が、許容できるリスク水準と結び付けて安全性インテグリティ水準を規定するであろう。

4. 用語定義：安全性インテグリティと信頼度

用語“安全性インテグリティ”と“信頼度”との違いは、本ガイドラインの文脈下において特別な意味を持つ。図３に示す、２つの部分Ａ、Ｂからなる円について考えてみる。円の全体は、あらゆる原因で発生する故障（failure）を表している。

・領域“Ａ”は、“偶然的ハードウェア故障”（Random Hardware Failure）による故障を表している。即ち、予測できない（即ち、偶然的な）回数だけ発生する様々な故障メカニズムに起因するハードウェア故障である（［訳注］故障メカニズムとは、「物理的、化学的、機械的、電気的、人間的原因などでアイテムが故障を起こす仕組み」である。）。

・領域“Ｂ”は、“系統的故障”（Systematic Failure）による故障を表している。即ち、システムの仕様作成、設計、構築、運転、あるいは保守でのある発展段階において作り込まれる誤りによる故障である。系統的故障は、複雑なシステムの文脈下では特に重大であり、それ故に、特にＰＥＳに関係する。

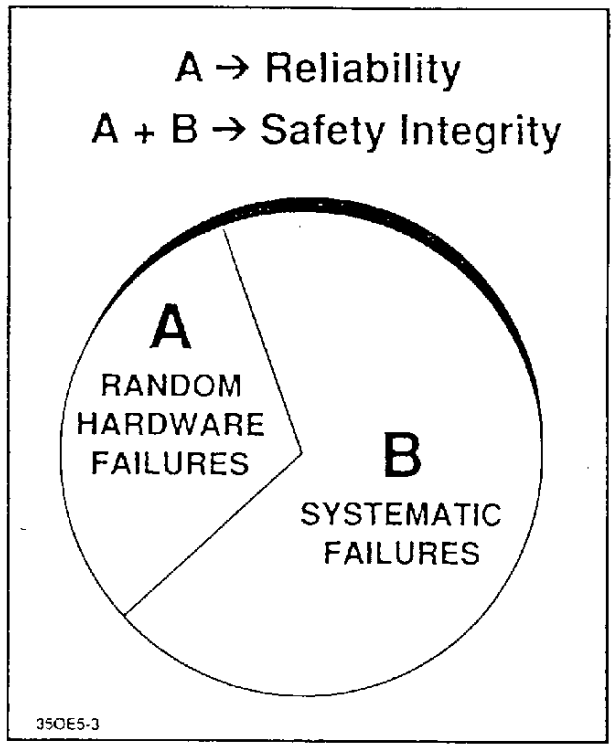


図３　安全性インテグリティと信頼度

用語“信頼度”は、偶然的ハードウェア故障に対して、システムの中に組み込まれている諸々の予防措置の度合を指すのに使用されている。特に、これらの故障は、起こり得る危険な事態に至ることになる故障である。本ガイドラインでは、この用語を通常使われている意味よりももっと狭い意味で使用しており、その故障によって危険モード（dangerous mode）に移行するハードウェア故障のみに限定するように使用されている。通常の用法では、この用語には、安全モードと危険モードの両方に移行する故障が含まれる。

用語“安全性インテグリティ”は、すべての故障（偶然的ハードウェア故障と系統的故障の両方）の原因に対して講じられた予防措置の度合を指すのに使用されている。それ故に、安全性インテグリティは、安全性要求仕様の誤りあるいは脱落、ソフトウェアのインテグリティ、ＳＲＳの電気的な妨害に対する免疫度、そして、システムを維持するために使用されている手段のような諸々によって影響をうける。

“偶然的ハードウェア故障”に関しては、信頼度予測技法を使用すれば、容認できる正確度で定量的に示すことは可能である。都合が悪いことに、“系統的故障”に関しては、定量化手段を使用しても、容認できるいかなる正確度でも予測することはできないので、定性尺度が採用されなければならない。

5．一般安全原則：システム安全原則

安全関連ＰＥＳでの安全運転を保証するには、ＰＥＳ故障を発生させるすべての様々な可能性のある原因を特定し、その各々に対して、十分な予防措置が講じられていることを保証する必要がある。本暫定規格では、次に示すように、故障の型を分類している。

・偶然的ハードウェア故障
・系統的故障

安全関連ＰＥＳに対するいかなる設計・評価戦略でも、両方の型の故障を考慮に入れなければならない。ＳＲＳでの危険モードに移行する故障を取り扱っている本暫定規格で採用した設計・評価に関する戦略では、偶然的ハードウェア故障と系統的故障の両方に対して十分な予防措置を提供することをもくろんでいる3つの“システム安全原則”（System Safety Element）（システム特性（system characteristic））に基づいている。システム安全原則は、下記の通りである。

・システム構成（Configuration）
（［訳注］例えば、チャネルの多重化）
・安全関連ハードウェアの信頼度
（［訳注］例えば、安全防護装置（Safeguard）の信頼度）
・系統的インテグリティ
（［訳注］例えば、要員や開発作業・活動の質の向上）

システム安全原則に関する正確な“一揃い”（package）は、達成すべき安全性インテグリティの水準に依存するはずであるので、それ故に、特定のアプリケーションに依存するはずである。

［注］提案された国際規格の将来作成される部分で、個別のシステムに要求されるインテグリティ水準を達成するための該当システム安全原則に関する最小の技術要件を開発することが提案されている。

6. 安全関連システム（ＳＲＳ）

ＳＲＳの概念は、暫定規格の基盤である。ＳＲＳは、次のように定義されている。

「当該システムの制御下にある装置（EUC: Equipment Under Control）に対して安全状態に到達させたり、あるいは、ＥＵＣに対して安全状態を維持したりするのに必要な要求された安全機能を、他のいかなるシステムとも独立に、実行するシステム」

「要求された安全機能の実行に必要とする安全性インテグリティ水準を自分自身で、あるいは他のＳＲＳを利用して達成するシステム」

用語ＳＲＳは、許容リスク水準を、他のシステムとは独立に、満足させることを可能にするシステムを言う。ＳＲＳは、２つの型：①制御システム、②保護システムに大別できる。すべての制御システムが、必ずしもＳＲＳと称される必要性がないことに注意すべきである。安全性インテグリティの要求された水準がＥＵＣの主制御システム中にある安全機能を実行することによって達成されるかもしれないし、あるいは、安全機能が安全専用の分離・独立したシステム（例えば、保護システム）によって実行されるかもしれない。人間がＳＲＳの一部となることができることに注意すべきである。

ＥＵＣの制御、保護、あるいは監視用の目的でシステムの階層を構成するすべてのこれらのシステムは、“システムの全構成（TC）”と称せられる。（［訳注］システムの全構成（TC）には、ＳＲＳと、非ＳＲＳであるすべての他のシステムが含まれる。）安全性インテグリティの必要な水準を一緒に達成するすべてのＳＲＳは、“ＳＲＳの全構成（TC）”と称せられる。

本暫定規格で開発された諸原則では、許容できるリスクを満たすのにたる安全性インテグリティ水準を達成するのに必要とするＳＲＳの全構成（TC）に適用され、多くのケースで、ＰＥＳからなるＳＲＳと非ＰＥＳからなるＳＲＳの両方を含んでいる。

7. リスクと安全性インテグリティ

本暫定規格では、トータル・システム・アプローチを採用しており、このアプローチの一部分では、ＳＲＳの安全性インテグリティからＥＵＣのリスクを分離することが必然的に必要となる。

“リスク”とは、規定されたハザード事象が発生する頻度、即ち、確率と、そのハザード事象の結末（consequence；［訳注］厳しさ(severity)のこと）との組み合せである（［注］リスクの概念には、２つの要素：①ハザード事象が発生する頻度、即ち、確率と、②そのハザード事象の結末（厳しさ）とが必ず入っている。）。

“安全性インテグリティ”とは、ＳＲＳが、規定された期間中、規定された諸条件下で、要求された機能を果たす尤度である。

達成すべき安全性インテグリティ水準を決定する際には、不安全状態に至らせるすべての故障（偶然的ハードウェア故障と系統的故障の両方）の原因に対して講じられた予防措置の度合を考慮する必要がある。偶然的ハードウェア故障は、危険モードに移行する故障の故障率のような尺度、あるいは、保護システムが要請された際に運転に失敗する確率を使用すれば、定量化可能である。しかし、システムの安全性インテグリティに関しては、正確に定量化することのできない多くの因子にまた依存するので、定性的に考慮されなければならない。

ある特定のアプリケーションでは、許容できると考えられるリスク水準がある。いかなる予防措置をも講じなかった場合のリスクの水準では許容できないという事態では、ＳＲＳを適用することにより、そのリスクが許容リスク水準まで軽減したことを保証する必要がある。この普遍的な概念が図４に示されている。図４から、例えば、安全関連制御システム（例えば、保護システム）を含めた安全手段を採用してハザード事象の発生する確率を減少させれば、許容リスク水準が満たされることがわかる。

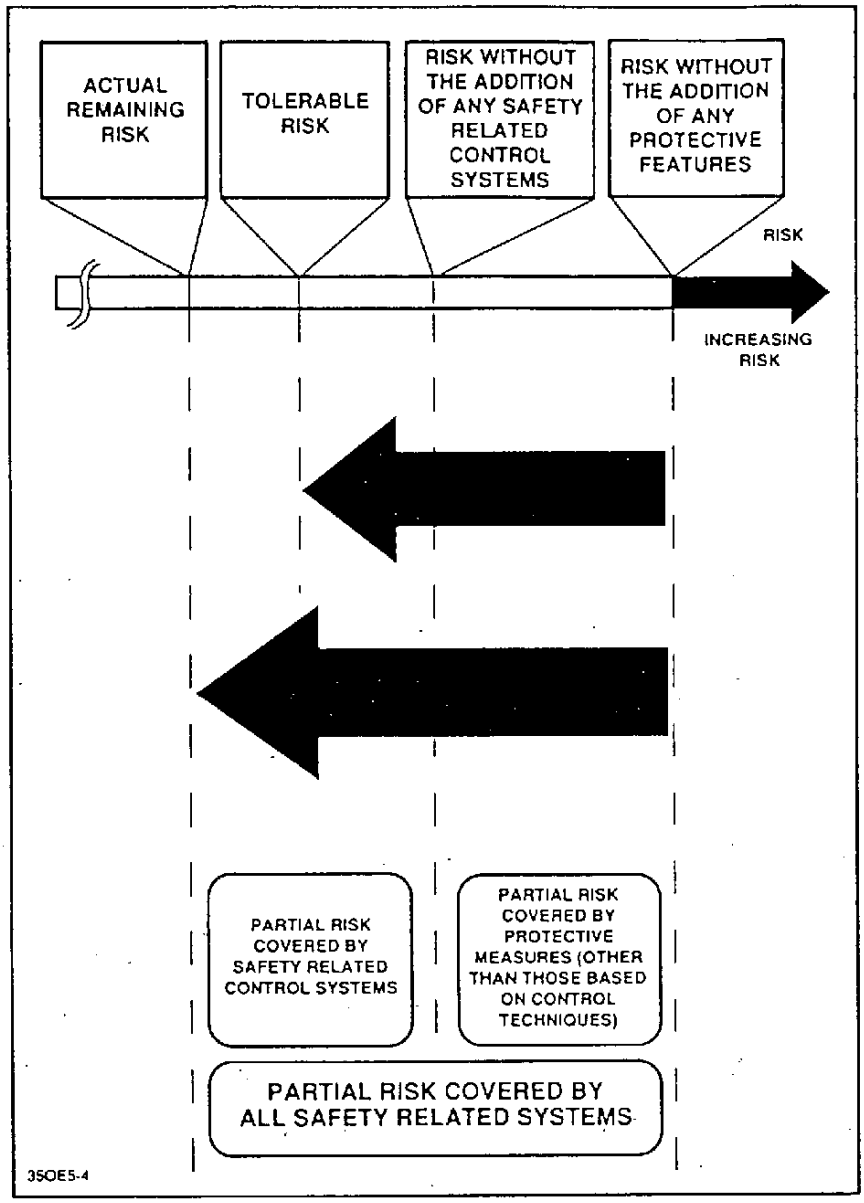


図４　リスクの軽減：普遍的な概念

許容リスク水準という概念の使用目的は、ハザード事象の発生する頻度（即ち、確率）と、そのハザード事象の結末（厳しさ）との両方で決まるそのリスク水準が許容であると考えられることについて言及するためである。ハザード事象を発生させるのは、ＥＣＵである。ＳＲＳは、ハザード事象の発生頻度（即ち、確率）、および／また、ハザード事象の結末（厳しさ）を軽減するように設計されている。

安全性インテグリティは、安全機能を遂行するＳＲＳの性能に関係する。要求された安全性インテグリティに関しては、安全性インテグリティ要求仕様（SIRS : Safety Integrity Requirements Specification）で規定され、そして、３つのシステム安全原則（①システム構成、②安全関連ハードウェアの信頼度、③系統的インテグリティ）の項で定式化されている（図８参照）。

許容リスク水準は、通常、問題としているアプリケーション領域での利害関係者によって設定される。即ち、それらは、社会的基準に基づいて決定される。安全性インテグリティは、ＳＲＳを工学用語で規定することを認める工学的概念である。ＳＲＳの全構成（TC）で達成される安全性インテグリティ水準が、許容リスク水準を適切に満たす必要があり、そして、問題としているアプリケーション領域によって設定される。リスクと安全性インテグリティとの分離が図５に示されている。

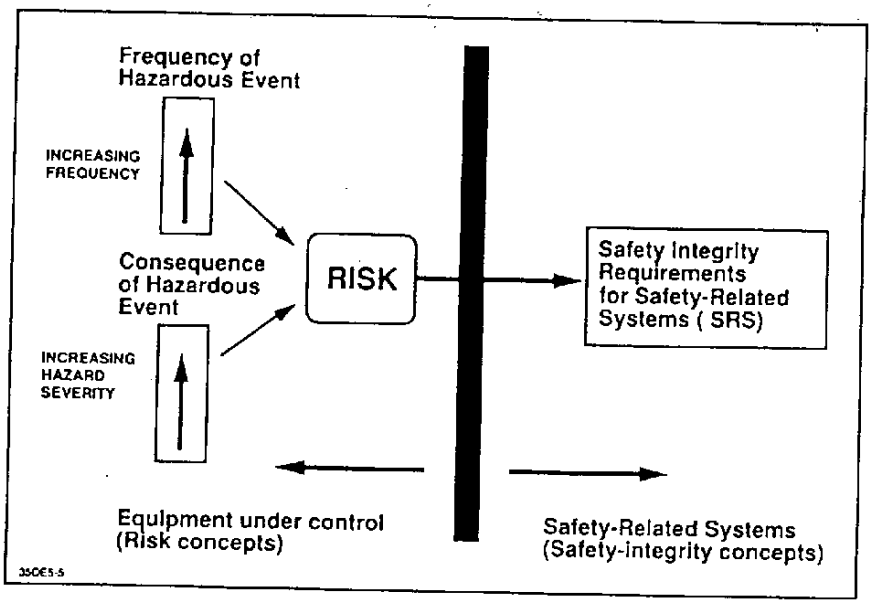


図５　リスクと安全性インテグリティとの分離

8. システムのインテグリティ水準とソフトウェアのインテグリティ水準

本暫定規格では、“システムのインテグリティ水準”と“ソフトウェアのインテグリティ水準”の概念を採用している。システムのインテグリティ水準の水準数に関してはまだ決定していないが、これに関しては、本暫定規格が完結する前の近い将来に完結する必要がある。安全関連ソフトウェアを取り扱っているＩＥＣ暫定規格[7]では、ソフトウェア用として４つのインテグリティ水準を規定している。この普遍的な概念が図６に示されている。

システムのインテグリティ水準とソフトウェアのインテグリティ水準の概念を図解した簡単な例の要点を以下に述べる（図７参照）。許容リスク水準が規定され、もし、個別のＳＲＳを組み合せることによって、インテグリティ水準２（即ち、ＳＲＳの全構成（TC）のインテグリティ水準が２である）が得られるのならば、この許容リスク水準が満たされると仮定する。各々のＳＲＳは、他のＳＲＳとは独立に、要求された安全機能を遂行する。個別システムの内の１つ（＃１）が布線式システム（hard-wired system）（非ＰＥＳ）で、残りのもう１つのシステム（＃２）はＰＥＳである。図７では、各ＳＲＳはインテグリティ水準３を有しているので、ＳＲＳの全構成（TC）のインテグリティ水準２が、２つの個別のＳＲＳを組み合わせることにより導き出されることが分かる。

［注］規定されたインテグリティ水準を有する個別のシステムを一緒にして加算するための体系的な諸規則や、ソフトウェアのインテグリティ水準をどうやって確定するかに関しては、まだ完結していない。

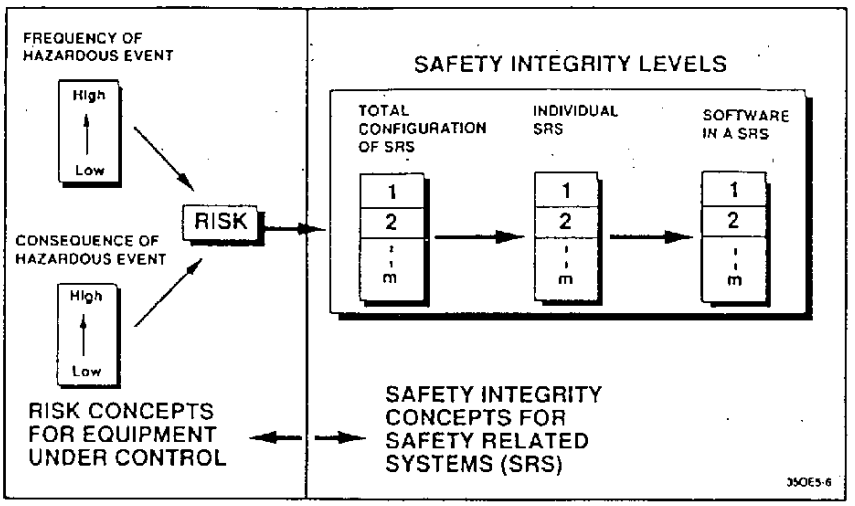


図６　ＥＵＣのリスク水準とＳＲＳの安全性インテグリティ水準：全体の枠組み

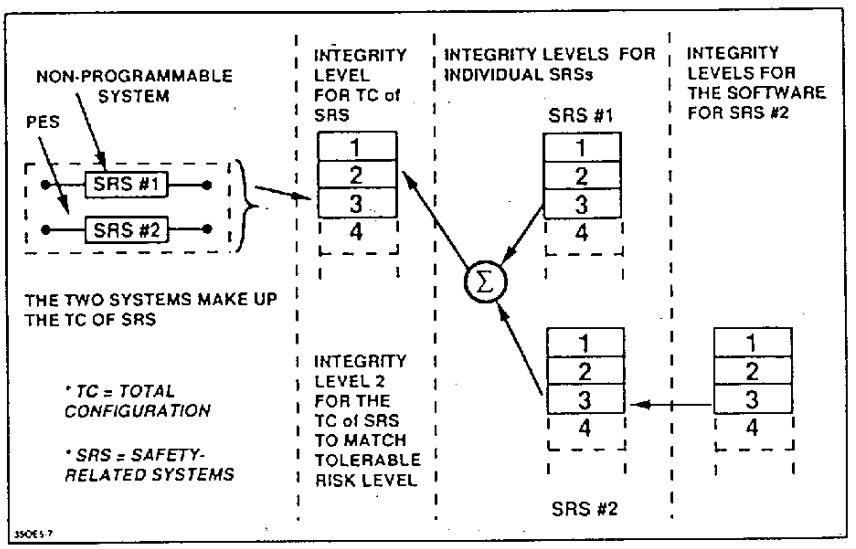


図７　インテグリティ水準の概念と記法を図解した簡略化された例

9. 安全性ライフサイクル

本暫定規格の開発中に、作業部会では、ＰＥＳに基づくＳＲＳを実現するに際して、ある特定の工程が必要であることが認識された。これらの工程は、次に示すように、多くの従来の設計プロジェクトと同様に、自然なシーケンスの実施順に整列する。

・解　析
・仕様化
・設計および構築
・妥当性検査
・保　守

ＰＥＳに基づくＳＲＳを開発し、証明するのに関係する独特の工程は、この設計シーケンスから抽出することができ、安全性ライフサイクルの基盤を形成する。安全性ライフサイクルの概念の利点は、次に示す通りである。

・それは、ＳＲＳの開発で必要とする工程のすべてを明示的に特定するための枠組みを規定し、従って、本暫定規格そのものに対する枠組みを規定している。

・それは、開発の諸工程での作業工程の実施順と構造面に関係している。

・それは、作成しなければならない解析書、設計書、そしてテスト書に対する構造を規定している。

・それは、従来の“安全関連でない”システムの開発ライフサイクルと一致している。それ故に、設計品質保証に関する既存の作業の大半を直接適用可能である。

安全性ライフサイクルの目下の形式を図８に示す。次に示す作業の実施順的な関係に関しては、現在、いくつかの論争がある。

・安全性要求仕様
・ＳＲＳの特定・指名（Designation）

何人かの作業部会のメンバーだけでなく、何人かのレビューア達も、実際に言われているように、安全性要求仕様の開発の前に、通常、ＳＲＳを設計していることに気付いている。この１つの構造的側面が起こり得る改造に支配されやすいということを理解すれば、安全性ライフサイクルの各ステップを以下に示すように記述できる。

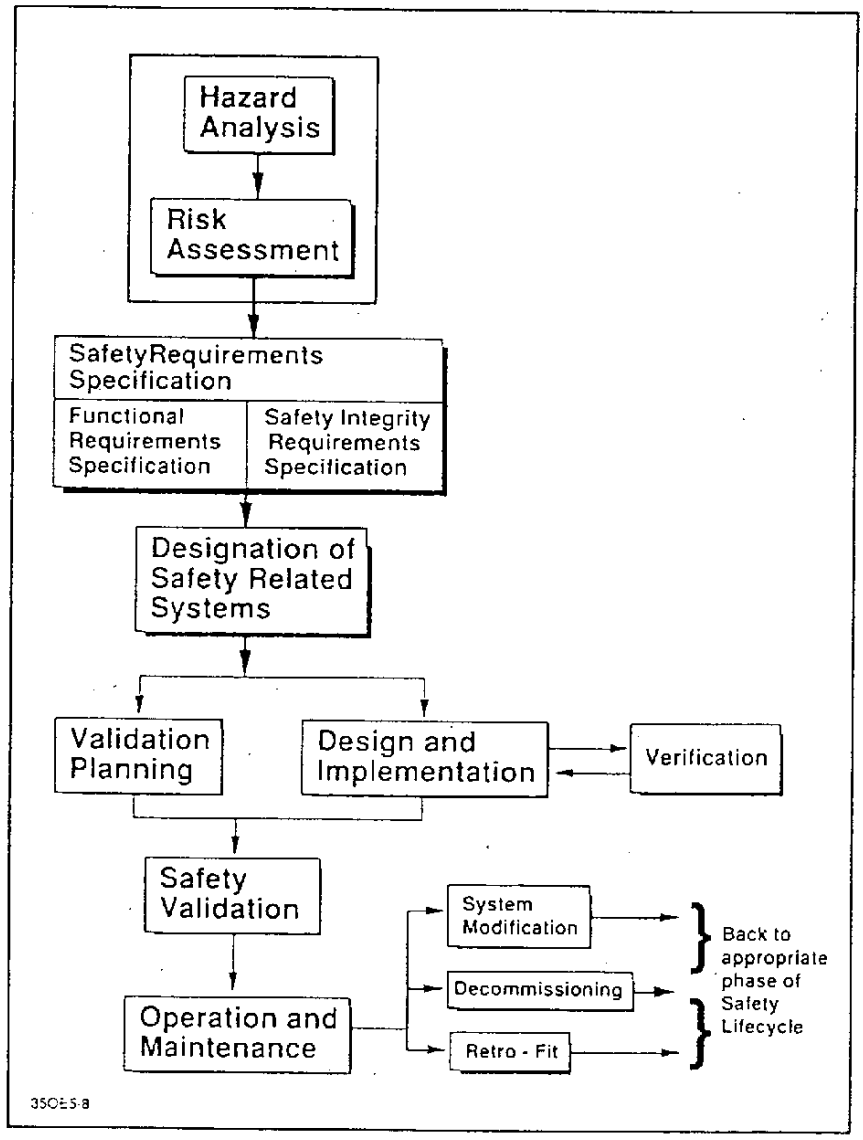


図８　安全性ライフサイクルのモデル

9.1 ハザード解析およびリスク評価

ハザード解析は構造的解析であり、その解析の度合は問題としているリスクの重大度と、その対象とするシステムの複雑度に依存する。それは、次に示すように、多様な受け入れ可能なツール類を使用して実現される。

- 故障の木解析（FTA: Fault Tree Analysis）
- 故障モードとその影響の致命度解析（FMECA: Failure Mode, Effect and Criticality Analysis）
- ハザード・オペラビリティ・スタディー（HAZOP: HAZard and OPerability studies）

ハザード解析での目標は、その事象が発生したら、どんなハザードを生じるかを検討し、その事象とともにすべてのハザードを特定することである。

リスク評価は、ＥＵＣのリスク水準を確定する解析である。リスク評価は、リスク水準に応じて、定性的か、あるいは定量的かのどちらかである。リスク評価での目標は、そのリスク水準が許容できるリスク水準よりも低いか、あるいは等しいかを決定することである。

9.2 安全性要求仕様

安全性要求仕様は、次のように分割される。

- 機能要求仕様（FRS: Functional Requirements Specification）
- 安全性インテグリティ要求仕様（SIRS）

“機能要求仕様”では、安全関連制御システムが遂行しなければならない“安全機能”を確定する。例えば、熱シャットダウン用の機能別安全性要件（functional safety requirement）は、下記の通りかもしれない。

if TEMP > HIGH SET **then** OPEN VENT and CLOSE FEED

“安全性インテグリティ要求仕様”では、安全機能を実行するのに必要な“安全性インテグリティ”を達成するための要件を確定する。安全性インテグリティ要件では、安全機能が実行される確実度を確定することをもくろんでいる。

安全性インテグリティ要求仕様では、次に示す3つのシステム安全原則に関する要件を確定する。

・システム構成
・安全関連ハードウェアの信頼度
・系統的インテグリティ

9.3 ＳＲＳの特定・指名

本暫定規格では、次に示す、“システム”の3つの水準を定義している。

・システムの全構成（TC）
・ＳＲＳの全構成（TC）
・ＳＲＳ

“システムの全構成（TC）”は、ＥＵＣの制御、保護、あるいは監視用の目的でシステムの階層を構成する構成要素すべてからなる。“ＳＲＳの全構成(TC)”は、安全性インテグリティの必要な水準を一緒に達成するすべてのＳＲＳからなる。そして、“ＳＲＳ”は、要求された安全機能の実行に必要とする安全性インテグリティ水準を自分自身で、あるいは他のＳＲＳを利用して達成するシステムからなる。

これら（ＳＲＳ）は、全体の安全性を達成する責任を有し、安全性ライフサイクルの残りの工程を適用する該当システムであるので、安全関連であるこれらのシステムを分離、特定（同定）、そして指名する（designate）ことは重要である。図4には、ＳＲＳの概念と、リスクを許容できるリスク水準まで軽減する際のＳＲＳの役割りについて示されている。

9.4 設計および構築

安全関連ＰＥＳの設計および構築に関しては、従来のシステムのそれと同じである。しかし、本暫定規格では、安全関連ＰＥＳに対して明確に要求される設計工程に関する、次に示す２つの特定の要件を要求している。

・設計および構築は、ＩＳＯ9001のような制定された品質規格に合致して遂行されること。

・安全性検証は、設計および構築工程と連動して遂行されなければならない。

安全性検証は、設計および構築工程の各発展段階で、（安全性要求仕様工程から開始して）当該発展段階から次の発展段階へと忠実に変換されたかの比較である。

9.5 安全性の妥当性検査

妥当性検査は、最終の運転可能なシステムと元々の安全性要求仕様との適合水準を決定する工程である。妥当性検査は、実行される包括的なテストと、適用される数学的で系統的な解析からなる。

9.6 運転および保守

本暫定規格での運転および保守に関する要件は、安全性能の監視と、設計変更に対する制御の維持とからなる。

9.7 システムの改造と退役

これらの分野に関しては、現行の本規格では扱われておらず、将来の作業での課題であると考えられている。

１０．今後の進め方

本暫定規格の開発の最終工程は、第１分冊“共通の要件”を完結させ、そして、将来発行される暫定版の各分冊の開発に着手する工程である。第１分冊の完結は、特定用途向け規格を開発する責任を有する人達が、第１分冊に述べられている要件をかれら自身の規格へ収録するために、極めて重要である。各国の国内規格委員会（NSC）から諸々の意見を受け取った後で、第１分冊は受理した意見を考慮に入れて完結できることが望ましい。第１分冊が発行された国際規格になる前に、少なくとも１回は、改定された第１分冊が意見を求めるために配布されることを計画している。第１分冊に関して、解決すべき残っている主要な課題は、以下の通りである。

①本規格の範囲：第１分冊と後続の各分冊では、これらのシステムが基盤としている技術をとわず、安全関連制御システムのすべての型を網羅すべきなのか、あるいは、本規格は、徹底的なテストが不可能であるようなプログラマブル電子デバイス、ＰＥＳ群、そして他の複雑な電子デバイスと一体となったＳＲＳに限定すべきなのか。

②インテグリティ水準：システムのインテグリティ水準の水準数を完結させることが必要である。

③インテグリティ水準の総和法：個別のＳＲＳのインテグリティ水準を総和してＳＲＣの全構成（TC）のインテグリティ水準を求める規則を開発しなければならない。

④システム安全原則＝システム構成：システム全体の設計に対して“頑健性”を追加することをもくろんでいるこのシステム安全原則に対する要件が完結しなければならない。

⑤安全性ライフサイクル：注意を注ぐべき重要な開発作業構成要素の順序が実施順に正しく並んでいるか、あるいは、安全性要求仕様やＳＲＳの特定・指名に関係して要求される修正があるか。

⑥システム-ソフトウェア・インタフェース：安全関連ソフトウェア用の暫定規格は、ＳＲＳ用の暫定規格との明確なインタフェースを持たなければならない。この要件が満足されていることを保証するのに、さらなる検討が必要である。

最後に、本暫定規格に関して、対象とする読者（即ち、特定用途向け規格の開発に参画する人達）から感想や意見をもらうことが、第１分冊の将来の有効性にとって最も重要である。この意見聴取は、本規格が取り扱わなければならない広く多様なシステムを網羅する点について、最終版の本規格を十分に頑健にし、そして、規格作成者や本規格を利用する人達にとって利用可能にするために必要である。まだ解決されていない多くの挑戦すべき課題があるし、重大な安全面や経済面での利点を有するすべてのアプリケーション領域に関係する標準的な枠組みを開発する機会が存在する。そのような機会を逃がすべきではない。

参考文献

[1] HSE Guidelines on Programmable Electronic Systems in Safety-Related Applications: "1 An Introductory Guide," ISBN 0 11 883913 6, "2 General Technical Guidelines," ISBN 0 11 883906 3, HMSO(Her Majesty's Stationary Office), PO Box 276, London SW8 5DT.

[2] DIN V VDE 0801: Principles for Computers in Safety-Related Systems (Grundsatze fur Rechner in System mit Sicherheitsaufgaben).

[3] Meffert, K., "Principles for Computers in Safety-Related Systems," ACOS Workshop on Safety-Related Control Systems(An International Standards Workshop on Programmable Electronic Systems for Use in Safety-Related Systems), IEE, Savoy Place, London, 8-9 March 1990.

[4] Clark, B.J. and Ward, G.R., "The Application of Programmable Electronics Systems to the Control of Machinery: The Scene in Europe," ACOS Workshop on Safety-Related Control Systems (An International Standards Workshop on Programmable Electronic Systems for Use in Safety-Related Systems), IEE, Savoy Place, London, 8-9 March 1990.

[5] Lagana, T., "Guidelines for Safe Automation of Chemical Processes," ACOS Workshop on Safety-Related Control Systems(An International Standards Workshop on Programmable Electronic Systems for Use in Safety-Related Systems), IEE, Savoy Place, London, 8-9 March 1990.

[6] IEC Draft Standard: "Functional Safety of Programmable Electronic Systems: Generic Aspects; Part 1:General Requirements," IEC Reference:65A(Secretariat)96, Sept. 1989.

[7] IEC Draft Standard: "Software for Computers in the Application of Industrial Safety-Related Systems," IEC Reference:65A(Secretariat) 94, Aug. 1989.

Ⅱ. 信号保安システムに関する相互受入れ

細　目　次

1．はじめに

2．論題の選択

3．技術委員会

4．国際的な枠組

5．法律的な側面

6．信号保安システムの相互受入れ
　6．1　要　件
　6．2　規　則
　6．3　安全性の度合
　6．4　リスクのレベル

7．安全性の証明

8．勧　告

9．結　論

10．今後の本委員会

参考文献

信号保安システムに関する相互受入れ
(Cross-Acceptance of Vital Signalling Systems)

著　者：E.O. Goddard(London Underground Limited)
C.H. Zufferey(SBB/CFF/FFS Switzerland)
出　典：Report by the Technical Committee, IRSE
(The Institution of Railway Signal Engineers),
March 1992, pp.10

1. はじめに

1990年のJacques Catrainの基調演説にて提案され、続く1990年9月11日の会議で承認された技術委員会は英国のＩＲＳＥ(The Institution of Railway Signal Engineers)内に設置され、鉄道信号工業会(Railway Signalling Profession)より委託されたテーマについて、1991／92年の会期末までにレポートをまとめた。

ここでは、そのレポートに沿い、作業内容と本委員会の結論およびその背景について述べる。

2. 論題の選択

1回目の会合では、多くの話題について議論された。しかし、すぐに、すべての信号技術者にとっての関心の的は、信号システムへの新技術導入の難しさであり、特に、それぞれのアプリケーションにおける安全性の証明に必要な手段とコストであることが判明した。

現在多くの努力が、ほとんどすべてのアプリケーションに対して、異なる形式でそしてしばしば異なる基準に基づいて安全性の証明に浪費されている。鉄道信号の理論は、それぞれの鉄道会社で、まったく独立して開発され、ひとつひとつの事故やニアミスから教訓が学ばれた。

システムにおける相互の互換性の実現を可能にする、共通の基準を採用することにより、欧州鉄道網を作り上げるため、資源を最大限に利用し、資金の有効な活用方法を生み出すことができる。

3. 技術委員会

本委員会は、欧州の鉄道関連の行政機関と、鉄道業者からの16名の上級技術者から成っている（付録1参照）。

我々は、新技術の導入や最新の安全性解析技術の適用から生じる、似たような問題を対象とする、多くの類似の委員会があるという認識から、レポートを作成した。

それ故、我々の主要な目的は、それらの委員会に影響を与え、最終的にはその作業結果を欧州鉄道共同体ＣＥＲ（European Railway Community）のワーキンググループに示すことであった。

4. 国際的な枠組

本委員会では、既に存在している各種の委員会を一つ一つ明確にしていくことにかなりの時間を費やした。図1には、欧州内で鉄道保安システムに従事しているいろいろなグループの相互の作業関連図を描いた。本委員会のメンバーは、既にそれらのたくさんのグループに、参加していたり、代表していた。そのため彼らの最新の成果を議論に持ち込んだり、他のグループの議論に寄与することができた。

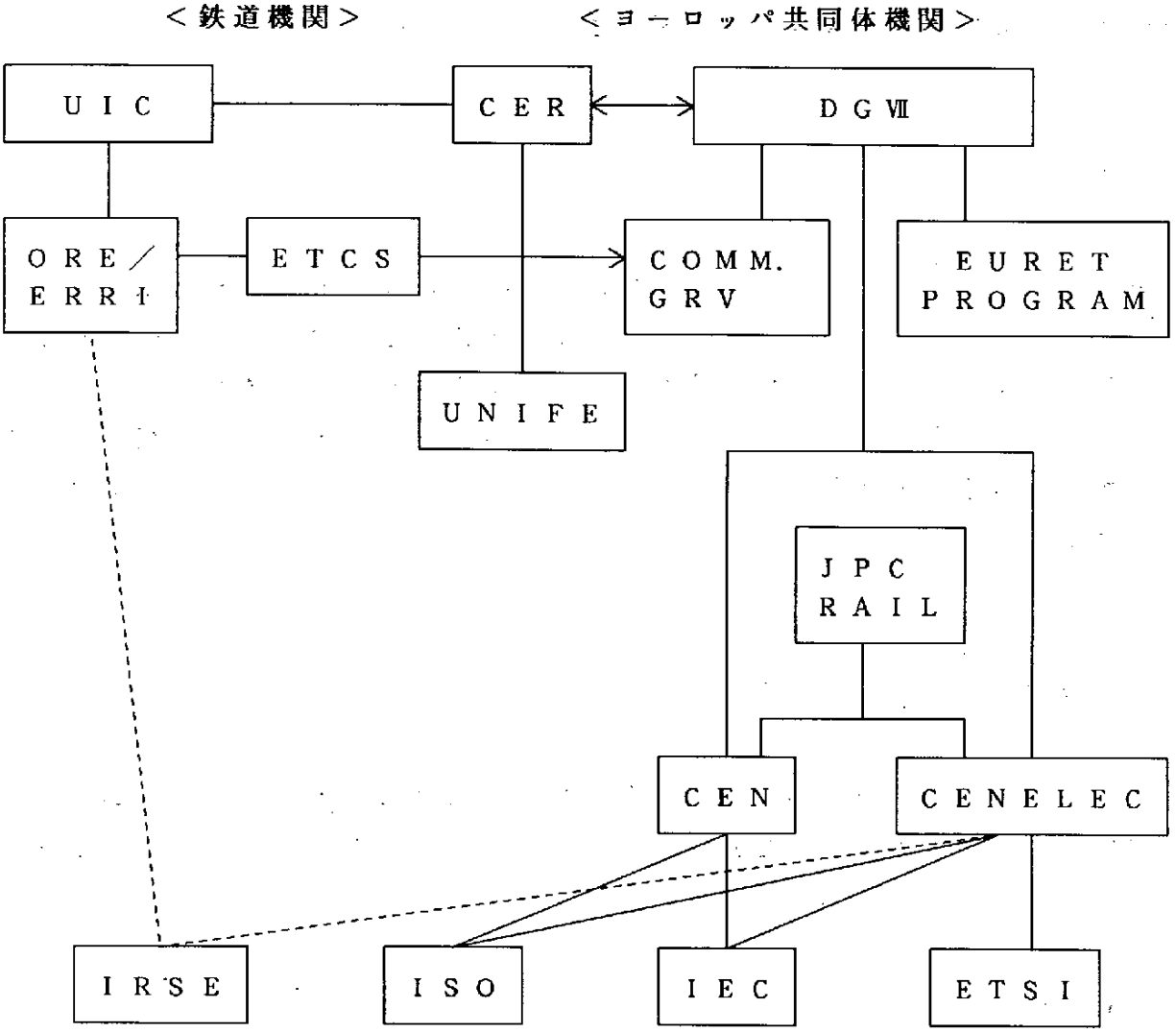
多くの基準がすでに存在しているが、どれも信号保安システムに対するすべての要求事項をカバーしているものはない。しかしながら、以下のものは、将来の基準を確立していく上で優れた土台になると考えられる。

(1) IEC/SC65A/WG9
産業用安全関連システムの分野で利用されるコンピュータ用ソフトウエア

(2) IEC/SC65A/WG10
プログラマブル電子システムの機能別安全性：汎用的な側面

(3) UIC738R(1990)
安全関連情報の処理と伝送

さらに、VDE 0831（Verband Deutscher Electrotechniker;ドイツ電気技術組合）やRIA 23（イギリスの規格）のような多くの国内規格が存在する。



CEN : Comité Européen de Normalisation
(European Committee for Standardisation)
CENELEC : Comité Européen de Normalisation Electrotechnique
(European Committee for Electrotechnical Standardisation)
CER : European Railway Community
COMM GRV: Commission Grade Vitesse
DG Ⅶ : Directorate-General Ⅶ
ETSI : European Telecomms Standardisation Institute
IRSE : The Institution of Railway Signal Engineers
ORE : Organisation of European Railways
UIC : Union Internationale des Chemins de Fer
(International Union of Roadways)
CEN/CENELECは強制的規制分野以外の欧州規格の制定機関である。

図１　国際規格化／標準化の概観

IEC/SC65Aの文書は原案としてのみ有効であるが、本委員会が作業を進めるのに有効に作られていた。英国規格も参照のこと。

5. 法律的な側面

レポートを作成するにあたり、本委員会では、考慮する必要のある法律的な側面があるかどうか考察した。そして、それぞれの国で示されている法律の枠組みは、本質的に同じで、共通の解決法が提案できるという結論に達した。

鉄道当局に適用される民事責任と、個人に適用される刑事責任には相違がある。安全システムの認定の基準はそれぞれの国で異なっており、レポートの中で要約されている。

6. 信号保安システムの相互受入れ

相互受入れ（Cross-acceptance）を可能にするために、本委員会ではいかなる信号装置の設計にも当てはまるたくさんの基本的な必要条件があることを提案した。

6.1 要　件

①信号装置の耐用期間を通して、緊急処置をしなければいけないときの鉄道それ自身の安全も含めた総合的なシステムの安全性を考慮すること。

②信号システムのいかなる部分の機能についても、要求される安全度を決定すること。

③要求される安全度のそれぞれに対して最小限の基準を確立すること。

④安全性は、装置の全ライフサイクルにおいて考慮されることによってのみ達成される（図2参照）。

⑤安全性は、仕様の決定から製造、設置そして保守および改修を含むすべての工程における厳密な管理を通してのみ、保証することができる。

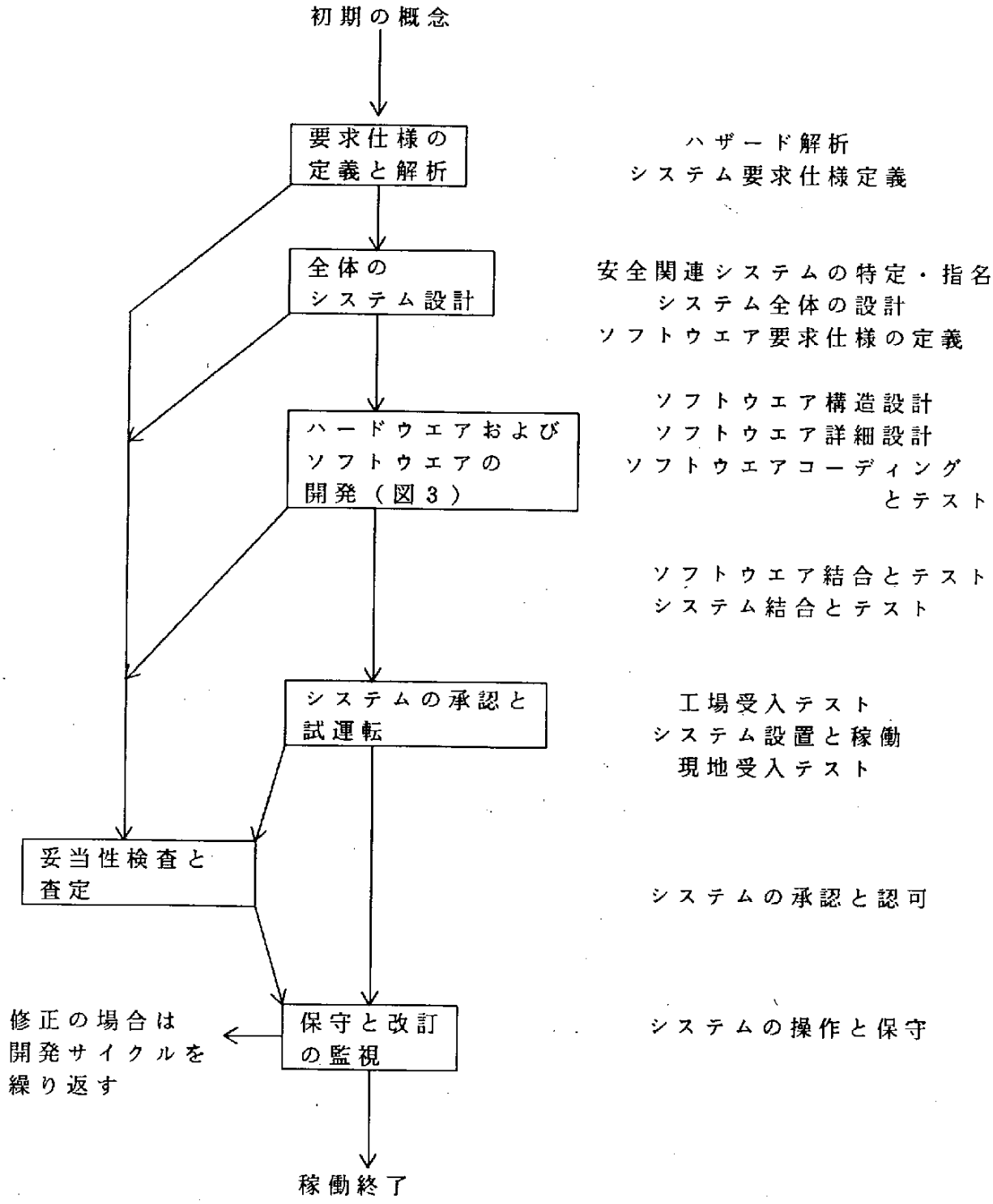


図2　システムのライフサイクル

6.2 規　則

上記の要件が達成されることを確実にするために、以下の規則に従わなければならない。

①新しく導入されるいかなるシステムも少なくともそれがとって代わるシステムと同等以上に安全でなければならない。

②システムのライフサイクルのすべての局面で、形式化され、よく記述された工程が実施されなければならない。

③システムは使用される周囲環境が、規定された範囲を越えても安全な動作が保証されるように設計されていること。

④単一の故障では、不安全状態となってはならない。

⑤単独の故障では不安全状態にはならないが、第二のもしくは続いて発生した故障との組み合わせで不安全状態となってしまう装置のあらゆる潜在故障が、発見され、無効にされることを確実にしなければならない。そのようなアクションは、要求された平均危険側故障時間間隔MTBWSF（Mean Time Between Wrong Side Failures）の値を満たす時間内で行われなければならない。

6.3 安全性の度合

安全システムに対する本質的な要求事項は、すべてのシステムが安全上同じ強度を持っておらず、それ故に、システムの設計にはそのシステムから要求される安全性の度合を反映することが必要であるということを認めることである。

IEC/SC65A/WG9のレポートは、システムが要求する安全の段階付けを設定するのに非常によい拠り所となると本委員会が考える、次に示す安全性インテグリティレベルの範囲を規定している。

レベル	要求される安全性インテグリティ
4	非常に高度
3	高度
2	中度
1	低度
0	安全に無関係

これに基づき、本委員会は鉄道信号システムに要求される安全度は、次に示すように分類できるという結論を得た。

度　合	機　　能
4	列車の衝突および脱線を防ぐ
3	列車運行を確定する
2	鉄道運行の管理を行う
1	乗客に情報を流す
0	情報システムの運用

6．4　リスクのレベル

伝統的に鉄道技術者は完全な安全を追い求め、鉄道の安全性レベルは絶えず他の形態の交通機関のそれよりも上であった。この伝統的なフェイルセーフの基準は信号保安システムのいかなる構成要素やサブシステムも故障モードが知られており、単独の故障では危険側の状態が発生しないように設計されているということを要求している。

今日では、信号技術者は装置の故障に対して、列車を止めることがもはや適切で安全な対応ではないということを認識すべきである。この新しい取り組みでは、装置やソフトウエアの起こり得るいかなる故障モードも考慮されていること、システムが、いかなる故障も検出され、規定された安全性のレベルに対して不安全な結果を引き起こすことを防ぐように設計されていることを要求している。

そのようなシステムの安全性レベルは、予測される（もしくは実際の）MTBWSFにより決定される。リスクの許容レベルは、先に定義した安全性レベルに依存する。

もう一つは、長年の慣行で認められた規則集を装置のライフサイクルに対して規定することは可能である。これは、そのような安全システムが定義され、設計され、制作され、取り付けられ、保守される方法を左右する規則集である。規則

類の厳格さの度合が非経済的、あるいは実施不可能なほど高くはならないという条件で、特定のアプリケーションにおいては、規則類の厳格さの度合を該当システムの安全度に一致させるだけで十分である。

7．安全性の証明

システムのライフサイクルにおけるすべての工程に関しては品質保証基準BS5750およびEN 29001に従って詳細に文書に記録されなければならない。システムの仕様決定、設計、試験および試運転の間に作られたすべての文書が安全性の証明を形成する。

本委員会は安全性の証明の作成で、以下の基本方針が採用されるべきであるということを承認した。

①安全性の証明は契約（施工）者により供給されること。

②安全性の証明は、開発サイクルのそれぞれのステージの製品がその段階での要求事項に従っていることの検証（Verification）と、供給される完成したシステムが該当要求事項に従っていることの妥当性検査（Validation）の２つの部分からなる（図３参照）。

③いくつかの方法と手順は安全性の証明を確立できるようにするために必須である。

④鉄道管理者は、監査やその他の方法を通して安全性の証明を確認すること。

⑤国際的な基準が採用されること。

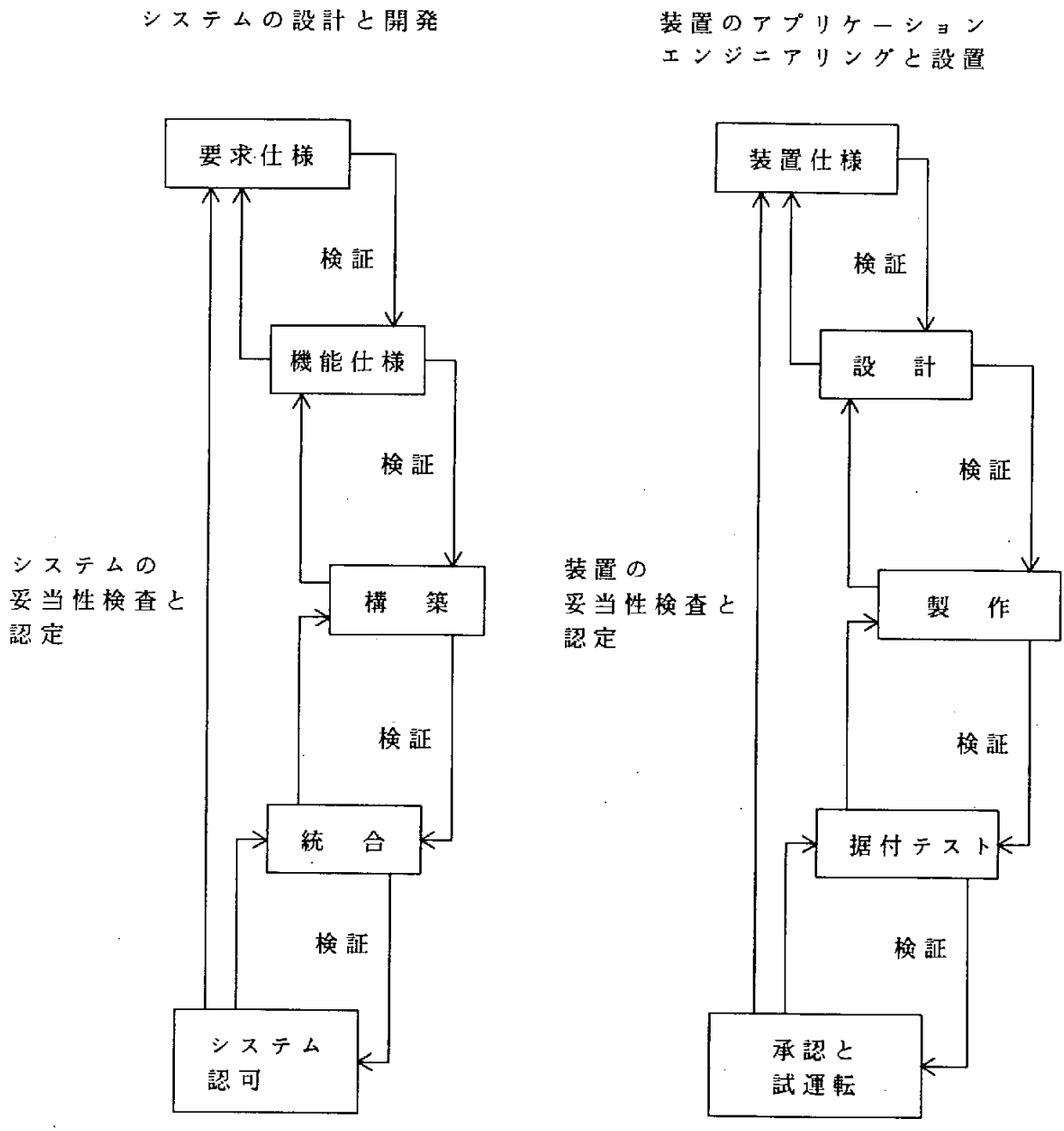


図３　開発とアプリケーションの期間中での検証と妥当性検査の工程

8. 勧　告

本委員会の主要な勧告は、鉄道事業全体の背景の中で新しいシステムが考えられるような信頼性、アベイラビリティ、保守性、安全性、セキュリティおよびコストを含む総合的な鉄道システム戦略が要求されるということである。

鉄道の構内に入ってから出るまでの乗客および乗務員の安全は、鉄道会社の責任である。この責任には、故障状態における安全の確保も含まれる。

本委員会の特定の勧告は、以下の通りである。

①鉄道会社は、プロジェクトの開始の時点でシステムに対する機能要求事項を明確にしなければならない。－要求仕様書

システムの詳細仕様は、要求仕様書に対応して契約（施工）者から提出され、全体として鉄道での特定のアプリケーションの安全およびセキュリティ環境に従って鉄道会社により完全に承認されなければならない。

②鉄道会社および契約（施工）者は、例えばIEC/SC65Aの文書のように各フェーズに対して規定している国際的な基準（リスク分析、設計、構築、試験、据え付け、試運転および保守に関する要求事項）に従うこと。

それぞれのフェーズは、文書管理ツールにて準備された文書により示される。それらの各フェーズは、契約（施工）者により検証され、妥当性検査計画に従ってプロジェクトの開始の時点からその寿命の終わりまで適用される正式な手順を通して妥当性が確認されること。

③一つの鉄道会社から他の鉄道会社へのシステムの移植を容易にするために、鉄道会社はシステムの妥当性検査の手順、検査された装置および／またはソフトウエアのバージョンおよびそれらのバージョンが区別できる方法を明らかに定義すべきであるということを強く勧告する。認証機関（監督官庁）は、再認証なしに移植可能な変更の範囲を定義すべきであるということもまた勧告する。

④システムソフトウエアとアプリケーションソフトウエア、プログラムとデータの間を明らかに分離すること。

⑤それぞれのシステムについて、どの部分が重要で、外部からの妨害や変更に対していかにして守られているかが、明細に述べられていること。システムのど

の部分が、システム構成に対する変更の影響を受けやすいデータを含んでいるかもまた示されること。

⑥安全システムの構成および設計は、自動ツール（CASE）を使うことによりサポートされること。また、特定のアプリケーションに対する安全システムの構成作業は、信号技術者によって使用可能な自動ツールの使用によってサポートされること。

⑦安全信号システムは、それらの構成作業および保守が契約（施工）者および鉄道会社の職員のどちらによってでも実施することができるように設計されていること。

⑧安全度0のシステムは、特別の注意を要求しないが、少なくともEN 29001で要求される品質保証の基準を満たし、会社の品質保証手順に合致すること。（このレポートの目的から、それらは安全システムであるとみなされない。よって上記の勧告は確立された良い慣行を構成するところを除いては適用されない。）

⑨安全度1および2のシステムは、それらが妥当性検査を要求しない場合を除いて、上記勧告の最初の7項目に従うこと。しかしながら、ハザード解析は要求されるし、バックアップ手順も確立されなければならない。

⑩安全度3および4のシステムは上記勧告の最初の7項目に加え、さらに以下のことに従うこと。

a. ハードウエアとアプリケーションソフトウエアの両方の妥当性が検査されること。

b. 十分な多様性がない場合には、システムソフトウエアもまた妥当性が検査されること。

c. 自動バージョン識別手段を使うことによって厳密なバージョン管理が採用されること。

d. 入出力は常に限定的な状態をとるようにするか、冗長化手段により要求される安全性レベルを満たすように設計されるべきである。

e. 特定のハードウエアとソフトウエアの詳細設計は、システムの妥当性を検査し、運用し、保守するために鉄道会社に公開されなければならない。

f. すべてのオンラインテスト用ソフトウエアは妥当性を検査されること。

g. 構造化高水準言語でプログラミングするとともに、形式的仕様記述言語の使用についても考慮を払うこと。それにより安全性面の数学的証明が可能となる。

9. 結　論

近代化された信号システムの生産は、従来の生産および保守に対する伝統的な慣行を見直すことを引き起こした。この発展について、ＩＲＳＥによって多くの文書が既に発行されており、本委員会はこの先行作業を我々のレポートに吸収できることを希望する。

信号技術者の役割もまた変わってきており、今日では、以前は鉄道の重要システムの一部分であると考えていなかった顧客安全での多くの局面（例えば拡声装置(Public Address Systems)や列車無線システム）を包含するようになっている。そのようなシステムの安全への寄与は否定できないが、列車の衝突を防ぐために規定されたそれらと同等の安全性レベルを達成することが要求されていないこともまた明白である。それ故に、システムはその機能に従った安全のいろいろな階級を手にいれるために仕様決定されなければならない。

安全性の証明は本質的な活動ではあるが、信号システムの製造コストを相当に上げることがあり得るマンパワーを要求する活動である。すべての鉄道当局が共通の基準類や規定類を利用するという条件で、あらゆるシステムの安全性を自分自身で証明することが本質ではなく、特定のアプリケーションが安全であるということを証明することのみが必要であるべきである。

現在ヨーロッパの各種の委員会によって実施されている作業は、共通基準に対する要求事項を満たす基準の作成へと導いている。特にIEC/SC65Aの作業は安全性の証明に関する基準を採用し、それ故に相互受入れを生じさせることを当然導くので推薦される。

本委員会は、その基準の信号技術者に対する解釈を確立し、他のヨーロッパの委員会の作業に、彼らの議論の形成段階で影響を与えることを可能としてきた。

１０．今後の本委員会

最後に本委員会自身の活動とその将来について言及しなければ本報告は不完全であろう。

おそらく記録されるべきもっとも注目に値することは、ヨーロッパの鉄道当局および契約（施工）者からの上級技術者からなる本委員会がほとんど欠席なしに10回の一連の会合を持てたということである。会合はレポートの作成に対し、非常に厳しい期限のおかげで能率的なやり方で開催された。それにも関わらず、すべてのメンバーは、その経験を楽しみ、有益なものとし、たくさんの新しい交流が確立された。

本委員会の継続が評議会により承認され、本委員会の委員長および幹事が毎年持ち回りで交代することが合意された。来年度の委員長はCharles Zufferey、幹事はPeter Stanleyとなるであろう。

いくつかの議論の後で、来年の議題として“The Design of Signalling Systems for Graceful Degradation”が選ばれ、1992年2月14日に承認された。

我々全員は、Jacques Catrainによって始められたこの作業が継続し、この方法により専門家の視点が確立され、その視点が適切な地域に広まることができることを希望する。

参考文献

BS89/33006DC(secretariat)
British Standard published version of IEC/SC65A/WG9.

BS89/33005DC(secretariat)
British Standard published version of IEC/SC65A/WG10.

RIA Technical Specification 23
Safety Related Software for Railway Signalling.

付録1．ＩＲＳＥ技術委員会のメンバーリスト（1990/91）

オーストリア	Helmut Steindl	Chief Signal Engineer	OeBB
ベルギー	Freddy De Vilder	Senior Signal Engineer	SNCB/NMBS
	Pierre Mertens	Director	ACEC Transport
フランス	Jean-Paul Guilloux	Chief Signal Engineer	SNCF
	Jacques Catrain	Director	GEC Alsthom
	Gerard Guiho	Director	GEC Alsthom
ドイツ	Albert Bidinger	Chief Signal Engineer	DB
	Horst Bimmermann	Director	Siemens
イタリア	Giuliano Cerullo	Chief Signal Engineer	FS
	Giacomo Astengo	Director	Ansaldo
オランダ	Peter F Otten	Head Electro Systems	NS
スイス	Charles H Zufferey	Chief Signal Engineer	SBB/CFF/FFS
	Adrian Exer	Director	Integra Signum
イギリス	Peter W Stanley	Assistant Director Signalling	BR
	John Mills	Director	Westinghouse Signals
	Eddie Goddard	The Signal and Control Systems Engineer	LUL(London Underground Limited)
委員長	Eddie Goddard		LUL
幹　事	Charles H Zufferey		SBB/CFF/FFS

本委員会の作業は1990年9月21日から1992年1月14日までの10回の会合で完了した。

付録２．手順の一覧表(1/2)

ハードウエア用の方法もしくは手順	M	HR	R
1 故障モード、効果と危険解析(FMECA)	×		
2 故障ツリー解析(FTA)		×	
3 共通モード故障解析(CMFA)	×		
4 異なるチームによる設計と検証	×		
5 全負荷試験	×		
6 機能テスト	×		
7 ホワイトボックステスト		×	
8 フリーテスト － “what if?” 法(a)	×	×	
9 シミュレーション			×
10 仕様に対する静的準拠性	×		
11 仕様に対する動的準拠性		×	
12 MTBWSFの計算（危険側故障）(b)	×	×	
13 ORE故障一覧(c)		×	
14 残余リスクの計算についての一覧表(d)			×
15 実用機／試作機の使用前の現地での試用		×	
16 品質保証要求事項 EN 29001	×		
17 ドキュメンテーションに関する定められた規則	×		

(a) 必須でなければ、５項とともに実施される。

(b) 要求された場合は、評価の目的に対して必須

(c) 列挙されていない構成要素に対しては、個々のドキュメントにより補われる。

(d) 安全回路に使われる過大値を決定する。

M＝必須　　HR＝強く勧告する　　R＝勧告された有効な方法

付録２．手順の一覧表(2/2)

ソフトウエア用の方法もしくは手順	M	HR	R
1 ソフトウエアエラー効果分析	×		
2 静的ソフトウエア分析		×	
3 動的ソフトウエア分析		×	
4 第三者によるコードインスペクション		×	
5 数学的証明付の形式的仕様		×	
6 妥当性が検証されているコンパイラ		×	
7 高水準言語		×	
8 目標のハードウエア上でテストされた機械コード	×		
9 異なるチームによる設計と検証	×		
10 プログラム中のすべての分岐を通過するテスト	×		
11 ホワイトボックステスト		×	
12 機能テスト	×		
13 仕様に対する静的準拠性	×		
14 仕様に対する動的準拠性		×	
15 防衛的プログラミング			×
16 構造化プログラミング規則	×		
17 実用機／試作機の使用前の現地での試用		×	
18 品質保証要求要求事項 EN 29001	×		
19 ドキュメンテーションに関する定められた規則	×		

システム用の方法もしくは手順	M	HR	R
1 ハザード解析レビュー	×		
2 故障ツリー解析(FTA)		×	
3 異なるチームによる設計と検証	×		
4 機能テスト	×		
5 シミュレーション		×	
6 仕様に対する静的準拠性	×		
7 MTBWSFの計算（危険側故障）		×	
8 実用機／試作機の使用前の現地での試用	×		
9 品質保証要求事項 EN 29001	×		
10 ドキュメンテーションに関する定められた規則	×		

M＝必須　　HR＝強く勧告する　　R＝勧告された有効な方法

Ⅲ. セーフティ・クリティカル・コンピュータ・ソフトウェア用の英国防規格の開発の根本的理由

細　目　次

主　旨

1．はじめに

2．セーフティ・クリティカル・コンピュータ・ソフトウェアの特徴

3．デジタル・コンピュータが提起する安全挑戦課題

4．セーフティ・クリティカル・ソフトウェアに関する英国防省方針の変遷

5．新しい方針指令の必要性

6．今後のセーフティ・クリティカル・システムに関する方針の基本的な特徴

7．ＭＯＤ規格の方針

8．暫定版の英国規格のハイライト

9．セーフティ・クリティカル・ソフトウェアでのＡＤＡの使用

10．今後の作業プログラム

11．おわりに

参考文献

セーフティ・クリティカル・コンピュータ・ソフトウェア用の英国防規格の開発の根本的理由

(Rationale for the Development of the UK Defence Standards for Safety-Critical Computer Software)

著　者：Michael J D Brown
(Ministry of Defence Procurement Executive, UK)

出　典：COMPASS'90, June 1990, pp.144-150

［主　旨］

ここ25年間、組み込み型コンピュータ・システムが、英空軍が操作する軍事装置中に存在するセーフティ・クリティカル機能を制御するのに使用されている。そのようなシステムの開発には国防省の安全検定手順が適用されており、その実績が、国防省が正当な誇りを持つている安全記録となっている。しかし、調達環境の変化や、技術革新は、今後2･3年以内に新しい開発検定手順を採用することを必要とする安全性に対して、数々の挑戦課題を提起している。本論文では、国防省が新しい規格を正式に採用する前に、公開討論会（public debate）に提出される新しい国防規格案（draft standard）を作成する際に影響を与えたこれらの挑戦課題について述べる。

1．はじめに

最近、コンピュータ制御システムを高水準のインテグリティが要求される用途に利用する際のソフトウェア工学に関する規格に対する大幅な改定が必要となっていることが明確になってきた。学界、産業界、そして政府機関の関心は、一般大衆層の関心と同じであり、その共通的な関心の的は、セーフティ・クリティカル用途である。COMPASS会議（Computer Assurance Conference on Computer Systems Integrity）のようなイベントの件数の増加は、専門家の関心の高さを反映している。一方では、国際技術規格や規制上の枠組み（regulatory framework）の開発に向けて、作業がある程度進んでいる。

英国の国防省（MOD: Ministry of Defence）は、セーフティ・クリティカル・コンピュータ用途の開発および検定に限定した規格案（draft standard）を発行した第１番目の政府機関の一つである。1988年でのより小規模な配布に続いて、1989年５月には、国防規格00-55および00-56（DEF-STAN 00-55、DEF-STAN 00-56）の暫定版（interim）のコピー数千部が、英国のセーフティ・クリティカル・ソフトウェア分野で活動しているとして知られているすべての人達や、ＭＯＤ規格の開発の過程での正規の一環である公の諮問（public consultation）への参加に興味を示した人達に配布された。学界、産業界、そして政府機関からの代表者約300名が、1989年６月にMalvernにある英国防省ＲＳＲＥ（Royal Signals and Radar Research Establishment；王立通信レーダー研究所）で開催された機密扱いではない２日間の討論集会に参加し、そして、その後に、個人または組織からの意見書、約1370件が受理された。

ＭＯＤは、軍事目的に使用される装置の安全性に関して、次に示すように、非常に広範囲の責任を負わなければならない。

①利用者として、ＭＯＤは、装置の操作や保守を担当している要員が死や損傷にさらされないことを、合理的に実施可能な（reasonably practicable）範囲内で、確実にすることを規定している英国の労働衛生安全法（UK Health and Safety at Work Act）（1974年）に従う法的な責任を有する。

②装置の所有者として、ＭＯＤは、事故による装置の破壊や損害による、経済的および軍事的な面での損失を最小にすることに関心がある。また、所有者として、ＭＯＤは、装置操作中に発生する、非政府の要員や財産に対して損害を与えるような事故を予防し、そして、その事故によって損害を与えた場合には、補償金を支払う法的な義務を有している。

③軍事装置が設計要件および操作要件を満たしているかを確かめる責任を有する規制機関（regulatory agency）として、ＭＯＤは公共の利益に対して公の義務を果している。事故が発生した際には、その事故調査結果に基づいて、安全検定過程自体を厳格な審査（scrutiny）を受けさせるべきであるかどうかを検討することが要諦である。

これらの分野での責任の各々が、セーフティ・クリティカル・コンピュータ・ソフトウェア用の英国規格案に重大な影響を及ぼした。本規格では、セーフティ・クリティカル・ソフトウェアの仕様作成および評価に形式的方法を使用することを条件にしている点が、この分野で重要な役割りを果たした、と最も明確に結論付けることができるかもしれない。他の提案された諸規格では、そのような確固たる要件を含めることを考えていない。ＭＯＤでは、すべての国のすべての用途で、そして、あらゆる目的で、この技術的アプローチが必ず最適であると断言していないし、英国防省の調達要件や法規で規定された要件に適合させる目的で、この技術的アプローチが故意に立案されたとも一切言っていない。とは言っても、同種の規格を作成することに関与する他の政府機関、あるいは規制機関にとって、同じような考慮が重要になると思われるので、ＭＯＤの規格案は役に立つモデルとして利用されるかもしれない。

2．セーフティ・クリティカル・コンピュータ・ソフトウェアの特徴

約25年間、デジタル・コンピュータが、英国の軍事システム中に存在するセーフティ・クリティカル機能を実行する構成要素として使用されている。本稿での当面の目的おいては、セーフティ・クリティカル・システムとは、「コンピュータが死あるいは損傷の原因となる可能性のあるエネルギーの放出あるいは指令を直接制御するようなシステムで、かつ、コンピュータ機能が非常に深く組み込まれ、あるいは、非常に迅速に操作しなければならず、人間による効果的な監視が不可能であるようなシステムである」と定義される。この定義は、兵器の装着および信管の取り付け（ordnance arming and fuzing）、本来的に不安定な航空機の飛行制御システム、あるいは高速道路の交通信号制御システムをも含めた多様な用途を含むほど十分に広範で、しかし、死あるいは損傷がシステムの機能不全（malfunction）の間接的な影響の結果として起こるような使い方で使用されているシステムを除くように意図されている。Benjamin Franklin は、Ｕ字形くぎ（horseshoe nail）が不足したせいで、国が亡びたと言った時に、その難題を非常に手際よく提起したが、Ｕ字形くぎに対する、すべての可能性のある使用および誤用（misuse）をすべて含む意味のある規格を規定することは不可能である。それ故に、多くの討論の後で、ＭＯＤでは、直接的なリスクに鋭く焦点を当てた規格の方が、直接的なリスクおよび間接的なリスクの両方を網羅することを試みる必要以上に一般化された予防措置ではなくて、全面的な安全保護を与えるので、より望ましいとして決定した。

英国防省のシステムでは、最初は、通常、セーフティ・クリティカル・コンピュータ用途は、伝統的な機械式あるいはアナログ電子式の構成要素との技術転換という形式で出現した。それ故に、それらは特定用途領域に関連する規制上の枠組み内で取り扱われていたことは、驚くべきことではない。このアプローチは、新しい技術を最大限に活用する一番の近道を提示しないかもしれないが、各用途分野での安全問題を取り扱う蓄積された経験との直接リンクの保存は、各後継システム（successive system）が導入されるにつれて、設計に起因する故障のリスクにさらされる度合を最小化するのに役に立つ。安全性は、当然のこととして、決しておろそかにすることはできないが、ＭＯＤの法規および手順の下で開発されたセーフティ・クリティカル・コンピュータ・システムでの設計に起因する故障の直接的な結果として誰も殺されたり、あるいは損傷したりしていないという歴史的な事実がある。コンピュータを過去においていかなる手段によっても自動化されたことがない新しいプロセスに適用している民間部門では、その安全記録にはいくつかの汚点が記されている。それらの汚点は、広範な大衆を懸念させるほどには、悪くはないが、しかし、経験の蓄積を超えたセーフティ・クリティカル用途にまでコンピュータの利用を広げるのは、より累進的なハザードなもくろみ（business）であることを示すのに足る汚点がある。ＭＯＤでは、その過去の安全記録の正当な誇りを持続しながらも、その将来に対して少しも自己満足しているわけではなく、コンピュータ用途での将来の開発を満足させるために、安全性に対する新しいアプローチの必要性をはっきりと認識している。

3．デジタル・コンピュータが提起する安全挑戦課題

典型的なアナログ電子システムと対比してみると、デジタル・コンピュータは、累進的に性能が低下するよりもむしろ、突発的に完全故障モードを呈する方がよりありそうである。事前の故障警告がなく、かつ、故障後の機械およびデータの状態が予測不可能であるので、信頼できる緊急時の切り放し（dependable emergency disconnection）、あるいはフェール・セーフ措置を考案することをより困難にしている。

ソフトウェアによって実現された計算手続き（calculation）の関数域（functional domain）は必ずしも連続ではないので、それ故に、要求された動作包絡面の境界（the perimeter of the required performance envelope）のまわりのすべての点で安全な出力が得られることを示すだけでは十分ではない。論理フロー

の注意を引かない組み合せは、予期しないはなはだしい誤り（gross error）の出現を導く可能性がある。例えば、２つのデータ項目の値がほとんど同じであるような、普通ではないあるケースにおいて、その２つのデータ項目の差を使用して除算を試みた場合、起こり得る悲惨な算術オーバフロー条件を発生させてしまう。ソフトウェア誤りの原因となる論理条件やデータ条件はランダムで不相関な事象によって引き起こされるものではないので、工学レベルでは、ソフトウェア誤りに対しては、慣習的に使用される確率論的な考えに基づくシステム信頼性評価で解析できるかに関しては非常に疑わしい。

ソフトウェアは完全に同じものを複製することができ、それを使用した結果、摩耗もしなければ、壊れることもないので、すべての機能不全は設計誤りの結果であると言うこともできるし、それ故に、それらは、システム設計者あるいは開発技術者の不作為あるいは作為（omission or commission）の明確な作用の帰結である。“予期できない欠陥”、即ち、“不可抗力”（Acts of God）と通常言われているこれらの故障がないことが、製造業者、所有者、使用者、そして規制機関に対して等しく適用される、起こり得る法律上の重大な結論である。

最後に、コンピュータに対する大衆の態度の中には、無知や恐れに関する強力な要素があることを認めなければならない。ソフトウェア・システムの設計の大半では、悲劇的な結末をもたらす誤りが作り込まれ、そのような誤りは実操作中のみだけでしばしば検出されるという豊富な証拠がある。最近の例では、論理設計中でのかなり小さな誤りであると思われていたことによって、米国の長距離電話システムが摩痩している。ソフトウェアをセーフティ・クリティカル用途に関与させる場合には、安全問題に関して、システム設計、構築、そして検定のすべての発展段階で人を納得させる位に本気で取りかかっていることを理解させることは、明らかに非常に重要である。

4. セーフティ・クリティカル・ソフトウェアに関する英国防省方針の変遷

セーフティ・クリティカル・コンピュータ・システムに関するＭＯＤ方針は、各用途分野に適用される個別の規制上の枠組みから進展していることは、既に指摘した。すべてのそのような方針の共通的な特徴は、調達庁に対し助言をし、各々の新しいシステムの開発に責任を有する産業界の主契約業者を監督するために、ＭＯＤの研究機構に所属する政府の科学者および技術者をその仕事に任命してい

る点である。そして、コンピュータを軍事システムの組み込み構成要素として使用するような分野、即ち、急速に進展する技術の分野では、政府の科学者は、最も新しい理論的技術および実践的技術を研究所から産業界に技術移転させたり、経験によって得た知識の最大限の交流を確保する点で、不可欠で価値のある役割りを果している。

ＭＯＤでは、非常に早い時期に、リアルタイム・コンピュータ・システム・ソフトウェアの構築に共通の高位プログラミング言語を採用することを決定し、1969年には、ＣＯＲＡＬ（米国のＪＯＶＩＡＬと類似の言語）の使用を強制した。この方針の採用動機は、主として、設計の明瞭性に関係し、ＭＯＤが金銭を支払って開発したソフトウェアをさらに一層活用したり、あるいは再利用したりする権利を保障（secure）するのと同時に、ＣＯＲＡＬが提供する設計抽象化の発展度合が、ソフトウェア工学手法の初期の開発の機会を与え、促進させた。1970年から1980年の間に、ＭＯＤのソフトウェア研究プログラムから広範なアイデアや実用ツールが輩出した。多分最も重要な開発成果としては、①有向グラフ理論をソフトウェア構造に適用したこと、②プログラムの制御フローとデータ利用を批判的に精査する技法を導き出したこと、そして、③現在の形式的方法に関する研究の大半に対する基盤を提供したこと、である。

5．新しい方針指令の必要性

ソフトウェア・インテグリティに対する新しい基盤を、次の10年以内に導入する必要があることが、1985年までに明らかになった。多くの要因がＭＯＤの関心の一因となっており、その内の最も重大な関心事は、①政府の役人が産業界でのソフトウェア開発作業を監視することが困難であること、②システムのＶ＆Ｖ（Validation and Verification；妥当性検査および検証）に関する技術的諸問題が発生していること、そして、③爆発性の兵器を装着し、信管を取り付ける作業にソフトウェア制御式処理装置を適用する必要性、である。

進展する英国防省の調達方針は、厳密な開発契約に通じている業者の競争を奨励した。その開発契約の下で、契約業者は、一定の金額内で、装置性能要件やプロジェクト・タイム・スケジュールを満たす責任を果たした。政府の役人が安全監視機能を遂行するのに介入するという構想は、そのような調達環境と全く両立しなかった。同時に、ＭＯＤの研究機構から民間部門の会社へ開発責任を委譲し

たことにより、伝統的な安全監視機能を実行するための適切な資格を有する政府の役人を見つけるのが累進的により困難になってきた。最後に、潜在的にセーフティ・クリティカル用途にコンピュータを採用する軍事システムの個数が急速に増えてきている。

システムのＶ＆Ｖに関する技術的諸問題は、累進的により解決困難になってきた。広範な特定の関心を象徴する例として、包括的な理論的知識が必ずしも存在するとは限らない力学的原理（dynamics）に基づく本来的に不安定な航空機あるいはプロセスを制御するのにコンピュータを適用することについて述べることができる。また、今や、コンピュータ・ハードウェア技術の進展は、ソフトウェア制御によって経済的に実現できるシステム機能の個数、複雑度、範囲、そして相互結合を拡大することを可能にしている。設計の複雑度の水準が高くなったことが、セーフティ・クリティカル・システムでの悲劇的な設計誤りをおかすより多くの、そしてよりとらえがたい機会の原因となっている。加えて、マイクロプロセッサおよびメモリあるいは周辺制御装置中の目立たないロジック誤りあるいはデータに依存する誤りが原因であると考えられるコンピュータ・ハードウェア設計誤りを示唆する証拠の一団が増えてきている。

ＭＯＤが認識している最後の要素では、爆発性兵器制御システムにソフトウェア制御を適用する必要性から生じたセーフティ・クリティカル・コンピュータ・システムを取り扱う新しい方針指令（policy directive）を必要としている。操作では、より洗練された制御を必要としており、それを実現するために、多分、ソフトウェア・ロジックの適用にありったけの期待をかけているが、そのことが、伝統的な時計製造業のほとんど完全な終焉の最大の原因となっている。従来は、時計製造業での精密機構の製造の能力が、兵器の装着および信管取り付け装置の経済的な製造の基盤を提供していた。英国において、兵器システムの安全性に関する基本原則は、兵器会議（Ordnance Board）が制定した（技術的操作に関する）取り扱い規則集（precept）に収録されており、安全であることが示されるまでは、すべてのものは危険であると仮定し、危険な誤りの存在に関する証拠がないことが、危険に関する証拠とはならない。それ故に、兵器会議では、これらの厳格な規則を、従来のコンピュータ・ソフトウェア開発慣行に適用することを承認できないと決定した。

今後、長期間にわたって、セーフティ・クリティカル用のソフトウェア工学を新しい基盤の上に置く必要性を示唆するたくさんの証拠が集まってきたことを直視し、ＭＯＤでは、1985年に、新しい方針、手順および技術規格を検討するセーフティ・クリティカル・ソフトウェア運営部会（Safety Critical Software Steering Group）の設立を決定した。

6．今後のセーフティ・クリティカル・システムに関する方針の基本的な特徴

運営部会では、ソフトウェア安全性に関してはセーフティ・クリティカル・システムでの全体のアーキテクチャから分離することができず、システム機能をセーフティ・クリティカル・システムのハードウェア構成要素とソフトウェア構成要素とに分離・分担させる件が重大な考慮事項であることが、着手してすぐに、はっきりと見えてきた。その原則から、最初に、技術規格と規制上の手順（regulatory procedure）の２つのセットを同時に作成することに専念することが必要であった。暫定版の国防規格00-55ではセーフティ・クリティカル・ソフトウェアの開発および検定を規制する予定で、一方、暫定版の国防規格00-56はセーフティ・クリティカルとして取り扱うべきシステム機能を決定するリスク評価の枠組みを規定している。この文脈では、“暫定版”という用語は、そのような規格に対する技術的に正当化する理由がまだ完全には確定されていないということを、公式に認めていることを意味し、それ故に、ＭＯＤ法規では、プロジェクト・マネジャーが、特定のケースで、そのような規格を励行する範囲や厳格さの度合を決定することに関して、ある程度の自由裁量を持っていることを認めている。運営部会では、今後、手法の研究やツールの開発が必要であると勧告しており、その場合には、暫定規格は、開発資金（funding）を決める際の技術要件の条項となるはずである。

運営部会では、次に示すように、技術規格および規制上の手順の開発に関する作業仮説とガイドラインのセットを確定した。

①ソフトウェア安全性は、一般大衆の関心を持つ公共の問題になると仮定されなければならない。公衆の信頼感は、“悪いことは秘密にしておくことだ”と言えることができるあらゆる情報を非公開にすることによって、徐々に弱められることになる。それ故に、すべての方針、規格、手法、そして支援ツールは機密扱いでなくすべきである。

②セーフティ・クリティカル・システムおよびソフトウェアの設計および構築に関する包括的で容認できる技術基盤はまだ存在しない。それ故に、可能な限り最も広範な知識ベース（intellectual base）へのアクセスが重要であり、方針、規格などを機密扱いではない領域に置いて、アクセスを容易にすること。

③特定のシステム設計およびアプリケーション・ソフトウェアの詳細は必要に応じて機密扱いにすること。

④軍事システムを巻き込んだ事故の調査は、批判的レビュー（judicial review）を受けること。

⑤セーフティ・クリティカル用のソフトウェア工学ツールおよび手法に関しては、セキュア・コンピューティング・システム（secure computing system）用の高インテグリティ設計原則を確立するために行われた研究を理解し、追いつくべきである。

⑥すべてのソフトウェア誤りは、実行装置に設置する際に複数個のコピーを製造する工程では作り込まれないという意味で設計誤りである。設計誤りがないことは、安全なソフトウェアにとって必要条件ではあるが、十分条件ではない。

⑦システムの安全性に関する証拠は、ソフトウェア開発工程での出荷可能な成果物であるべきである。

⑧調査の目的で、システムの安全性に関する証拠を、後日、独立の検査官が検証することが可能であるべきである。

⑨セーフティ・クリティカル・システムに関しては安全であると証明されていなければならない。安全であることが証明されうるソフトウェアは、少なくとも、入力データ域全体にわたって論理的に解析可能で、数値的に決定可能である性質を有すること。

また、運営部会は、完全に強制的な規格の導入目標を1995年に設定した。暫定規格は、できるだけ早く、願わくは、1990年の終りまでには、利用可能にすべきである。

これまでに出現した規格案は、変更を受け、そして、確定したＭＯＤ方針を提示していない作業用文書であることを、はっきりと理解することが非常に重要である。公の諮問や公開討論会の過程では、規格に盛り込まれている寄稿した如何なるアイデアも社会の共有財産（Public Domain；［訳注］著作権、特許権などが自由に誰でもが使用できるという権利消滅状態）になることに同意することを条件にして、いかなる利益団体からの寄稿をも受け入れやすい状態を持続している。しかし、構築や開発の際に使用されるソフトウェア・ツールに対する私的な知的所有権の存在に関して、そのようないかなる制限を設けるつもりもない。はっきり言えば、運営部会では、暫定規格中の要件に関する確固たる条項の存在自体が、高インテグリティ・ソフトウェアの設計および構築で使用する累進的によくなるツールおよび手法に対する独立の開発を促進することを願っている。

7．ＭＯＤ規格の方針

ＭＯＤは、完全な国内用途用の規格よりも国際規格を、軍規格よりも一般（即ち、民間部門）規格を選ぶ、明確で理路整然とした方針を持っている。暫定規格を非常に長期間使用することは、今後出現する英国家規格あるいは国際規格をＭＯＤでの使用に採用できるか否かを決める際のベンチマーク（benchmark；測定基準）を確立することになる。しかし、現時点では、ＩＥＣ（International Electrotechnical Commission；国際電気標準会議）で開発された規格は、詳細な要件を設定するよりも、特定用途向け規格に含まれるべき事項を規定したメタ規格の形式を採りそうである。それ故に、国際用途用の共通の主要な規格の有望な候補規格は、まだ断固として制定されていない、と言える。

8．暫定版の英国規格のハイライト

ここでは、我々の要件と本稿の前章までで述べた原則とが一致していることを強調する目的で、暫定版のＭＯＤ規格の主要な特徴を紹介する。

操作員あるいは傍観者を傷つける可能性のある軍事装置に組み込まれるすべてのコンピュータ・システムは、そうでないと証明されるまでは、セーフティ・クリティカル機能を実行すると仮定される。それ故に、本規格では、プロジェクト・ライフサイクル中の設計および開発工程のすべての範囲を網羅するだけでなく、

設計・動作安全検定合格書（design and operating safety certificate）の発行についても規定する。装置のライフサイクルでの後続の諸工程には、大量生産（series production）、役務中利用、予防保守、修理、修正、そして廃棄が含まれる。システムの安全性が、量産および役務の期間中に危なくされないことを確実にするために、積極的な手段（positive measure）を採らなければならないことが明白に見えているのに反し、本規格に対する主要な酷評は、ライフサイクル中での開発および設計の部分の方に向けられている。

概念定式化段階でさえも、予備ハザード・リスト（Preliminary Hazard List）に入れるべきシステムの主要な機能構成要素を十分に明瞭に特定することが可能であるべきである。心に描いたシステム・アーキテクチャを適切に調整することによって、ハザードの厳しさをおおいに軽減したり、そして、ハザードがシステムの様々な構成要素全体にわたって散在している範囲を狭めたりすることが可能かもしれないので、できるだけ早期の段階でハザードを特定することの重要さに関しては、あまり極度には重要視することはできない。ハザード・リストは、実現可能性調査およびプロジェクト定義工程を通じて累進的に洗練化され、プロジェクトのフル・スケール開発段階での基盤として使用される公式の調達仕様書に対する支援文書である最終のハザード解析報告書（HAR: Hazard Analysis Report）になる。それ故に、システム開発工程の初めの時点でシステム安全性の基礎（foundation）を築くことには、組み込み型コンピュータおよびソフトウェアの使用に特有なこれらの構成要素をも含めてすべての契約上の安全要件に合致させる責任を有する主契約業者による安全担当者（Safety Officer）の任命が含まれる。また、早期の段階で、即ち、最初からは必ずしも必要としないが、しかし、詳細開発作業が実際に始まるよりも遅くない段階で、独立の安全監査官（ISA: Independent Safety Auditor）が任命される。英国の用語では、ＩＳＡは、主契約業者と正式の下請け契約をした専門家集団のソフトウェア会社でありうるが、しかし、ＭＯＤ調達庁のプロジェクト・オフィスへの／からの情報アクセスに対する完全なトランスペアレントな権利を有している。

開発工程の期間中では、ハザード解析報告書（HAR）が、システム設計を洗練化する目的で継続的にレビューされる。本稿での対象である暫定規格では、装置開発でのコンピュータ・ソフトウェア構成要素を主対象としているので、ＭＯＤ法規および手順に関する全体の既存の枠組み内でのシステム全体の安全管理は、調達過程全体にわたっての継続要件であることを、はっきりと理解しなければならない。

システムがいったん抽象アーキテクチャ構成要素に分離されると、ソフトウェア設計工程での初期の並列作業では、抽象アーキテクチャ構成要素を機能モジュールに分割する。分割過程が繰り返されると、システム・アーキテクチャとソフトウェア・モジュールとがかなり一致するようになることはありそうではあるが、しかし、作業を継続することによって、設計者は、セーフティ・クリティカル・ソフトウェア機能を分離され、かつ良く隔離されたモジュールに閉じ込めるようにする。このやり方では、形式的方法を使用することにより、ソフトウェア仕様および設計の正当性に関して制定している本規格の要件に合致させることがより容易になる。この点に関しては、ＭＯＤ規格を遵守すれば、ソフトウェア設計において実際にシステムの安全性を向上させるように意思決定をするようになりそうである。それ故に、ＭＯＤセーフティ・クリティカル・ソフトウェア運営部会のメンバーは、本規格で採用しているアプローチがあまりにも学術的であるという、産業界からの批評に答えるに際して、極度に弁明しようとは思っていない。また、暫定規格の広範な配布を追跡することによって、今や、開発の初期段階でシステム仕様および設計の妥当性を確定することに関して、より厳格さを要求したことによる全般的な効果によって、開発コストとタイムスケール（timescale；時間の物差し）に関する与えられた制約内で、納入されるソフトウェアに対する全般的な品質を向上させそうである、という非常に広範な意見の一致がある、ということが言える。

セーフティ・クリティカルであると証明されたソフトウェア・モジュールの設計に対する暫定規格での主要な要件は、次に示すように要約される。

①プログラム構造、制御フロー、他のソフトウェア・モジュールとのデータおよび制御インタフェース、直接結合されるハードウェア装置とのデータおよび制御インタフェース、メモリ編成、そして、フェール・セーフ戦略を網羅した完全な設計書が必要である。

②すべての設計書およびソース言語形式のソフトウェアだけでなく、支援ツールやコンパイラなどを網羅する厳格な構成管理（Configuration Control）を実施すること。また、構成管理は、オブジェクト・コード・モジュールまでにも広げる必要がある。そして、構成管理では、ハードウェア構成とソフトウェア構成との一貫性に関しては、システムのスイッチを入れるたびごとに、さもなければ、初期化されるたびごとに、検証可能になるように、それ自身で、内部的な“示差的特徴”（fingerprint）を持っていなければならない。

③モジュール仕様は、Z（［訳注］Zermelo(人名)から命名；Zermelo-Fraenkel集合論と述語計算に基づいた英オックスフォード大学で開発されたソフトウェア仕様記述技法）、ＶＤＭ（［訳注］Vienna Development Method）、あるいはＯＢＪ（［訳注］代数的仕様記述言語）のような形式的言語での記法で表現されていること。仕様は、すべての潜在的にハザードであるデータ・フロー領域および制御フロー領域での一貫性および完備性を確定するために解析されなければならない。

④モジュール仕様は、ソフトウェア設計でのすべての段階にわたって検証可能なように入念に作られ、実行可能なオブジェクト・コードとして実現すること。

本規格では、設計書に記録されなければならない情報の種類について、かなり詳しく規定している。

基本的な要件は、すべてのセーフティ・クリティカル・ソフトウェアが、操作域にわたって、形式的に検証された仕様と合致していることを確定するために、Ｖ＆Ｖを受けることである。従って、暫定規格では、次に示すような指令（directive）を含んでいる。

①次に示すようなソース・コードに対する静的解析を実施すること。
- データ使用解析：値が設定される前に変数が使用されていないこと；未使用の変数がないこと。
- 制御フロー解析：制御の到達しないコードがないこと；意図されないループがないこと；複数個の入口を持つループがないこと。
- 情報フロー解析：望まない（unwanted）依存連鎖がないこと、即ち、欠けてる（missing）依存連鎖があること。

②すべてのプログラム分岐および機能が実行されたことを確実にするために、シミュレーションおよびハードウェア不正行為テスト（hardware rig testing）による動的解析を実施すること。テストでは、制御フローとデータ変数とのすべての関連ある組み合せを網羅することは、通常不可能であるので、テストは、しばしば、自己完結的ではない。しかし、静的解析と連係して実施することによって、不完全な動的テストでも、ソフトウェア正当性に関する連係実施による必要不可欠な証拠を提供することができる。

③正当性に関する形式的証明と非証明的論証が提供され、点検されなければならない。公理的証明技術の現在の技術水準では（たぶん、出現するには非常に長期間必要とするが、）フル・スケールのシステム・ソフトウェアに対する厳格な証明が不可能である、と認識している本規格のこの領域は断定的な明瞭さにいくらか欠けている。このように、本規格では、英国の労働衛生安全法で課している法的義務に一致する“合理的に実施可能な範囲内で”システムが安全である推定の根拠が確かであることを独立の安全監査官に納得させるのに必要な十分な形式的論証の提示を要求している。

9．セーフティ・クリティカル・ソフトウェアでのＡＤＡの使用

英国防省は、組み込み型ソフトウェア用途を含んだ国防コンピュータ・システムにＡＤＡを使用することを正式の要件として発表した第１番目の国家の一つである。運営部会では、現時点では、セーフティ・クリティカル・システムに対して、ＡＤＡ方針の適用を免除することを求める必要性もないし、それを正当化する理由もないと結論付けた。ありていに言えば、共通の標準言語を使用することによって、既存の専門知識の最も広範な利用可能な蓄積が使用でき、市場において最も良い入手可能なコンパイラおよびソフトウェア開発ツールが使用できるので、ソフトウェア安全性にとっては明白に有利になると考えている。逆に、専用の高インテグリティ用プログラミング言語の使用を強制することによって、実際には、自分自身で開発したコンパイラや支援ツールに依存することになるので、セーフティ・クリティカル・ソフトウェアを作成する際のリスクに直面するはずである。というのは、最も高い信頼度水準を確立するのには、自分自身でコンパイラや支援ツールを開発したのでは不十分かもしれない。セーフティ・クリティカル・ソフトウェアに関しては、プログラムの正当性の形式的証明用の機構を提供するように特別に設計された高位言語で書かれるべきであると、運営部会は誠意を持って提案している。その見方はまったく正しいかもしれないが、運営部会では、そのような言語用のコンパイラや支援ツールすべてを開発するのには、我々の方針の限界範囲として確定した５～10年間のタイムスケールを超えてしまうと考えている。

セーフティ・クリティカル・ソフトウェアの解析に対する要件では、ＡＤＡ言語が提供する構文要素および機能（construct and facility）の全範囲内で、使用制限を設けない構文要素および機能がどれかを規定する必要があることが直ち

にわかる。それ故に、本規格には、公の諮問期間中に、かなりの量のコメントを引き起こした、いわゆる“禁止すべき慣行”（banned practice）のリストが含まれている。避けるべき設計慣行リスト中の主要な要素は、次に示す通りである。

①動的メモリ管理
②再　帰
③浮動小数点演算
④セーフティ・クリティカル・モジュール中での実行時のデータ・エントリー
⑤マルチプログラミング
⑥割り込み（システム時計変数の更新は除く）
⑦動的プログラム再構成
⑧オブジェクト・コードのパッチング（patching）（例えば、マシン・コードの挿入）

これらの慣行が、常に本質的に危険であるとことを意味しているのではなく、むしろ、それらを使用することによって、プログラムの解析をする際、即ち、ソフトウェアが安全であると証明されいる状態にあることを実証する際の我々の能力に対して、本規格中のそのようなリストに含まれるものが、現在克服することのできない障害物を発生させてしまうからである。プログラムのＶ＆Ｖ手法が開発されれば、現在使用を禁止されている慣行の内のいくつかを使用してもよいようになることが期待できるかもしれない。しかし、他の点では妥当なＡＤＡ構文要素の使用に関しても、いくつかの明示的な制限は必ず残るかもしれない。

禁止すべきプログラミング慣行リストに加えて、ＡＤＡ言語自体の意味の定義に関して、若干の分野で不完全あるいは非一貫である。例としては、①例外処理およびタスキング・モデルの操作、②ソフトウェア・モジュールのコンパイル順序にプログラムの意味が依存すること（順序依存性）、③（アドレス節、仮パラメータ変数間での値の伝達のような）明白にコンパイラの作成の仕方に依存するような機能、④派生型の別名、そして、⑤汎用オブジェクトと現在のその汎用体の具現化との依存関係、が含まれる。ＡＤＡは最悪ではなく、組み込み型ソフトウェアで使用されている他の言語（ＪＯＶＩＡＬ、ＰＡＳＣＡＬ、ＣＯＲＡＬ、ＭＯＤＵＬＡ、ＲＴＬ-２など）よりも、多くの点でより良いということを、ここで強調すべきである。

それ故に、運営部会では、国防規格00-55で規定している解析（容易性）要件をセーフティ・クリティカル・ソフトウェアが満足できることを確実にするには、プログラミング慣行規格が必要であると考えている。従って、運営部会では、望ましくないプログラミング慣行を自動的に回避する適切な支援ツールの開発を促進するために、ＡＤＡの文と構文要素に関する標準レパートリの使用を強制するかどうかを検討している。英国には、ＳＰＡＲＫ（［訳注］SPADE Ada Kernel; SPADE:Southampton Program Analysis Development Environment）と呼ばれる適切な候補が既に存在している。ＳＰＡＲＫにおいては、フル言語のサブセットでは、標準ＡＤＡ文があいまいさのないやり方で使用されていることを検証する適切な解析ツールが使用できるように、強制的なコメント文によって不足情報が補われている。英国の研究活動に比べて、約３年間も遅いが、同じようなアプローチが米国にも存在し、それは、ＡＶＡ（Annotated Verifiable ADA；注釈付き検証可能ＡＤＡ）研究プロジェクトである。ＳＰＡＲＫおよびＡＶＡの規則に則って書かれたソフトウェアは、すべてのコンパイラとの完全な両立性を有しており、ＳＰＡＲＫもＡＶＡもＡＤＡの“サブセットはない”という規則に違反していない。

１０．今後の作業プログラム

英国規格案に対する公の諮問の第一ラウンドでは、約1370件のコメントが寄せられ、今や、セーフティ・クリティカル用途用のソフトウェア工学に関しては現在の形式的方法に関する研究から派生した技法の使用に基づく、強固な工学基盤の上に置かれるべきであるという基本概念を支持する良い意見の一致が見え始めた。それ故に、過去９カ月間に、特定の用途領域の特定の問題に焦点を当てている産業の代表者やその他の人達との一連のより詳細な諮問を実施した。この点に関しては、セーフティ・クリティカル・ソフトウェア規格の要件と航空電子装置の開発に既に適用されている規格とを統合するには、膨大な作業が必要である。この領域においては、基本規格には、制定された耐空性要件（airworthiness requirement）（［訳注］航空機が航空に適する安全性の基準を満たした状態にあること）との完全な両立性を確実にするためには、追加的な詳細が必要である（［訳注］基本規格は広範囲にわたる規定、またはある特定の分野に対する全般的な規定をもつ規格である。基本規格には、直接適用される規格としての機能をもつこともあるし、また、他の規格の基盤としての機能をもつこともある。）。今後発生する非一貫性のリスクを避けるために採用されるアプローチは、航空電

子システムに固有の要件すべてを、航空機の設計要件を制定した国防規格00-970に入れることになる。軍艦のような、その他の主要な領域で同じようなアプローチが採用されることがベストであるように思われる。その場合でも、既に制定された設計および安全性に関する手順および法規を考慮に入れなければならない。現行規格を、国防用途部門に対するメタ規格（暫定ＩＥＣ規格と完全に両立する概念）と考えれば、今や、1990年の末まで、あるいは、多分、1991年の初めに、最終的な形での英国規格を発行することは可能であるように見える。

ＭＯＤ運営部会は、達成すべき進展の度合に非常に満足しているのにも係わらず、次に示すような、たくさんの領域で追加作業や新しい技術規格や手順規格が必要になることは明らかである。

①要件取得過程は、明らかに、弱点の主要な源である。（セーフティ・クリティカル領域であろうとなかろうと、）コンピュータ・システムを構築する際の多くの主要な困難さの原因は、ユーザの操作要件に関する条項が不完全あるいは非一貫であることによる。それ故に、ユーザ要件を詳細な操作要件に変える過程の精度および厳格さを向上させる必要がある。我々は、操作担当ユーザとシステム開発者の間での対話を、Ｚ、ＶＤＭ、あるいはＯＢＪのような現在利用可能な形式的仕様記述言語のいずれかを使用することによって促進することができる、と期待することが現実的であるとは信じていない。それ故に、英語のサブセット、多分、操作担当ユーザが自分のニーズを表現するのに適している、複雑な条件の依存関係をあいまいさなく表現できるようにする補足的な記法を持ったものを開発することが賢明であると考えている。

②現在の設計慣行での別の認識されている弱点は、どの機能をハードウェア・ロジックで実現し、どの機能をソフトウェアで実現するかを決定することを十分に検討することなしに、抽象的な(実現方法に拘束されない)言い方（abstract term）で複雑なシステムを記述する充足手法（satisfaction method）が欠如している点である。それ故に、多分、ハードウェアとソフトウェアの最適な機能分担の解析では、最大の効果を発揮していないかもしれない。それ故に、運営部会では、完全なシステム記述言語を、多分、ＥＬＬＡあるいはＶＨＤＬ（VHSIC Hardware Description Language; VHSIC: Very High Speed Integrated Circuit）のような現在利用可能なハードウェア記述言語のいずれか１つを発展させることによって、開発する必要性を認識している。

③安全性に関する入力（Safety input）をコンピュータ・ハードウェア設計手順に置き換えることが必要であると考えている。そのため、今後の作業での目標は、構成要素の信頼度と、論理的な非一貫状態が発生した際に自分自身でそれを解決するボーティング機構（voting mechanism）によって制御されているシステム冗長性を有する設備の用意との間の相互作用に関するより良い研究開発を可能にすることである。これらのニーズのいくつかは、進行中のＭＯＤの全般的な研究プログラムによって取りかかることも可能ではあるが、しかし、新しい研究が必ず依頼されるべきであることは、まったく明白である。

運営部会では、ＭＯＤ、民間部門担当の政府部門、民間部門、そして学術研究者との間での共同研究の重要さを認識している。それ故に、英国でのこれらの様々な利益団体と既に手を結んでいる。また、より広範な共同研究の可能性に関しては、欧州共同体での様々な制定された規則（institution）の文脈下で、英通商産業省の指導の下で調査されている。本稿の著者は、ソフトウェア安全性の諸問題をできるだけ広範な基盤に基づいてアプローチする中に潜在的な大きな利益があると信じている。精一杯の共同研究を行うことは、労力の重複によって価値ある資源の無駄使いをすることを意味していないであろう。また、共同研究では、特に、ソフトウェア工学のような比較的成熟していない分野の場合には、多様なバックグラウンドからの著しく異なるアイデアの間での相互作用から生じる全体的な利益を最大にする必要がある。

１１．おわりに

英国防省セーフティ・クリティカル・ソフトウェア運営部会では、セーフティ・クリティカル・コンピュータ・システムおよびソフトウェアの開発に対する新しいアプローチを明らかにすることに関して、かなり進展させた。システム・ハザード解析と、ソフトウェアの開発および検証に形式的方法を使用することを基本にした新しい国防規格は、市民からのコメントを得るために発行され、たくさんの支持する返答が収集された。今後の研究や追加の規格に関する様々な要件が特定されたが、しかし、新しいアプローチを英国の国防装置調達慣行に導入するよりも前に解決しておく必要のある問題はない。運営部会では、軍事コンピュータ用途の分野で米国およびその他のＮＡＴＯ加盟国と、そして民間産業、学界との協力関係が発展することを期待している。

参 考 文 献

[1] UK Interim Defence Standard 00-55: Requirements for the Procurement of Safety Critical Software in Defence Equipment, Ministry of Defence, Directorate of Standardisation, Kentigern House, 65 Brown St., Glasgow G2 8EX, May 1989.

[2] UK Interim Defence Standard 00-56/1: Requirements for the Analysis of Safety Critical Hazards, Ministry of Defence, Directorate of Standardisation, Kentigern House, 65 Brown St., Glasgow G2 8EX, May 1989.

Ⅳ. 潜在的に危険な産業でのソフトウェア安全性検定

細　目　次

主　旨

1．はじめに

2．一般的な諸問題

2．1　ソフトウェア検定とは

2．2　問題は何なのか

2．3　ソフトウェア安全性検定

2．4　ソフトウェア安全性検定は必要か

3．原子力産業でのコンピュータの利用

3．1　英国での許認可

3．2　英国以外の国での許認可

3．3　Westinghouse社のＩＰＳ

4．ＵＫＡＥＡでの関連する研究

4．1　ＩＳＡＴ

4．2　ＳＰＥＣＫ

4．3　ＧＲＡＳＳ

4．4　ＲＥＱＵＥＳＴ

4．5　ＳＣＯＰＥ

5．他の潜在的に危険な産業への適用

謝　辞

参考文献

潜在的に危険な産業でのソフトウェア安全性検定

(Software Safety Certification in Potentially Hazardous Industries)

著　者：Chris Dale(United Kingdom Atomic Energy Authority,
National Centre of Systems Reliability)

出　典：Software Certification ed. by de Neumann, B.,
London: Elsevier Applied Science, Reading, 1989, pp.100-112

[主　旨]

本論文では、原子力産業でのソフトウェア安全性検定に関する諸問題での関連する経験に基づいて、潜在的に危険な産業でのソフトウェア安全性検定に関する諸問題について検討する。原子力分野でも、他のエンジニアリング分野と同様に、コンピュータおよび高度技術の出現によって、現在既に達成されているものをしのぐ安全で高信頼なシステム制御をより当然採用する好機となっている。全世界的な原子力産業では、ソフトウェアに基づくシステムの安全性の正当化理由を示す固有の困難さのために、これらの新しい技術を注意深く採用した。ＵＫＡＥＡでは、ソフトウェア検定に関する諸問題に関係する広範な研究開発を実施している。これには、ソフトウェア信頼性評価、リアル・タイム・システムの形式的仕様、そして、超高信頼システムのテストに関する実験的な研究が含まれる。この研究と、潜在的に危険な産業での今後のソフトウェア検定活動との関連について述べる。

1. はじめに

本論文の目的は、潜在的に危険な産業で利用されているソフトウェアを含むシステムに対する安全性検定（safety certification）について検討し、そして、ＵＫＡＥＡ（United Kingdom Atomic Energy Authority）で実施している関連する研究について述べることである。考慮対象のシステムは、システム中に常駐しているソフトウェアがシステムの安全運転に対して潜在的に影響を与えるすべてのシステムである。英国の動力用原子炉（power reactor）のシールド（shield）設計の際に利用されるコンピュータ・プログラムのような、システムの設計の際に利用されるソフトウェアに対する評価（assessment）もまた重要ではあるが、これは本論文の対象範囲外である。

いくつかの一般的な検定に関する諸問題について検討した後で、原子力産業でのソフトウェア検定活動をレビューし、そして、ＵＫＡＥＡの関連する研究を要約する。本論文のまとめでは、ソフトウェアの検定に関する原子力産業での経験を他の潜在的に危険な産業へ適用することについて検討する。

2. 一般的な諸問題

2.1 ソフトウェア検定とは

多くの型のエンジニアリング・システムでは、しばしば、安全性の面から検定を受けることを要請されている。例えば、民間航空機では耐空性（airworthiness）の検定、そして、原子力発電プラントでは運転の許認可（licence）を必要としている。ある種のエンジニアリング・システムでは、これらのシステムにコンピュータ・ソフトウェアが使用されるずうっと以前より、検定の必要性があったという事実を承知の上で、例えば、航空電子工学のように、その検定がソフトウェアを含むこれらのシステムの諸側面を網羅している場合には、ソフトウェア検定に関する疑問が自然とわいてくる。それにもかかわらず、一般には、ソフトウェア検定とは何であって、それをどのようにして実施するかについて明確な意見の一致がない。

Pascalのようなプログラミング言語用のコンパイラに対する検定では、問題としているコンパイラに対して、固定された一連のテストを受けさせることによって実施され、そして、一般に、コンパイラの完全性（integrity）に対する信頼（confidence）は、これらのテストが、コンパイラが保持している諸機能を大いに反映しているという仮定に依存する。このやり方で検定された製品を信用（reliance）するには、明確な（述べられていない）限界がある。セーフティ・クリティカル・アプリケーションの場合には、コンパイラは、広範な利用に基づいて、その性能や諸特性について良く理解されているという意味で、成熟しているべきである。

ソフトウェア正当性に関する検定は、厳密な精査（examination）に耐えられない、別の人を引き付ける、概念（notion）でもある。絶対的な意味では、ソフトウェアの正しい一片はない。即ち、ソフトウェアは、仕様のようなある物に対してのみ正しいのである。問題としている仕様が形式的仕様で与えられた場合には、ソフトウェアがその仕様に対して正しいということを証明するのは、原理的に可能である。しかし、実際には、これは、非常に制限された意味においてのみ実施可能である。仕様それ自身は、変化しやすい“現実世界”の要件を抽象化したもので、その抽象化したものが不完全であるかもしれないという別の問題が存在する。

本論文の目的にかなうように、ソフトウェア検定とは、「ソフトウェアによって提供される実際のサービスと、その規定されたサービスとの等価を評価（assessment）する」と定義する。ソフトウェアの安全性の検定においては、安全性に対して潜在的に影響を及ぼすサービスの諸側面のみに関心がある。

2.2 問題は何なのか

複雑な論理的な諸要件を、経済的に、高信頼に処置できるというコンピュータの実際的な能力によって、あらゆる種類のエンジニアリング・システムに広範にコンピュータがますます採用されてきている。ソフトウェアの修正（適切に制御されたやり方で、かつ、適切なV&V(Validation and Verification; 妥当性検査および検証）チェックを受ける）は、相当する布線式システム（hard-wired system）に比べて、より容易であり、そして、コンピュータ技術を利用する以外には考えられないほどの、より多くの情報を運転員に提供できる。これらは、ソフトウェア・システムを採用することによって生じる非ディペンダビリティ（un

dependability）であるという評判にもかかわらず、技術者に対して、布線式デジタル／アナログ・ハードウェアよりも、コンピュータ技術を採用させることをせきたてる諸々の理由である[1]。

多くの意味において、ソフトウェアはハードウェアと非常に異なっており、これらの相違点が、ソフトウェアの検定に対し、特別な注意を払う必要性を決定付けている。Parnas et al.[1]は、これらの相違点を詳細に検討している。彼らは、①複雑性、②誤り感度性、③テストの困難性、④故障（failure）の相関性、そして、⑤専門資格基準の欠如、の諸問題を特定している。

①複雑性：

これが、ソフトウェアとハードウェアとの間での最も明らかな相違点である。そのタスクが、ハードウェアで実現するにはあまりにも複雑である場合には、しばしば、ソフトウェアによる解決策が採用される。

②誤り感度性：

これが、諸事態（situation）が一見ほんの少し変化した結果として、システムの挙動を大きく変化させるというソフトウェアの特性である。“許容”（tolerance）という工学概念は、ソフトウェアに対しては意味をなさない。

③テストの困難性：

ソフトウェアは、十分なテストを実施するのが疑いもなく困難である。これに対する一つの理由は、２つのテストの間に存在する諸事態に対するテストの結果に関する内挿（interpolation）はまったく信頼に値しない（untrustworthy）という点である。

④故障の相関性：

同じ仕様に対して独立にプログラムされたソフトウェアのそれぞれの一片が、共通要因誤り（common error）をおどろくほどたくさん持っている[2]。ある程度の信頼性の向上を確信を持って予期することができることは明白ではあるが、このことが、ソフトウェアに基づくシステムにおいて多様性（diversity）に対して期待できる事柄に関して厳しい制約を課している[3]。

⑤専門資格基準の欠如：

他のエンジニアリング界では認定された技術者がいるのに、ソフトウェア界では、技術や慣行がまだ十分には成熟していないため、認定された技術者がいない。

以上述べたように、ソフトウェアの特有な性質のため、ソフトウェア検定のプロセスに関して多くの固有の困難さがあることがわかった。

2.3 ソフトウェア安全性検定

ここまでは、安全性あるいは信頼性についてほとんど述べなかった。これ以降では、その安全運転がソフトウェアに依存するシステムの安全性の検定について注目する。

安全性と信頼性とは密接に関連した概念である。しかし、達成すべき信頼度の所定の水準に対する検定と、特定のシステムが安全であるかの検定との間を区別することが要諦である。ソフトウェア信頼性解析では、典型的には、故障がどの位起こりやすいかという頻度を定量化することを試みることに関心がある。一方、ソフトウェア安全性解析では、むしろ、ソフトウェア・セーフティ・ハザード（software safety hazard）とシステム・セーフティ・ハザード（system safety hazard）を特定し、それらがどのように関連があるかを調べることに関心がある。

最初に与えたソフトウェア検定の定義では、ソフトウェアの機能属性と非機能属性を直接観察（observation）し、直接測定するやり方に基づく検定を示唆した。それ故に、その開発プロセスだけを対象として製品を検定することは妥当ではない。即ち、製品が精査されうる状態にある場合に限って、製品が検定可能となり、そして、開発プロセスに関するいかなる情報も、製品精査（product examination）の際の支援に利用されるべきである。はっきり言えば、セーフティ・クリティカル・アプリケーションの場合には、製品の諸属性だけを対象とする検定では決して目的を達成することができず、その開発プロセスが受け入れられるということが事前に分かっていることが、製品の検定に対する前もって必要となる条件であるべきである。

同様に、信頼度の特定の水準の達成と、この水準が実際に達成されたことの実証との間を区別することが要諦である。即ち、安全なシステムを構築することと、安全なシステムが構築されたことの実証との間には、相違がある。ソフトウェア工学技法を適切に適用することによって、時々は、非常に高信頼なソフトウェアを製造し、そのソフトウェアを特定のシステムに組み込んでもセーフティ・ハザードを引き起こさないということは疑いがない。しかし、少なくとも、潜在的には、特定の開発戦略が採用されている場合には、ある種の典型的な製品群から期待できる信頼度の水準について述べることは可能かもしれないが、最良の技法が利用されているという事実だけでは、特定の製品に関連する信頼度の水準が達成されていると言うことはできない。所定の製品の信頼度に関する実証は、その特定の製品の信頼度を測定することによってのみ得ることが可能である。

それ故に、信頼度の所定の水準に関して特定の製品によって達成されたという検定は、その製品を実運用するか（ある全体のシステムの一部として可能な場合）、あるいは、現実の使用と関連しているテスト制度（regime of testing）によってその製品をテストするかのいずれか、または両方に基づく時のみ可能となる。ソフトウェア・テスト環境と現実の利用環境とを関連付ける唯一の既知のやり方は、テスト環境中でその使用（あるいは様々な使用）をまねることである。これ自身は非常に挑戦的な活動であり、この活動が、ソフトウェア信頼度測定が検定に対してどの位有用であるかの度合の限界を設定してしまう。

所定の開発プロセスを適切に適用しているかに関する検定は、ソフトウェアの信頼性あるいは安全性の検定を構成してないことを強調しなければならない。高品質（の作業による）開発が、通常、高品質製品を生み出すことを期待するのは正当ではあるが、このことを保証することはできない。

2.4 ソフトウェア安全性検定は必要か

特定のシステムの文脈下で、ソフトウェア安全性の検定を実施するには、次に示す3つの大きな障壁がある。

- 技術的困難さ
- 必要とするコスト
- ソフトウェアの可視性の欠如

第１番目の項目は別の所で扱い、そして、第２番目の項目は重要ではるが、本論文の対象範囲外である。第３番目の項目は、コンピュータ制御式装置が産業プラントのセーフティ・クリティカル領域で利用するために調達され、かつ、供給業者が、ソフトウェア面の検定を可能にするのに必要な技術的詳細を提供するつもりがない、あるいは提供できない場合に、典型的に発生する。

セーフティ・クリティカル用のＰＥＳ（Programmable Electronic System；プログラマブル電子システム）はセーフティ・クリティカル領域以外で利用されているＰＥＳと、しばしば同一であることを特に心に留めておいていれば、このことはまったく困難とはならない。明らかに、製造会社は、検定に関する特別な要請がある比較的小さな市場に興味をなくしている。一般的なケースでのこの問題に対する安易な解答はない。

その問題に対して、利用可能な多くの特定の具体例の中に１つの解決策がある。システム・レベルでの入念な設計によって、ソフトウェアの安全性に依存することを軽減したり、あるいは除去したりすることが可能であるケースがしばしばある。例えば、機械的あるいは電子的なインタロック（interlock）は、システムがそのソフトウェアによって誤って生成された危険な命令を実行することを防ぐのに、時々採用可能である。

このことは、ソフトウェアを全体として検定する必要性について考慮する必要性を明らかにするのに役立ち、システムがうまく設計されている場合には、検定に対する諸障壁を、経済的なやり方によって、克服することが可能であることを実証した。

３．原子力産業でのコンピュータの利用

原子力産業内では、コンピュータがプラントの通常制御・監視系に長年利用されており、そのため、コンピュータを安全関連情報システムに導入しようとしている。仏国や独国では、安全保護系にコンピュータを利用した原子力発電所が許認可されている。これらの様々な種類のシステムに対する検定の必要性の度合は、安全性に対するその影響度合にはっきりと依存し、そして、等級別の要件に応じて、いくつかの分類方式（classification scheme）が提案されている[4]。ＩＡＥＡ（International Atomic Energy Agency；国際原子力機関）のような機関で

は、協議や協調の機構を提供しているが、必要とする検定手法は、各国の許認可法規が異なっていることもあって、各国で別々に開発される傾向にある。

3.1 英国での許認可

英国では、全面的にコンピュータに基づく安全保護系を利用した動力用原子炉はない。Sizewell-Bプラントは、より“伝統的な”ＳＰＳ（Secondary Protection System；２次安全保護システム）を保有しているが、コンピュータに基づくＰＰＳ（Primary Protection System；１次安全保護システム）を採用する最初の原子力発電プラントとなる予定である。（［訳注］Sizewell-B原子炉のコンピュータ式運転停止システム(Shutdown system)は10万行のソース・コードで、300～400台のマイクロコンピュータからなり、制御機能と運転停止機能の両方を保持している[24]。）その他の安全関連の役割り（より新しいＡＧＣＲ(Advanced Gas Cooled Reactor；改良型ガス冷却原子炉)での燃料補給のような）にコンピュータを利用した原子炉はいくつかあるので、原子力発電所内での安全関連の役割りにコンピュータを利用するのに伴う特定の困難さについて研究すべきことがたくさんある。

原子力発電所の運転の許認可は、ＮＩＩ（Nuclear Installations Inspectorate；核施設査察官）の助言を受けて、ＨＳＥ（Health and Safety Executive；労働衛生安全庁）によって認められる。安全性の合格水準（acceptable level）にあることを実証するのは、許認可される見込みのある業者（prospective licensee）（通常はＣＥＧＢ(Central Electricity Generating Board；中央電力発電委員会)）の責任（responsibility）である。

ＣＥＧＢでは、許認可に際して適用される設計安全性評価基準[5]と設計安全性ガイドライン[6]を利用する。その評価基準およびガイドラインでは、原子炉安全保護系の性能を定量化するＰＲＡ（Probabilistic Risk Assessment；確率論的リスク評価）を利用することを期待し、そして、現時点では、“共通要因除去”（common mode cut off）として知られているものを利用することを提案している。火災、爆発、そして保守／設計フォールト（fault）が原因であるシステム故障の発生確率を10^{-5}以下にすることを規定している。慣行では、数値10^{-5}は極端な状況（circumstance）下でのみ受け入れられ、通常は10^{-4}で、そして、ソフトウェアに関心がある場合には、10^{-3}が現時点でのより分別のある数値であるかもしれない。そのような上限の狙いは、高信頼度を達成するための冗長度を可能な限り

制限する合理的な上限を設定する点にある。また、システムがコンピュータに基づいている場合には、ソフトウェア中によく内在しているかもしれない設計誤りの潜在性を認めているためである。共通要因除去の帰結としては、仏と米の両国で現在慣行となっているものとは異なり、もっとも起こりやすいフォールトに対して、安全保護用の2次装置を用意しなければならないことになる。

特に英国で採用されている別の重要な原理はフェール・セーフ（fail-safe）の原理である。即ち、これは、故障が安全側になるように要求する設計原則である。しばしば、これは、プラント内で故障が起こった時に、故障を検知し、プラントを安全条件に移行させることを意味する。

ＮＩＩでは、許認可申請を裁くために、以下に示す事項を含む安全性評価原則[7]を利用している。

①安全保護系での単一故障が、運転をさまたげないこと。

②適切で安全な制限値群は、安全保護パラメータ群として特定され、これらの制限値内では、満足のいく運転であることを示すこと。

③直接計測の方が間接計測の方よりもこのましい。（［訳注］炉の性状などを計測して（直接計測という）プロセスを制御することが望ましいが、計測器の信頼性や測定の遅れ、コストなどから限定されるので、一般には、温度、圧力、流量、液位などのプロセスの状態を表す変数を計測して（間接計測という）プロセスを制御する。）

④予期しないイベントが起こることを想定し、十分な冗長性および多様性を用意しておくこと。

⑤信頼度要件が非常に高く、かつ、システムの信頼度が疑わしい場合には、多様性を用意しておくこと。

⑥証明済みの構成要素のみを使用すること。

⑦システムの不安全な故障については、合理的に実行できる時には、警報を出すこと。

3.2 英国以外の国での許認可

EhrenbergerとBloomfieldは、原子力産業でコンピュータを利用することに関連する許認可面での諸問題について詳細なレビューを準備した[4]。このレビューでは、米、英、独、そして仏について詳細に調べている。重要な指摘事項のいくつかを、以下で略述する。

米国には、原子炉用のコンピュータに基づく安全保護系の許認可に関する比較的多くの経験がある。3つの安全保護システムが許認可された。この内の第1番目の安全保護システムは、否認され、全面的に書き直して、許認可された。許認可はNRC（Nuclear Regulatory Commission；米原子力規制委員会）によって実施される。NRCでは、許認可される業者が作成した高品質である論拠（argument of high quality）を監査する。その際には、コンピュータ技術では、原子炉用安全保護系を十分に構築できるという暗黙の仮定（了解事項）がある。NRCが利用した詳細化されたアプローチは、最近の10年間に開発されたものである。問題にしている3つの安全保護システムの各々は、許認可プロセスに合格した。米Westinghouse社のIPS（Integrated Protection System；統合型安全保護システム）に関するNRCの許認可活動については、後で詳しく検討する。

仏国は、原子炉安全保護系にコンピュータを利用する点に関するかぎり、世界中で最も進んだ国であることを十分に論証できる。許認可された重要なシステムを以下に列挙する。

①SPIN：PWR（Pressurised Water Reactor；加圧水型原子炉）用安全保護システム
②CONTROBLOC：プラント制御・監視システム；PWRで使用される。
③TCI：運転員支援システム
④TRTC：Superphenix高速増殖原子炉で使用される燃料要素監視システム
⑤Superphenix安全保護系の自己テスト部分；安全保護系自体の本質的な部分は布線式である。

仏国でのコンピュータに基づく安全保護系の開発は、SPINと併行して開発された規格[8]に従って実施された。その規格は、関連する国際規格[9]と非常に共通点が多い。これらのシステムの許認可の評価基準には、プログラムは単純な構造で、高位言語とOS（Operating System）は利用せず、システムをプロトタ

イピングで作成し、ソフトウェア・レベルおよびシステム・レベルでの活動をレビューすることが含まれている。また、良い品質の装置の利用、徹底的な機能テスト、良いチーム編成、そして独立の検証が要諦である。

仏国では、特に、構築されたシステムのテスト容易性（testability）を重視している。これは、変更されていないソフトウェアが、システムの他の部分に対して、追加、削除、あるいは修正を行った後でも、正しく機能し続けることを示すために、保守後のテストを経済的なやり方で実施可能にするためである。

以上述べた国以外の諸国では、原子力発電の安全関連の側面にコンピュータを利用する開発はほとんど行われていないが、コンピュータ利用の傾向は世界中でどこも同じである。詳細は参考文献[4]にでている。各国では、そのようなシステムの許認可に関して非常に異なるアプローチが採用されている。英国では、コンピュータに基づく安全保護系が原因である故障に対して防護するために、機能的に多様な安全保護系を要求している。一方、仏国と米国では、安全保護系に対する信頼度の必須水準はコンピュータ単独で達成可能であるとしているので、ありがたい（happy）。３国中、仏国だけ、高位言語の利用の禁止を要求している。

３．３ Westinghouse社のＩＰＳ

Westinghouse社のＩＰＳはＰＷＲ用の完全な統合型制御・安全保護システムである[10]。このＩＰＳは米国と英国での許認可を受けたので、許認可活動の主題としては特に興味をそそる。まず最初に米国での許認可活動の経験を要約し、次に英国でのそれを要約する。

ＩＰＳは、安全機能とその他の通常制御機能の側面とにマイクロプロセッサを大量に使用して製作された。また、保守時に混入される故障の可能性に対して再保証をするという条件で、自己テスト機構がシステムの中に組み込まれた。そのソフトウェアは高位のソース・コードで、６万行からなり、独立のＶ＆Ｖを利用して開発された。

米国のＮＲＣは、そのシステムに対して、システム設計仕様工程からシステム・テスト工程までの開発サイクル中で８回の監査を実施して評価した（［訳注］1979年1月～1980年9月）。ある特定の機能群およびパラメータ群のみだけに詳細なレビューを制限し、開発プロセスを通じて、それらを追跡した。その目標は、

Westinghouse社のＶ＆Ｖプロセスを評価して、問題となる領域でのフォールトを精査して、証拠を見つけることである。監査に加えて、アセンブラ言語で書かれた通信ソフトウェアの一部分を解析するために、スニーク解析（sneak analysis）[11]が実施された。

上述した監査では、設計多様性の問題については注意を向けていなかったので、これを実行するため、ＮＲＣは深層防護・多様性ガイドライン（Defence-in-Depth and Diversity Guidelines）の一式[11]を開発し、適用した。これらのガイドラインでは、機能単位、即ち、ブロック（block）間の相互依存を考慮に入れて、調査すべきブロックの故障の組み合せを決定する規則を規定し、そして、プラント・システム間の独立性に対する要件と、２個あるいはそれ以上の機能単位の共通要因故障（common failure）に対する許容について注意を向けている。

ＮＲＣは、独立のＶ＆Ｖ／レビューの戦略が非常に効果的で、それによって、高信頼システムにたどりつくという結論で、ＩＰＳに対して暫定的な設計承認（Provisional Design Approval）を与えた。

Sizewell-BプラントのＰＷＲでは、実際にＩＰＳを開発したWestinghouse社が提供するＰＰＳを保有する予定である。ＩＰＳとＰＰＳとの違いには、ソフトウェアの更新、ソフトウェアに対しては異なる高位言語の利用、改良された冗長性、そして、フェール・セーフの強化が含まれる。形式的手法や仕様記述言語は直接には利用されなかったけれども、ソフトウェア開発方法論の変更には、一般的に言うと、仕様の強化、初期のステージでの検証の強化、そして、ツールの強化が含まれる。

英国のＮＩＩでは、全体の安全保護系が受け入れられるかについて考察し、原子炉安全保護系にマイクロプロセッサ技術を利用することを（細部はともかく）原則として受け入れた[12]。ＮＩＩは、ＰＰＳの受け入れに関する判定をまだ行っていないが、再び繰り返されたソフトウェア（reiterated software；再利用ソフトウェアのこと？）に対する徹底的なチェックの必要性を含めて、注目すべきいくつかの領域を特定した[13]。また、ＣＥＧＢは、可能性のあるトリップ・アルゴリズム（trip algorithm）、運転員に対する情報の提供、そして、テスト機構に関しての各利点を認めて、マイクロプロセッサに基づく原子炉安全保護の原理は受け入れた。また、ＣＥＧＢは、入念な独立の評価が必要であることも認識した[14]。

4. UKAEAでの関連する研究

4.1 ISAT

フェール・セーフ原理は、英国での原子炉安全性評価の焦点であり、そして、ISAT（Individual Sub-Assembly Temperature monitoring；個別サブアッセンブリ用温度監視システム）は、高速増殖原子炉用にUKAEAが開発した高信頼フェール・セーフ監視システムである[15]。このシステムは、注目すべきパターンを含んでいる動的出力信号が生成されるようなやり方で、テスト用入力信号と実入力信号とを多重送信し（multiplex）、交互に重ねる（interleave）。このパターンは布線式のパターン認識ロジックで調べられ、ずれ（discrepancy）が検知されると、原子炉トリップ信号（trip signal）が生成される。即ち、ハードウェア故障がある場合に、正しいパターンが生起するとは考えられない。この設計は、通常、フェール・セーフ領域では絶対不可欠である詳細なFMEA（Failure Modes and Effect Analysis；故障モード影響解析）を利用することなしにフェール・セーフであることを示すことのできるシステムを導き出す。

また、ISATシステムは、ソフトウェアのフォールトが原因である故障から防御されていると主張（claim）されている。しかし、この主張の妥当性には、いくつかの条件、例えば、ソフトウェア中に欠如した経路（path）がないという条件を要する。このような主張は、ISATのように、非常に単純なソフトウェアに基づくシステムの場合のみ裏付けることができる。

UKAEAのWinfrith支所での現在の研究は、ISAT原理を、PWR原子炉の安全保護系で使用されているようなもっと複雑なアルゴリズムにまで拡張できるようにするやり方を精査している。研究中の1つのアイデアでは、逆アルゴリズム（inverse algorithm）$f^{-1}(Y)$を使用する。即ち、炉心安全保護アルゴリズム

$$Y=f(X)$$

の出力群Yを、もとの入力値群Xが何であったかを計算するプロセス

$$Z=f^{-1}(Y)$$

に対する入力群として与える。その時には、これらの“計算された”入力群Z自身は、安全保護アルゴリズムに与えられた入力群Xに等しくなるはずである（次式参照）。

$$Z=f^{-1}(Y)=f^{-1}(f(X))=X$$

すべてがうまくいく場合には、（システムが現実の入力値群Xと、再計算された入

力値群Zを交互に処理するので、）そのようなシステムは安定する。即ち、この安定性からのいかなる発散（divergence）も誤り検出機構として利用可能である。

これらのアイデアについて研究開発すべきことがまだたくさんある。即ち、解くべき諸問題としては、いくつかの複雑な関数に対する逆関数の導出と、いくつかの比較的単純な関数に対して存在する逆の多様性（multiplicity of inverse；逆の重複度）である。これらの諸問題にもかかわらず、このアプローチは、２つの多様な実現されたものが、同じデータをたとえ処理しないというやり方であっても、システムの設計に多様性を強要するアプローチである。

4.2 SPECK

ＳＰＥＣＫは、典型的な高速増殖原子炉の安全性の要件に関するあいまいでなく、完備な、かつ自己一貫性のある記述をしたいという要求に応えて、ＵＫＡＥＡのHarwell支所で開発された形式的仕様化ツールである。特定の技術的な困難さとしては、数千の入力信号群、タイム・クリティカルな応答、そして比較的複雑なアルゴリズムが含まれる。

数学的システム・モデルは、各レベルの各々のサブシステムが、単一入力から単一出力を生成する単一変換を実行するように、階層的な様式で構築された。ＳＰＥＣＫの特定の強みは、仕様での時間に関する側面のモデリングを可能にし、その結果、デッドロック（deadlock）や応答時間を解析可能にしている。

ＳＰＥＣＫに対するユーザ・インタフェースは、プラント技術者側の取り扱いやすさ（tractability）と、数学的形式性の要求との間の溝を埋めることを試みている。可能性のある今後の研究としては、形式的枠組み内での安全性要件と信頼性要件との合体が含まれる。

4.3 GRASS

ＵＫＡＥＡ内のＮＣＳＲ（National Centre of Systems Reliability；国立システム信頼性センター）では、Alveyでのソフトウェア信頼度モデリングの一環として行われている、システムの安全性および信頼性の評価を支援するガイドライン一式の開発に協力している。ＧＲＡＳＳ（Guidelines on Reliability and Safety Assessment of Software；ソフトウェア安全性・信頼性評価ガイドライン）

では、システム全体の評価の文脈内で、ソフトウェア面を評価する枠組みを規定することになっている。

特定のシステムの安全性を評価するに際しては、まず第1番目に、そのシステムの開発を導くプロセスの質を考慮することが基本である。理想的には、情報の紛失を避けるために、この活動は、開発と併行して実行されるべきである。その評価のこの側面だけでは、個々の製品の検定結果の正当化理由を示すのには、不十分である。それを補完するには、システム開発に払われるべき注意さを、システムに課せられた安全性要件に釣り合わせるようにすることが必要である。評価のこの側面に取り組むために、ＧＲＡＳＳでは、査定者（assessor）が、開発プロセスの評価の期間中に正指示子（positive indicator）あるいは負指示子（negative indicator）として捜すべきものを詳細化することになっている。

ソフトウェアの場合、評価可能な第2番目の側面は製品それ自体であり、プロセス評価の期間中に収集した証拠に対して、さらなる証拠を提供するために、ソース・コードに対して様々な静的解析を実施することが可能である。ここでは、ＧＲＡＳＳは、様々な種類の静的解析ツールからの出力を解釈するための助言をし、そして、ＳＦＴＡ（Software Fault Tree Analysis；ソフトウェア故障の木解析）[16]のような技法を利用するための基本的なガイダンスを与えることになっている。

評価の最後の側面は、典型的には、その製品が最終的に利用される可能性のある環境と同じ環境でテストすることによって、製品の挙動を評価することである。ＧＲＡＳＳは、ソフトウェア信頼度成長モデル[17,18,19]を網羅する、この場合に利用可能な手法で、単純な標本論およびポアソン統計に基づく、あまり複雑ではない、より望ましい手法[1,20]についてのガイダンスを与えることになっている。

ＧＲＡＳＳは、潜在的に危険な産業での、ソフトウェアに基づくシステムの検定で重要な役割りを果たすことが期待される。

4.4 REQUEST

NCSRとWinfrith支所の双方は、適切なプロトタイプ・ツールで支援されたソフトウェアの品質および信頼性を測定・モデル化する、改良されて、妥当性が立証された技法を規定することを狙いとするESPRIT－Iプロジェクト（European Strategic Programme for Research and Development in Information Technology、1984年～1988年）P300（1985年7月から60ヵ月）であるREQUEST（Reliability and Quality of European Software Technology；欧州ソフトウェア信頼性・品質技術）に参加している。主要な関心事は、仕様化から実使用までの開発ライフサイクルのできるだけ多くの範囲に及ぶメトリックス（metrics）とモデルを開発することと、そして、プロジェクト管理および品質管理で意思決定をしたり、制御したりするのに必要な有用でタイムリーな情報を規定することである。

REQUESTの研究は、あらゆる等級での検定にとって意味がある。即ち、同じ機能を実行するソフトウェアの異なる版の間の依存性のモデリングについて注目に値する進歩があったので[21,22,23]、高信頼システムの信頼度モデリングについて実施される研究は、安全性の検定にとっても、特に重要である。

4.5 SCOPE

NCSRとWinfrith支所の両方は、技術的に妥当性が立証され、法的に認可された欧州ソフトウェア検定手順を規定することを狙いとするESPRIT－IIプロジェクト（1989年～1993年）P2151（1989年1月から48ヵ月）であるSCOPE（Software Certification on Program in Europe；欧州ソフトウェア検定方式）に協力する予定である。このプロジェクトでは、ソフトウェアによって提供される実際のサービスと、その規定されたサービスとの間での十分性（adequacy）を直接評価する、製品検定で利用される検定技術の確立に関心がある。SCOPEでは、様々な産業での、ソフトウェアに基づくシステムが提供するサービスの検定に関する規定を決めることになっている。

5. 他の潜在的に危険な産業への適用

全世界的な原子力産業には、安全関連システムおよびセーフティ・クリティカル・システム中でソフトウェアを使用する際の諸問題を取り扱う点に関して非常に多くの経験がある。他の危険な産業で、これらの諸問題を取り扱うこの経験を活用することはありそうでもあり、かつ望ましい。原子力産業では、非常に早期に信頼性・安全性に関する諸問題に取り組み、現在も常に力点を置いており、学んだ教訓は、原子力産業のみならず、より一般的な産業に適用できるはずである。

ソフトウェア検定に関しては、セーフティ・クリティカル・アプリケーションで要求される信頼度の水準を達成するために、コンピュータに基づくシステムの能力が広く受け入れられていることを見てきた。この領域での唯一の除外は、最もクリティカルな領域においては、追加の“バック・アップ”システムを要請している国（英国も含めて）もある点である。しかし、信頼度の必要な水準を達成するのに失敗するリスクを最小化しようと試みるために、様々な国で採用している諸原理が大きく違っている。

これらの相違を実証するために、構築言語の選択に関する問題について考えてみる。仏国では、高位言語およびコンパイラの利用は禁止されている。英国では、高位言語の利用は一般的に好ましく、但し、広く利用されていることという注意書き付きで、従って、よく妥当性が立証されたコンパイラを利用しなければならない。そして、米国では、Westinghouse社がＩＰＳ用に作成した専用高位言語を採用しており、それ故に、広く利用されているコンパイラが欠如しているという点を受け入れている。

この例は、ソフトウェア工学に関する共通の主題を例証するのに役立つ。了解した諸問題に対する最適な解決策についての意見に大きな、時には両立しない、相違がある。この特定の例においては、これらの方針（政策）の相違は、ある1つの国で開発されたシステムが、別の国で受け入れられると認められる度合を制限してしまう。そのような方針の相違が、潜在的に危険な産業をおおったならば、そのようなシステムの国際貿易での巨大な障壁となってしまう。ＳＣＯＰＥプロジェクトの狙いの1つは、これらの方針の相違を、少なくとも欧州の文脈下では、除去することである。

そのような方針の相違を導く可能性のある現在進行中の例は、英国防産業によって提供される。ここでは、すべてのセーフティ・クリティカル・ソフトウェア用の仕様の作成に形式的数学手法を導入する方向へ移行した。この方針は、ソフトウェア開発用のこれらの形式的手法が、セーフティ・クリティカル・システムの開発に関して最良の現在可能性のある慣行を代表しているという暗黙の仮定に基づいている。これは多分正しいが、もちろん、異論のない主張ではない。

特定のソフトウェア開発手法で、どんなに技術的なよさがあっても、重要な点は、安全性の検定が安全性の正当化理由に対する批判的な評価にかかっている点である。既に上で指摘したように、まだ十分には成熟していない技術分野において、ある特定の慣行を強制したり、あるいは禁止したりすることを要請する方針は、お互いに矛盾する危険にぶちあたってしまう。そのような方針は、異なる地域で非互換の規格を採用することを確実に導いてしまうであろう。

原子力産業に所属する様々な国家機関が開発したソフトウェアに基づくシステムに関する許認可方式すべてが、大ざっぱに言えば、プロセスおよび製品の両方が高品質であるという論拠（argument of excellence）に対するレビューおよび考査（appraisal）に基づいている。この枠組みでは、開発者と許認可者は、特定の規格類や手順類について、時期尚早的で、規範的な採用を強制するような付随的な危険がない場合に限って、新しい手法が利用可能になった時に、新しい手法を採用することを認めている。これは、特に、ここ１・２年以内に、欧州に存在する予定のオープン市場の観点から、すべての産業が採用すべき重要な教訓である。

原子力産業は、初期の頃より、信頼性・安全性技術の開発を先導した。その技術の大半は、今や、広範な諸産業の利益追求のために用いられている。ソフトウェアを含むシステムの安全性検定は非常に未成熟な学問分野（discipline）ではあるが、しかし、原子力産業では、既に特定の諸問題に取り組んだ非常に多くの経験を身に付けており、この領域について非常に多くの重要な研究を実施し続けている。

この経験や背景は、他の潜在的に危険な産業が、原子力産業内で蓄積されたソフトウェア安全性検定についての経験を活用することによって、今や利益を得ることができるようになったことを印象付けている。

謝　辞

この研究は、Alveyのソフトウェア信頼度モデリング・プロジェクト（ＳＥ／０７２）の一環として、Alvey理事会によって一部支援されていた。

参考文献

[1] Parnas, D.L., van Schouwen, A., and Kwan, P., "Evaluation Standard for Safety Criticality Software," IAEA Meeting on Microprocessors in Systems Important to the Safety of Nuclear Power Plants, London, May 1988; Parnas, D.L., van Schouwen, A.J., and Kwan, S.P., "Evaluation of Safety-Critical Software," Communications of the ACM, Vol.33, No.6, pp.636-648, (June 1990).

[2] Knight, J.C. and Leveson, N.G., "An Experimental Evaluation of the Assumption of Independence in Multi-Version Programming," IEEE Transactions on Software Engineering, Vol.SE-12, No.1, pp.96-109, (Jan. 1986).

[3] Bishop, P.G., Esp, D.G., Barnes, M., Humphreys, P., Dahll, G., and Lahti, J., "PODS - A Project on Diverse Software," IEEE Transactions on Software Engineering, Vol.SE-12, No.9, pp.929-940, (Sept. 1986).

[4] Ehrenberger, W.D. and Bloomfield, R.E., Licensing Issues Associated with the Use of Computers in the Nuclear Industry, CEC Nuclear Science and Technology Final Report, GRS mbH, Cologne, Mar. 1987.

[5] Design Safety Criteria for CEGB Nuclear Power Stations, Central Electricity Generating Board Report HS/R167/81 (Revised), Mar.1982.

[6] Pressurised Water Reactor Design Safety Guidelines, Central Electricity Generating Board Report DSG2 (Issue A), Apr. 1982.

[7] Safety Assessment Principles for Nuclear Power Reactors, HM Nuclear Installations Inspectorate, Apr. 1979 (Published by HMSO, July 1982).

[8] Colart, J.M. and Grisollet, J., Safety Requirements Related to the Software Used in the Safety Systems of Nuclear Reactors, S.A.F. Report No.32.

[9] Software for Computers in the Safety Systems of Nuclear Power Stations, IEC, Geneva, 1985.

[10] Gallagher, J.M., "Software for Computers in Safety Systems of Nuclear Power Plants," IFAC SAFECOMP '83, Pergamon Press, 1983.

[11] A Defense-in-Depth and Diversity Assessment of the RESAR-414 Integrated Protection System, US Nuclear Regulatory Commission Report NUREG-0493, 1979.

[12] A Review by HM Nuclear Installations Inspectorate of the Sizewell B Preconstruction Safety Report, NII Report, HMSO, July 1982.

[13] Sizewell B Public Enquiry: Proof of Evidence on HM Nuclear Installations Inspectorate's View of the Central Electricity Generating Board's Safety Case, NII, HMSO, Mar. 1983.

[14] Sizewell B Public Enquiry: Proof of Evidence by Central Electricity Generating Board, HMSO, 1983.

[15] Keats, B.A., "Failsafe Criteria for Computer-Based Reactor Protection Systems," Nuclear Energy, Vol.19, pp.423-428, (1980).

[16] Leveson, N.G. and Harvey, P.R., "Analyzing Software Safety," IEEE Transactions on Software Engineering, Vol.SE-9, No.5, pp.569-579, (Sept. 1983).

[17] Jelinski, Z. and Moranda, P.B., "Software Reliability Research," in Statistical Computer Performance Evaluation ed. by Freiberger, W., Academic Press, 1972, pp.465-484.

[18] Littlewood, B., "A Bayesian Differential Debugging Model for Software Reliability," Proceedings of Workshop on Quantitative Software Models, 1979, pp.170-181.

[19] Musa, J.D., Iannino, A., and Okumoto, K., Software Reliability: Measurement, Prediction, Application, McGraw-Hill, 1986.

[20] Ball, A., Butterfield, M.H., and Dale, C.J., "The Achievement and Assessment of Safety in Systems Containing Software," IAEA Specialists' Meeting on the Use of Digital Computing Devices in Systems Important to Safety, Saclay, France, Nov. 28-29, 1984.

[21] Saglietti, F. and Ehrenberger, W., "Software Diversity - Some Considerations about its Benefits and its Limitations," IFAC SAFECOMP '86, Sarlat, France, 1986.

[22] Saglietti, F. and Ehrenberger, W., "Considerations on Software Diversity on the Basis of Experimental and Theoretical Work," ESPRIT '86: Results and Achievements, Directorate General XIII (editors), Elsevier Science Publishers B.V. (North-Holland), 1987.

[23] Saglietti, F., "A Theoretical Evaluation of the Acceptance Test as a Means to Achieve Software Fault-Tolerance," IFIP/IFAC Working Conference on Hardware and Software for Real Time Process Control, Warsaw, 1988.

[24] Leveson, N.G., "High-Pressure Steam Engines and Computer Software," ICSE-14, Melbourne, Australia, May 1992, pp.2-14.

V. 参考文献一覧

細　目　次

1．著者名順（ABC順）

2．発行年別（1980年～1993年）

V. 参考文献一覧

ここでは、ソフトウェアの安全性に関する広範な参考文献を著者名順（ABC順）（第１章）と発行年別（1980年～1993年）（第２章）に分類して一覧してある。

[会議名の略称一覧]

COMPASS: The i-th Annual Conference on Computer Assurance: Systems Integrity, Software Safety and Process Security
SAFECOMP: Safety of Computer Control Systems 19XX (Proceedings of the IFAC(International Federation of Automatic Control) Symposium)
ICSE: International Conference on Software Engineering

1．著者名順（ABC順）

Abboytt, R.J., "Resourceful Systems for Fault Tolerance, Reliability, and Safety," ACM Computing Surveys, Vol.22, No.1, pp.35-68, (Mar. 1990).（浦野訳,「フォールトトレランス, 信頼性, そして安全性のための"機略"システム」, bit 別冊 ACM COMPUTING SURVEYS '90 コンピュータ・サイエンス, 共立出版, 1992年7月号別冊, pp.5-36.）

Abdel-Ghaly, A.A., Chan, P.Y., and Littlewood, B., "Evaluation of Competing Software Reliability Predictions," IEEE Transactions on Software Engineering, Vol.SE-12, No.9, pp.950-967, (Sept. 1986).

ACM, "ACM Code of Ethics and Professional Conduct," Communications of the ACM, Vol.35, No.5, pp.94-99, (May 1992).

AFISC SSH 1-1: System Safety Handbook Software System Safety, Department of the Air Force, Headquarters Air Force Inspection and Safety Center, 5 September 1985.

Alzerra, D. and Galivel, G., "Validating the SACEM Railway Control System Using the IDAS Software Test and Debugging Tool," IFAC SAFECOMP '89, Dec. 1989, pp.59-63.

Apostolakis, G.(ed), Probabilistic Safety Assessment and Management: Proceedings of the International Conference on Probabilistic Safety Assessment and Management(PSAM) held February 4-7, 1991 in Bevery Hills, California, New York: Elsevier, Reading, 1991, pp.1545.

Avizienis, A. and Kelly, J.P.J., "Fault Tolerance by Design Diversity: Concepts and Experiments," IEEE Computer, Vol.17, No.8, pp.67-80, (Aug. 1984).

Avizienis, A. and Laprie, J.-C.(eds), Dependable Computing for Critical Applications, Wien: Springer-Verlag, Dependable Computing and Fault-Tolerant Systems, Vol.4, Reading, 1991, pp.429.

Avizienis, A., "Software Fault Tolerance," Proceedings of the IFIP 11th World Computer Congress, San Francisco, Calif., Aug./Sept. 1989, pp.491-498.

Baker, H.G., "Factoring Redundancy," Communications of the ACM, Vol.34, No.5, pp.98-99, (May 1991).

Barnes, M., "Dependable Computing in the UK," in Dependable Computing for Critical Applications, ed. by Avizienis, A. and Laprie, J.-C., Wien: Springer-Verlag, Reading, 1991, pp.3-21.

Bell, R. and Pantony, M.F., "Framework for the Design and Assessment of Safety Related Control Systems," in Failsafe Control Systems: Applications and Emergency Management, ed. by Warwick, K. and Tham, M.T., London: Chapman and Hall, Reading, 1991, pp.70-92.

Bell, R. and Smith, S., "An Overview of IEC Draft Standards: Functional Safety of Programmable Electronic Systems," COMPASS '90, June 1990, pp.151-163.

Bell, R., "Guidance on the Use of Programmable Electronic Systems in Safety Related Applications," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.263-270.

Belli, F. and Jedrzejowicz, P., "An Approach to the Reliability Optimization of Software with Redundancy," IEEE Transactions on Software Engineering, Vol.17, No.3, pp.310-312, (Mar. 1991).

Bennett, P.A., "Experiences Assessing High Integrity Software," IEE Colloquium on 'Control Systems Software Reliability for Industrial Applications', London, Oct. 1989, pp.2/1-2/3.

Bennett, P.A., "Forwards to Safety Standards," Software Engineering Journal, Vol.6, No.2, pp.37-40, (Mar. 1991).

Bennett, P.A., "The March Towards Standards in Safety Related Systems," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.106-110.

Bhansali, P.V., "Survey of Software Safety Standards Shows Diversity," IEEE Computer, Vol.26, No.1, pp.88-89, (Jan. 1993).

Bishop, P.G., Esp, D.G., Barnes, M., Humphreys, P., Dahll, G., and Lahti, J., "PODS - A Project on Diverse Software," IEEE Transactions on Software Engineering, Vol.SE-12, No.9, pp.929-940, (Sept. 1986).

Bjorner, D. and Druffel, L., "Position Statement: The ICSE-12 Workshop on Industrial Experience Using Formal Methods," ICSE-12, Nice, Mar. 1990, pp.264-266.

Bjorner, D., Hoare, C.A.R., and Langmaack, H.(eds), VDM '90: VDM and Z - Formal Methods in Software Development; Proceedings of the Third International Symposium of VDM Europe held April 17-21, 1990 in Kiel, FRG, Berlin: Springer-Verlag, Reading, 1990, pp.579.

Bloomfield, R.E. and Froome, P.K.D., "Aspects of the Licensing and Assessment of Highly Dependable Computer Systems," in Safe and Secure Computing Systems, ed. by Anderson, T., Oxford: Blackwell Scientific Publications, Reading, 1989, pp.1-39.

Bloomfield, R.E., Froome, P.K.D., and Monahan, B.Q., "Formal Methods in the Production and Assessment of Safety Critical Software," Reliability Engineering and System Safety, Vol.32, Nos.1/2, pp.51-66, (1991). Also available in Software Reliability and Safety, ed. by Littlewood, B. and Miller, D., London: Elsevier Applied Science, Reading, 1991, pp.51-66.

Bowen, J.P. and Stavridou, V., "Formal Methods and Software Safety," IFAC SAFECOMP '92, Oct. 1992, pp.93-98.

Bowman, W.C., Archinoff, G.H., Raina, V.M., Tremaine, D.R., and Leveson, N.G., "An Application of Fault Tree Analysis to Safety Critical Software at Ontario Hydro," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.363-368.

Brilliant, S., Knight, J.C., and Leveson, N.G., "Analysis of Faults in a N-Version Programming Experiment," IEEE Transactions on Software Engineering, Vol.SE-16, No.2, pp.238-247, (Feb. 1990).

Brilliant, S., Knight, J.C., and Leveson, N.G., "The Consistent Comparison Problem in N-Version Software," IEEE Transactions on Software Engineering, Vol.SE-15, No.11, pp.1481-1484, (Nov. 1989).

Brooks, F.P., Jr., "No Silver Bullet - Essence and Accidents of Software Engineering," IEEE Computer, Vol.20, No.4, pp.10-19, (Apr. 1987).

Brown, M.J.D., "Rationale for the Development of the UK Defence Standards for Safety-Critical Computer Software," COMPASS '90, June 1990, pp.144-150.

Brown, M.L., "Software Safety for Complex Systems," Proceedings of the Seventh Annual Conference of the IEEE/Engineering in Medicine and Biology Society, Sept. 1985, pp.210-216.

Brown, M.L., "Software Systems Safety and Human Errors," COMPASS '88, June/July 1988, pp.19-28.

Buckley, T.F., Jesty, P.H., and West, M.M., "Some Results from DRIVE," COMPASS '91, June 1991, pp.17-26.

Buckley, T.F., Jesty, P.H., Hobley, K., and West, M., "DRIVE-ing Standards:- A Safety Critical Matter," COMPASS '90, June 1990, pp.164-172.

Butler, R.W. and Finelli, G.B., "The Infeasibility of Experimental Quantification of Life-Critical Software Reliability," ACM SIGSOFT Software Engineering Notes, Vol.16, No.5, pp.66-76, (Dec. 1991). Also available in Proceedings of the ACM SIGSOFT '91 Conference on Software for Critical Systems, New Orleans, Louisiana, December 4-6, 1991.

Cha, S.S., Leveson, N.G., and Shimeall, T.J., "Safety Verification in MURPHY Using Fault Tree Analysis," ICSE-10, Singapore, Apr. 1988, pp.377-386.

Charon, P., Kahn, P., Le Henaff, A., Mignot, J., and Nicolas, O., "Sneak Analysis Evaluation: General Statement on 'Sneak' Approach and Practical Case Studies," Proceedings of the European Safety and Reliability Conference '92(ESRC '92), Copenhagen, Denmark, June 10-12, 1992, pp.1147-1155.

Chelini, J.V., "A Discussion on the Ada Run-Time Environment in Safety Critical Applications," ACM SIGSOFT Software Engineering Notes, Vol.17, No.4, pp.24-27, (Oct. 1992).

Chokhani, S., "Trusted Products Evaluation," Communications of the ACM, Vol.35, No.7, pp.64-76, (July 1992).

Chudleigh, M., "Developing Software for Safety-Critical Applications," The Nuclear Engineer, Vol.30, No.1, pp.14-21, (1989).

Chudleigh, M., "Software and Safety: How Compatible are They?," Information and Software Technology, Vol.32, No.5, pp.323-329, (June 1990).

Cobb, R.H. and Mills, H.D., "Engineering Software under Statistical Quality Control," IEEE Software, Vol.7, No.6, pp.44-54, (Nov. 1990).

Cooling, J.E. and Hughes, T.S., "Animation Prototyping of Real-Time Systems Specifications," The 5th Annual European Computer Conference: Advanced Computer Technology, Reliable Systems and Applications(CompEuro '91), 1991, pp.562-566.

Cooling, J.E., Software Design for Real-Time Systems, London: Chapman and Hall, Reading, 1991, pp.506.

Craigen, D. and Summerskill, K.(eds), Formal Methods for Trustworthy Computer Systems(FM89); Report from FM89: A Workshop on the Assessment of Formal Methods for Trustworthy Computer Systems held July 23-27, 1989 in Halifax, Canada, London: Springer-Verlag, Reading, 1990, pp.248.

Craigen, D., "FM89: Assessment of Formal Methods for Trustworthy Computer Systems," ICSE-12, Nice, Mar. 1990, pp.233-235.

Craigen, D., "Tool Support for Formal Methods," ICSE-13, Austin, May 1991, pp.184-185.

Cristian, F., "Correct and Robust Programs," IEEE Transactions on Software Engineering, Vol.SE-10, No.2, pp.163-174, (Mar. 1984).

Currit, P.A., Dyer, M., and Mills, H.D., "Certifying the Reliability of Software," IEEE Transactions on Software Engineering, Vol.SE-12, No.1, pp.3-11, (Jan. 1986). (Currit, P.A., Dyer, M., and Mills, H.D., "Correction to 'Certifying the Reliability of Software'," IEEE Transactions on Software Engineering, Vol.SE-15, No.3, p.362, (Mar. 1989)).

Dahll, G., Mainka, U., and Martz, J., "Tools for the Standardised Software Safety Assessment(The SOSAT Project)," IFAC SAFECOMP '88, Nov. 1988, pp.1-6.

Dale, C., "Software Safety Certification in Potentially Hazardous Industries," in Software Certification, ed. by de Neumann, B., London: Elsevier Applied Science, Reading, 1989, pp.100-112.

Dandanell, B., "Rigorous Development Using RAISE," ACM SIGSOFT Software Engineering Notes, Vol.16, No.5, pp.29-43, (Dec. 1991). Also available in Proceedings of the ACM SIGSOFT '91 Conference on Software for Critical Systems, New Orleans, Louisiana, December 4-6, 1991.

Daniels, B.K.(ed), Safety and Reliability of Programmable Electronic Systems: Proceedings of the Programmable Electronic Systems Safety Symposium(PES-3) held May 28-30, 1986 in Channel Islands, UK, London: Elsevier Applied Science, Reading, 1986, pp.270.

Daniels, B.K., "Guideline Framework for the Assessment of PES," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.41-50.

Daniels, B.K., "Harmonisation of Safety Standards for PES," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.251-262.

Daughtrey, H.T., Levinson, S.H., and Kimmel, D.M., "Developing a Standard for the Qualification of Software for Safety System Applications," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.687-692.

Davis, P.I., "Certification of Safety-Critical Software by Licensed Software Engineers," IEEE Computer, Vol.25, No.12, pp.72-73, (Dec. 1992).

Deckers, J. and Schabe, H., "The Sneak Circuit Analysis - An Investigation Method for Aerospace Systems," Proceedings of the European Safety and Reliability Conference '92(ESRC '92), Copenhagen, Denmark, June 10-12, 1992, pp.607-618.

Denning, P.J., "Human Error and the Search for Blame," Communications of the ACM, Vol.33, No.1, pp.6-7, (Jan. 1990).

Dobson, J., Laprie, J.-C., and Randell, B., "Predictably Dependable Computing Systems: An ESPRIT Basic Research Action," in Software Reliability and Metrics, ed. by Fento, N. and Littlewood, B., London: Elsevier Applied Science, Reading, 1991, pp.111-131.

DOD-STD-2167A: Military Standard Defense System Software Development, Department of Defense, Washington, D.C., 29 Feb. 1988.

DTI and NCC, The STARTS Guide - A Guide to Methods and Software Tools for the Construction of Large Real-Time Systems, Manchester: NCC Publications, 2nd ed(2 vols), 1987, pp.496.

DTI and NCC, The STARTS Purchasers' Handbook - Procuring Software-Based Systems, Manchester: NCC Publications, 2nd ed, May 1989.

Duke, E.L., "V&V of Flight and Mission-Critical Software," IEEE Software, Vol.6, No.3, pp.39-45, (May 1989).

Dunham, J.R. and Finelli, G.B., "Real-Time Software Failure Characterization," COMPASS '90, June 1990, pp.39-45.

Dunham, J.R., "Experiments in Software Reliability: Life-Critical Applications," IEEE Transactions on Software Engineering, Vol.SE-12, No.1, pp.110-123, (Jan. 1986).

Duran, J.W. and Ntafos, S.C., "An Evaluation of Random Testing," IEEE Transactions on Software Engineering, Vol.SE-10, No.4, pp.438-444, (July 1984).

Eagle, K.H. and Agarwala, A.S., "Redundancy Design Philosophy for Catastrophic Loss Protection," Proceedings of the Annual Reliability and Maintainability Symposium 1992, Las Vegas, Nevada, Jan. 1992, pp.1-4.

Eckhardt, D.E. and Lee, L.D., "A Theoretical Basis for the Analysis of Multiversion Software Subject to Coincident Errors," IEEE Transactions on Software Engineering, Vol.SE-11, No.12, pp.1511-1517, (Dec. 1985).

Ehrenberger, W., Roan, A., and Robert, P., "Software Certification Programme in Europe - SCOPE," in Software Reliability and Metrics, ed. by Fento, N. and Littlewood, B., London: Elsevier Applied Science, Reading, 1991, pp.132-148.

Erb, A., "Safety Measures of the Electronic Interlocking System 'ELEKTRA'," IFAC SAFECOMP '89, Dec. 1989, pp.49-52.

Fetzer, J.H., "Program Verification: The Very Idea," Communications of the ACM, Vol.31, No.9, pp.1048-1063, (Sept. 1988).

Fields, B. and Elvang-Goransson, M., "A VDM Case Study in MURAL," IEEE Transactions on Software Engineering, Vol.18, No.4, pp.279-295, (Apr. 1992).

Finnie, B.W. and Johnston, I.H.A., "Practical Experience in the Assessment of Existing Safety Critical Computer Based Systems," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.95-98.

Fraser, M.D., Kumar, K., and Vaishnavi, V.K., "Informal and Formal Requirements Specification Languages: Bridging the Gap," IEEE Transactions on Software Engineering, Vol.17, No.5, pp.454-466, (May 1991).

Friend, J.R., "A Review of Computer Controlled Systems Safety and Quality Assurance Concerns for Acquisition Managers," COMPASS '92, June 1992, pp.105-122.

Froome, P.K.D. and Monahan, B.Q., "Formal Methods for the Development and Assessment of Microprocessor Based Protection Systems," in Microprocessor Based Protection Systems, ed. by Churchley, A., London: Elsevier Applied Science, Reading, 1991, pp.133-152.

Fryer, M.O., "Risk Assessment of Computer Controlled Systems," IEEE Transactions on Software Engineering, Vol.SE-11, No.1, pp.125-129, (Jan. 1985).

Gabrielian, A. and Franklin, M.K., "Multilevel Specification of Real-Time Systems," Communications of the ACM, Vol.34, No.5, pp.50-60, (May 1991).

Gantenbein, R.E., "An Annotated Bibliography of Dependable Distributed Computing," ACM SIGOPS Operating Systems Review, Vol.26, No.2, pp.60-81, (Apr. 1992).

Geary, K., "Beyond Good Practices - A Standard for Safety Critical Software," in Achieving Safety and Reliability with Computer Systems, ed. by Daniels, B.K., Proceedings of the Safety and Reliability Society Symposium(SARSS '87) held November 11-12, 1987 in Altrincham, Manchester, UK, London: Elsevier Applied Science, Reading, 1987, pp.232-241.

Gerhart, S., "Formal Methods: An International Perspective," ICSE-13, Austin, May 1991, pp.36-37.

Gerhart, S., "Preliminary Summary: FM89 Assessment of Formal Methods for Trustworthy Computer Systems," Proceedings of the ACM SIGSOFT '89: The Third Symposium on Software Testing, Analysis, and Verification(TAV3), Key West, Florida, Dec. 13-15, 1989, pp.152-156. Also available in ACM SIGSOFT Software Engineering Notes, Vol.14, No.8, pp.152-155, (Dec. 1989).

Gerhart, S.L., "Applications of Formal Methods: Developing Virtuos Software," IEEE Software, Vol.7, No.5, pp.7-10, (Sept. 1990).

Goddard, E.O. and Zufferey, C.H., Report by the Technical Committee: Cross-Acceptance of Vital Signalling Systems, The Institution of Railway Signal Engineers, Mar. 1992, pp.10.

Goldberg, J., "Some Principles and Techniques for Designing Safe Systems," ACM

SIGSOFT Software Engineering Notes, Vol.12, No.3, pp.17-19, (July 1987).

Gonzalez Sanz, G., "Standardization for Safety Software: Current Status and Perspectives," The Fourth Software Engineering Standards Application Workshop(SESAW '91), San Diego, May 1991, pp.84-105.

Gove, R.A. and Heinzman, J.L., "Safety Criteria and Model for Mission-Critical Embedded Software Systems," COMPASS '91, June 1991, pp.69-73.

Gray, E.M. and Thayer, R.H., "Requirements," in Aerospace Software Engineering: A Collection of Concepts, ed. by Anderson, C. and Dorfman, M., Washington, DC: AIAA, Reading, 1991, pp.89-122.

Greenberg, R., "Software Safety Using FTA Technique," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.86-95.

Grimm, K., "An Effective Strategy and Automation Concepts for Systematic Testing of Safety Related Software," IFAC SAFECOMP '89, Dec. 1989, pp.71-79.

Gruman, G., "Software Safety Focus of New British Standard," IEEE Software, Vol.6, No.3, pp.95-96, (May 1989).

Guarro, S.B., Wu, J.S., Apostolakis, G.E., and Yau, M., "Embedded System Reliability and Safety Analysis in the UCLA ESSAE Project," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.383-388.

Guiho, G. and Hennebert, C., "SACEM Software Validation," ICSE-12, Nice, Mar. 1990, pp.186-191.

Halbwachs, N., Lagnier, F., and Ratel, C., "Programming and Verifying Real-Time Systems by Means of the Synchronous Data-Flow Language LUSTRE," IEEE Transactions on Software Engineering, Vol.18, No.9, pp.785-793, (Sept. 1992).

Hall, A., "Seven Myths of Formal Methods," IEEE Software, Vol.7, No.5, pp.11-19, (Sept. 1990).

Hansen, M.D. and Watts, R.L., "Software System Safety and Reliability," Proceedings of the Annual Reliability and Maintainability Symposium 1988, Los Angeles, CA, Jan. 1988, pp.214-217.

Helps, K.A., "Some Verification Tools and Methods for Airborne Safety- Critical Software," Software Engineering Journal, Vol.1, No.6, pp.248-253, (Nov. 1986).

Heninger, K., "Specifying Software Requirements for Complex Systems: New Techniques and their Applications," IEEE Transactions on Software Engineering, Vol.SE-6, No.1, pp.2-13, (Jan. 1980).

Hill, J.V., "Software Development Methods in Practice," in Microprocessor Based Protection Systems, ed. by Churchley, A., London: Elsevier Applied Science, Reading, 1991, pp.101-118.

Hill, J.V., "The Development of High Reliability Software - RR&A's Experience for Safety Critical Systems," BCS/IEE Conference on Software Engineering, Liverpool, 1988, pp.169-172.

Horstmann, C., "Arguing Against Certification," Communications of the ACM, Vol.34, No.10, p.13, (Oct. 1991).

Hughes, T.S. and Cooling, J.E., "Real-Time Systems - Animation Prototyping of Formal Specifications," The Third IEE/BCS International Conference on Software Engineering for Real Time Systems, Cirencester, Sept. 1991, pp.51-56.

IAEA, Manual on Quality Assurance for Computer Software Related to the Safety of Nuclear Power Plants, International Atomic Energy Agency, Technical Reports Series No.282, Vienna, 1988, pp.104.

IEC Draft Standard: Functional Safety of Programmable Electronic Systems: Generic Aspects; Part 1: General Requirements, IEC Reference:65A(Secretariat) 96, Sept. 1989, pp.104.

IEC Draft Standard: Software for Computers in the Application of Industrial Safety-Related Systems, IEC Reference:65A(Secretariat)94, Aug. 1989, pp.165.

ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety Related Systems, IEC Reference:65A(Secretariat)122, Nov. 1991, pp.88.

Jackson, T., "Practical RAMCAD," 1987 Proc. Tech. Meet. (Institute of Environmental Sciences, USA), Vol.33, pp.134-140.

Jacky, J., "Inside Risk: Risks in Medical Electronics," Communications of the ACM, Vol.33, No.12, p.138, (Dec. 1990).

Jaffe, M.S. and Leveson, N.G., "Completeness, Robustness, and Safety in Real-Time Software Requirements Specification," ICSE-11, Pittsburgh, Pennsylvania, May 1989, pp.302-311.

Jaffe, M.S., Leveson, N.G., Heimdahl, M.P.E., and Melhart, B.E., "Software Requirements Analysis for Real-Time Process-Control Systems," IEEE Transactions on Software Engineering, Vol.17, No.3, pp.241-258, (Mar. 1991).

Jahanian, F. and Moke, A.K., "Safety Analysis of Timing Properties in Real-Time Systems," IEEE Transactions on Software Engineering, Vol.SE-12, No.9, pp.890-904, (Sept. 1986).

Jennings, N., "Aerospace Software Engineering in the United Kingdom," in Aerospace Software Engineering: A Collection of Concepts, ed. by Anderson, C.

and Dorfman, M., Washington, DC: AIAA, Reading, 1991, pp.545-559.

Jesty, P.H., Lamb, D.A., Buckley, T.F., and West, M.M., "Choice of Design Methodologies for Demanding High Integrity Industrial Control Systems," The Third IEE/BCS International Conference on Software Engineering for Real Time Systems, Cirencester, Sept. 1991, pp.16-21.

Johnston, I.H.A. and Wood, A.P., "Integrity Levels and Assessment Levels in Practice," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.70-74.

Johnston, I.H.A., "The Impact of Social Factors on Acceptable Levels of Safety Integrity," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.157-161.

Keene, S.J., Jr., "Assuring Software Safety," Proceedings of the Annual Reliability and Maintainability Symposium 1992, Las Vegas, Nevada, Jan. 1992, pp.274-279.

Kelly, J.P.J., McVittie, T.I., and Yamamoto, W.I., "Implementing Design Diversity to Achieve Fault Tolerance," IEEE Software, Vol.8, No.4, pp.61-71, (July 1991).

Kersken, M, "Interaction of Man and Tools During Verification and Validation of Safety Critical Software," Proceedings of the Topical Meeting on Advances in Human Factors Research on Man/Computer Interactions: Nuclear and Beyond, June 1990, pp.47-52.

Kersken, M. and Saglietti, F.(eds), Software Fault Tolerance: Achievement and Assessment Strategies, Research Reports: ESPRIT Project 300 REQUEST (Reliability and Quality of European Software Technology), Vol.1, Berlin: Springer-Verlag, Reading, 1991, pp.243.

Klein, P., "The Safety-Bag Expert System in the Electronic Railway Interlocking System Elektra," Expert Systems With Applications, Vol.3, No.4, pp.499-506, (1991).

Knight, J.C. and Ammann, P.E., "Issues Influencing the Use of N-Version Programming," Proceedings of the IFIP 11th World Computer Congress, San Francisco, Calif., Aug./Sept. 1989, pp.217-222.

Knight, J.C. and Leveson, N.G., "An Experimental Evaluation of the Assumption of Independence in Multi-Version Programming," IEEE Transactions on Software Engineering, Vol.SE-12, No.1, pp.96-109, (Jan. 1986).

Kriewall, T.J. and Long, J.M., "Computer-Based Medical Systems," IEEE Computer, Vol.24, No.3, pp.9-12, (Mar. 1991).

Lafontaine, C., Ledru, Y., and Schobbens, P.-Y., "An Experiment in Formal Software Development Using the B Theorem Prover on a VDM Case Study,"

Communications of the ACM, Vol.34, No.5, pp.62-87, (May 1991).

Laprie, J.-C. and Littlewood, B., "Probabilistic Assessment of Safety- Critical Software: Why and How," Communications of the ACM, Vol.35, No.2, pp.13,16,21, (Feb. 1992).

Laprie, J.-C.(ed), Dependability: Basic Concepts and Terminology in English, French, German, Italian and Japanese, Wien: Springer-Verlag, Dependable Computing and Fault-Tolerant Systems, Vol.5, Reading, 1992, pp.265.

Laprie, J.-C., "Hardware-and-Software Dependability Evaluation," Proceedings of the IFIP 11th World Computer Congress, San Francisco, Calif., Aug./Sept. 1989, pp.109-114.

Laprie, J.-C., "On the Assessment of Safety-Critical Software Systems," ICSE-12, Nice, Mar. 1990, p.222.

Laprie, J.-C., Arlat, J., Beounes, C., and Kanoun, K., "Definition and Analysis of Hardware- and Software-Fault-Tolerant Architectures," IEEE Computer, Vol.23, No.7, pp.39-51, (July 1990).

Lee, P.A. and Anderson, T., Fault Tolerance - Principles and Practice, Second, revised edition, Wien: Springer-Verlag, Dependable Computing and Fault-Tolerant Systems, Vol.3, Reading, 1990, pp.320.

Leveson, N.G. and Harvey, P.R., "Analyzing Software Safety," IEEE Transactions on Software Engineering, Vol.SE-9, No.5, pp.569-579, (Sept. 1983).

Leveson, N.G. and Stolzy, J.L., "Safety Analysis of Ada Programs Using Fault Trees," IEEE Transactions on Reliability, Vol.R-32, No.5, pp.479-484, (Dec. 1983).

Leveson, N.G. and Stolzy, J.L., "Safety Analysis Using Petri Nets," IEEE Transactions on Software Engineering, Vol.SE-13, No.3, pp.386-397, (Mar. 1987).

Leveson, N.G., "An Outline of a Program to Enhance Software Safety," IFAC SAFECOMP '86, Oct. 1986, pp.129-135.

Leveson, N.G., "Building Safe Software," in Software Reliability: Achievement and Assessment, ed. by Littlewood, B., Oxford: Blackwell Scientific Publications, Reading, 1987, pp.1-18.

Leveson, N.G., "Evaluation of Software Safety," ICSE-12, Nice, Mar. 1990, pp.223-224.

Leveson, N.G., "Safety as a Software Quality," IEEE Software, Vol.6, No.3, pp.88-89, (May 1989).

Leveson, N.G., "Safety Assessment and Management Applied to Software," in

Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.377-382.

Leveson, N.G., "Safety," in Aerospace Software Engineering: A Collection of Concepts, ed. by Anderson, C. and Dorfman, M., Washington, DC: AIAA, Reading, 1991, pp.319-337.

Leveson, N.G., "Safety-Critical Software Development," in Safe and Secure Computing Systems, ed. by Anderson, T., Oxford: Blackwell Scientific Publications, Reading, 1989, pp.155-162.

Leveson, N.G., "Software Hazard Analysis Techniques," in Software System Design Methods: The Challenge of Advanced Computing Technology, ed. by Skwirzynski, J.K., Berlin: Springer-Verlag, Reading, 1986, pp.681-699.

Leveson, N.G., "Software Safety in Computer-Controlled Systems," IEEE Computer, Vol.17, No.2, pp.48-55, (Feb. 1984).

Leveson, N.G., "Software Safety in Embedded Computer Systems," Communications of the ACM, Vol.34, No.2, pp.34-46, (Feb. 1991).

Leveson, N.G., "Software Safety: Why, What, and How," ACM Computing Surveys, Vol.18, No.2, pp.125-163, (June 1986). (金岡訳,「ソフトウェアの安全性：なぜ, 何か, いかに」, bit別冊 ACM COMPUTING SURVEYS '86 コンピュータ・サイエンス, 共立出版, 1988年5月号別冊, pp.21-54.)

Leveson, N.G., "The Challenge of Building Process-Control Software," IEEE Software, Vol.7, No.6, pp.55-62, (Nov. 1990).

Leveson, N.G., Cha, S.S., and Shimeall, T.J., "Safety Verification of Ada Programs Using Software Fault Trees," IEEE Software, Vol.8, No.4, pp.48-59, (July 1991).

Leveson, N.G., Cha, S.S., Knight, J.C., and Shimeall, T.J., "The Use of Self Checks and Voting in Software Error Detections: An Empirical Study," IEEE Transactions on Software Engineering, Vol.SE-16, No.4, pp.432-443, (Apr. 1990).

Leveson, N.G., "High-Pressure Steam Engines and Computer Software," ICSE-14, Melbourne, Australia, May 1992, pp.2-14.

Lyu, M.R., "Assuring Design Diversity in N-Version Software: A Design Paradigm for N-Version Programming," in Dependable Computing for Critical Applications 2, ed. by Meyer, J.F. and Schlichting, R.D., Wien: Springer-Verlag, Reading, 1992, pp.197-218.

MacMillan, K.L., "Software Safety Analysis," Proceedings of the Seventh Annual Conference of the IEEE/Engineering in Medicine and Biology Society, Sept. 1985, pp.1222-1229.

Mahony, B.P. and Hayes, I.J., "A Case-Study in Timed Refinement: A Mine Pump," IEEE Transactions on Software Engineering, Vol.18, No.9, pp.817-826, (Sept. 1992).

McDermid, J.A., "Issues in Developing Software for Safety Critical Systems," Reliability Engineering and System Safety, Vol.32, Nos.1/2, pp.1-24, (1991). Also available in Software Reliability and Safety, ed. by Littlewood, B. and Miller, D., London: Elsevier Applied Science, Reading, 1991, pp.1-24.

McDermid, J.A., "Safety Arguments, Software and System Reliability," Proceedings of the 1991 International Symposium on Software Reliability Engineering(ISSRE '91), Austin, Texas, May 17-18, 1991, pp.43-50.

McFarland, M.C., "Ethics and the Safety of Computer Systems," IEEE Computer, Vol.24, No.2, pp.72-75, (Feb. 1991).

McKinlay, A.VI, "Software Safety Handbook," COMPASS '89, June 1989, pp.14-17.

McKinlay, A.VI., "The State of Software Safety Engineering," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.369-376.

Meffert, K., "Standardisation for Computer Safety - The Current Situation in Germany," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.247-250.

Melford, R.J., "Development to Begin on Standards of Ethical Practices for Computing Professionals," IEEE Computer, Vol.25, No.4, pp.76-78, (Apr. 1992).

Meyer, B., "On Formalism in Specifications," IEEE Software, Vol.2, No.1, pp.6-26, (Jan. 1985).（羅，佐伯訳，「仕様をきちんと書くには」，IEEEソフトウェア '88, 岩波書店，1988年8月，pp.139-154.）

Meyer, J.F. and Schlichting, R.D.(eds), Dependable Computing for Critical Applications 2, Wien: Springer-Verlag, Dependable Computing and Fault-Tolerant Systems, Vol.6, Reading, 1992, pp.437.

MIL-STD-1521B(USAF): Military Standard Technical Reviews and Audits for Systems, Equipments, and Computer Software, Department of Defense, Washington, D.C., 4 June 1985.

MIL-STD-480B: Military Standard Configuration Control - Engineering Changes, Deviations and Waivers, Department of Defense, Washington, D.C., 15 July 1988.

MIL-STD-882B: Military Standard System Safety Program Requirements, Department of Defense, Washington, D.C., 30 Mar. 1984.

Miller, D.R., "The Role of Statistical Modeling and Inference in Software Quality Assurance," in Software Certification, ed. by de Neumann, B., London: Elsevier Applied Science, Reading, 1989, pp.135-152.

Miller, L.A., "Dynamic Testing of Knowledge Bases Using the Heuristic Testing Approach," Expert Systems With Applications, Vol.1, No.3, pp.249-269, (1990).

Mills, H.D., "Structured Programming: Retrospect and Prospect," IEEE Software, Vol.3, No.6, pp.58-66, (Nov. 1986). (近藤, 前沢訳, 「構造化プログラミング ― 回想と展望」, IEEEソフトウェア '88, 岩波書店, 1988年8月, pp.93-103.)

Mills, H.D., Dyer, M., and Linger, R.C., "Cleanroom Software Engineering," IEEE Software, Vol.4, No.5, pp.19-25, (Sept. 1987).

Nelson, V.P., "Fault-Tolerant Computing: Fundamental Concepts," IEEE Computer, Vol.23, No.7, pp.19-25, (July 1990).

Neumann, P.G.(ed), "Risks to the Public in Computers and Related Systems: Subsection on Certification of Professionals," ACM SIGSOFT Software Engineering Notes, Vol.16, No.1, pp.24-32, (Jan. 1991).

Neumann, P.G.(ed), "Risks to the Public in Computers and Related Systems: Subsection on Safety Standards," ACM SIGSOFT Software Engineering Notes, Vol.16, No.3, p.28, (July 1991).

Neumann, P.G., "Inside Risk: Are Dependable Systems Feasible?," Communications of the ACM, Vol.36, No.2, p.146, (Feb. 1993).

Neumann, P.G., "Inside Risk: Certifying Professionals," Communications of the ACM, Vol.34, No.2, p.130, (Feb. 1991).

Neumann, P.G., "On Hierarchical Design of Computer Systems for Critical Applications," IEEE Transactions on Software Engineering, Vol.SE-12, No.9, pp.905-920, (Sept. 1986).

Neumann, P.G., "Safety of Computer Systems in Aerospace Applications," Proceedings of AIAA Computers in Aerospace Conference Ⅶ, Oct. 1989, pp.970-974.

Neumann, P.G., "Inside Risk: The Human Element," Communications of the ACM, Vol.34, No.11, p.150, (Nov. 1991).

Neumann, P.G., "Inside Risk: Where to Place Trust," Communications of the ACM, Vol.35, No.10, p.138, (Oct. 1992).

Norman, D.A., "Commentary: Human Error and the Design of Computer Systems," Communications of the ACM, Vol.33, No.1, pp.4-5,7, (Jan. 1990).

Norman, D.A., "Design Rules Based on Analyses of Human Error," Communications of the ACM, Vol.26, No.4, pp.254-258, (Apr. 1983).

Ntafos, S.C., "A Comparison of Some Structural Testing Strategies," IEEE Transactions on Software Engineering, Vol.SE-14, No.6, pp.868-874, (June 1988).

Parnas, D.L. and Clements, P.C., "A Rational Design Process: How and Why to Fake It," IEEE Transactions on Software Engineering, Vol.SE-12, No.2, pp.251-257, (Feb. 1986).

Parnas, D.L. and Weiss, D.M., "Active Design Reviews: Principles and Practices," ICSE-8, London, Aug. 1985, pp.132-136.

Parnas, D.L., "Education for Computing Professionals," IEEE Computer, Vol.23, No.1, pp.17-22, (Jan. 1990).

Parnas, D.L., "Software Aspects of Strategic Defense Systems," American Scientist, Sept./Oct. 1985, pp.432-440.（高榎訳,「ＳＤＩは不可能である - ソフトウェア工学からの視点 - 」, 世界, No.489, pp.298-319, (1986年6月号), 岩波書店.）

Parnas, D.L., van Schouwen, A.J., and Kwan, S.P., "Evaluation of Safety-Critical Software," Communications of the ACM, Vol.33, No.6, pp.636-648, (June 1990).

Pearson, J., "Proposed HSE Guidelines on Emergency Shutdown Systems," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.55-58.

Peters, J.F., III and Ramanna, S., "Verifying Command Sequences for Satellite Systems," IEEE AES Systems Magazine, Vol.7, No.10, pp.14-21, (Oct. 1992).

Petrella, S., Michael, P., Bowman, W.C., and Lim, S.T., "Random Testing of Reactor Shutdown System Software," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.681-686.

Petroski, H., To Engineer is Human: The Role of Failure in Successful Design, New York: St. Martin's Press, 1985.（北村訳, 人はだれでもエンジニア：失敗はいかに成功のもとになるか, 鹿島出版会, 1988年4月, pp.290.）

Pilaud, E., "Some Experiences of Critical Software Development," ICSE-12, Nice, Mar. 1990, pp.225-226.

Pilsworth, R., "Defence Standards 00-55 & 00-56," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.103-105.

Potocki De Montalk, J.P., "Computer Software in Civil Aircraft," COMPASS '91, June 1991, pp.10-16.

Potter, B., Sinclair, J., and Till, D., An Introduction to Formal Specification and Z, London: Prentice Hall International, Reading, 1991, pp.304.

Probert, P.J. and Lamb, D., "Validation Methods Using Formal Techniques," The Third IEE/BCS International Conference on Software Engineering for Real Time Systems, Cirencester, Sept. 1991, pp.244-249.

Pyle, I.C., Developing Safety Systems: A Guide Using Ada, London: Prentice Hall International, Reading, 1991, pp.254.

Ramamoorthy, C.V., Prakash, A., Tsai, W-T., and Usuda, Y., "Software Engineering: Problems and Perspectives," IEEE Computer, Vol.17, No.10, pp.191-209, (Oct. 1984).

Rapps, S. and Weyuker, E.J., "Selecting Software Test Data Using Data Flow Information," IEEE Transactions on Software Engineering, Vol.SE-11, No.4, pp.367-375, (Apr. 1985).

Ravn, A.P., Rischel, H., and Stavridou, V., "Provably Correct Safety Critical Software," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.13-18.

Reed, W., "Safety Critical Software in Traffic Control Systems," IEE Colloquium on 'Safety Critical Software in Vehicle and Traffic Control', London, Feb. 1990, pp.2/1-2/5.

Reibman, A.L. and Veeraraghavan, M., "Reliability Modeling: An Overview for System Designers," IEEE Computer, Vol.24, No.4, pp.49-57, (Apr. 1991).

Rettig, M., "Testing Made Palatable," Communications of the ACM, Vol.34, No.5, pp.25-29, (May 1991).

Rouse, W.B., "Human-Computer Interaction in the Control of Dynamic Systems," ACM Computing Surveys, Vol.13, No.1, pp.71-99, (Mar.1981).

Rushby, J., "Validation and Testing of Knowledge-Based Systems: How Bat Can It Get?," The AAAI-88 Workshop on Verification and Validation of Expert Systems, Dec. 1988, pp.1-9.

Sadeghi, T. and Motamed, F., "Evaluation and Comparison of Triple and Quadruple Flight Control Architectures," IEEE AES Systems Magazine, Vol.7, No.3, pp.20-31, (Mar. 1992).

Saeed, A., Anderson, T., and Koutny, M., "A Formal Model for Safety- Critical Computing Systems," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.1-6.

Saeed, A., de Lemos, R., and Anderson, T., "The Role of Formal Methods in the Requirements Analysis of Safety-Critical Systems: A Train Set Example," Digest of Papers Fault-Tolerant Computing: The Twenty-First International Symposium(FTCS-21), Montreal, Canada, June 25-27, 1991, pp.478-485.

Sayet, C. and Pilaud, E., "An Experience of a Critical Software Development," Digest of Papers Fault-Tolerant Computing: The Twentieth International Symposium(FTCS-20), Newcastle upon Tyne, England, June 26-28, 1990, pp.36-45.

Schneidewind, N.F.(ed), "Are Software Engineering Process Standards Really Necessary?," IEEE Computer, Vol.25, No.11, pp.82-84, (Nov. 1992).

Schoitsch, E., "The Interaction between Practical Experience, Standardization and the Application of Standards," IFAC SAFECOMP '89, Dec. 1989, pp.17-24.

Schoitsch, E., Dittrich, E., Grasegger, S., Kropfitsch, D., Erb, A., Fritz, P., and Kopp, H., "The ELEKTRA Testbed: Architecture of a Real-Time Test Environment for High Safety and Reliability Requirements," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.59-65.

SEB6-A(EIA): System Safety Engineering in Software Development, Electronic Industries Association, EIA Safety Engineering Bulletin No.6-A, Apr. 1990.

Selby, R.W., Basili, V.R., and Baker, F.T., "Cleanroom Software Development: An Empirical Evaluation," IEEE Transactions on Software Engineering, Vol.SE-13, No.9, pp.1027-1037, (Sept. 1987).

Sethy, A., "Connection Between Reliability and Signalling-Safety in Railway Technology," Proceedings of the Annual Reliability and Maintainability Symposium 1992, Las Vegas, Nevada, Jan. 1992, pp.75-79.

Shagnea, A.M., Hayhurst, K.J., and Withers, B.E., "Data Collection and Descriptive Analysis: A First Step for Developing Quality Software," COMPASS '91, June 1991, pp.173-179.

Shaw, M., "Abstraction Techniques in Modern Programming Languages," IEEE Software, Vol.1, No.4, pp.10-26, (July 1984).（毛利，林訳，「プログラミング言語と抽象化」，IEEEソフトウェア '88，岩波書店，1988年8月，pp.47-62.）

Shimeall, T.J. and Leveson, N.G., "An Empirical Comparison of Software Fault Tolerance and Fault Elimination," IEEE Transactions on Software Engineering, Vol.17, No.2, pp.173-182, (Feb. 1991).

Shimeall, T.J., McGraw, R.J., Jr., Gill, J.A., "Software Safety Analysis in Heterogeneous Multiprocessor Control Systems," Proceedings of the Annual Reliability and Maintainability Symposium 1991, Orlando, Florida, Jan. 1991, pp.290-294.

Shore, J., "Why I Never Met a Programmer I Could Trust," Communications of the ACM, Vol.31, No.4, pp.372-375, (Apr. 1988).

Singh, A.D. and Murugesan, S., "Fault-Tolerant Systems," IEEE Computer, Vol.23, No.7, pp.15-17, (July 1990).

Smith, I.C., Wall, D.N., and Baldwin, J.A., "DARTS - An Experiment into Cost of and Diversity in Safety Critical Computer Systems," IFAC SAFECOMP '91, Oct./Nov. 1991, pp.35-40.

Sparkman, D., "Standards and Practices for Reliable Safety-Related Software Systems," Proceedings of the Third International Symposium on Software Reliability Engineering(ISSRE '92), Research Triangle Park, North Carolina, Oct. 7-10, 1992, pp.318-327.

Spivey, J.M., "Specifying a Real-Time Kernel," IEEE Software, Vol.7, No.5, pp.21-28, (Sept. 1990).

Stavridou,V. and Ravn,A., "A Comment on the Revised UK MOD Standards 00-55/56," ACM SIGSOFT Software Engineering Notes, Vol.16, No.4, p.22, (Oct. 1991).

Strandberg, K., "IEC 300: The Dependability Counterpart of ISO 9000," Proceedings of the Annual Reliability and Maintainability Symposium 1991, Orlando, Florida, Jan. 1991, pp.463-467.

Taylor, J.A., "Training, Career Development and Registration for Safety Critical Software Systems Specialists," COMPASS '91, June 1991, pp.29-35.

Thevenod-Fosse, P., "Software Validation by Means of Statistical Testing: Retrospect and Future Direction," in Dependable Computing for Critical Applications, ed. by Avizienis, A. and Laprie, J.-C., Wien: Springer-Verlag, Reading, 1991, pp.23-50.

Thomas, M., "Assessing Failure Probabilities in Safety-Critical Systems Containing Software," ICSE-12, Nice, Mar. 1990, p.227.

Thomas, M., "Limits and Customers are Driving Change," IEEE Software, Vol.9, No.2, p.10, (Mar. 1992).

Toola, A., "Elicitation of Safety Requirements for Process Automation," Reliability Engineering and System Safety, Vol.35, No.3, pp.209-215, (1992), (Published in 1991).

Tripp, L.L., "What is the Future of Software Engineering Standards?," ACM SIGSOFT Software Engineering Notes, Vol.17, No.1, pp.58-61, (Jan. 1992).

Tripp, L.L., Struck, W.F., and Pflug, B.K., "The Application of Multiple Team Inspections on a Safety-Critical Software Standard," The Fourth Software Engineering Standards Application Workshop(SESAW '91), San Diego, May 1991, pp.106-111.

UK Interim Defence Standard 00-50: The Procurement of Safety Critical Software in Defence Equipment: Part 1: Requirements, Part 2: Guidance, Ministry of Defence, Directorate of Standardization, Kentigern House, 65 Brown Street, Glasgow G2 8EX, 5 April 1991.

UK Interim Defence Standard 00-56/1: Requirements for the Analysis of Safety Critical Hazards, Ministry of Defence, Directorate of Standardisation, Kentigern House, 65 Brown St., Glasgow G2 8EX, May 1989.

van Baal, J.B.J., "Hardware/Software FMEA Applied to Airplane Safety," Proceedings of the Annual Reliability and Maintainability Symposium 1985, 1985, pp.250-255.

Wallace, D.R., Kuhn, D.R., and Ippolito, L.M., "An Analysis of Selected Software Safety Standards," COMPASS '92, June 1992, pp.123-136. Also available in IEEE AES Systems Magazine, Vol.7, No.8, pp.3-14, (Aug. 1992).

Wallace, D.R., Kuhn, D.R., and Ippolito, L.M., "An Analysis of Selected Software Safety Standards," IEEE AES Systems Magazine, Vol.7, No.8, pp.3-14, (Aug. 1992). Also available in COMPASS '92, June 1992.

Ward, N.J., "The Static Analysis of Critical Software," in Microprocessor Based Protection Systems, ed. by Churchley, A., London: Elsevier Applied Science, Reading, 1991, pp.121-130.

Warwick, K. and Tham, M.T.(eds), Failsafe Control Systems: Applications and Emergency Management, London: Chapman and Hall, Reading, 1991, pp.246.

Wichmann, B.A.(ed), Software in Safety-Related Systems, Chichester: John Wiley & Sons, Reading, 1992, pp.304.

Wiener, L., "A Trip Report on SIGSOFT '91," ACM SIGSOFT Software Engineering Notes, Vol.17, No.2, pp.23-38, (Apr. 1992).

Wing, J.M., "A Specifier's Introduction to Formal Methods," IEEE Computer, Vol.23, No.9, pp.8-24, (Sept. 1990).

Wingrove, A.A., "Software Certification," in Software Reliability and Metrics, ed. by Fento, N. and Littlewood, B., London: Elsevier Applied Science, Reading, 1991, pp.192-216.

Wirthumer, G., "VOTRICS - Fault Tolerance Realized in Software," IFAC SAFECOMP '89, Dec. 1989, pp.135-140.

Wray, A.M., "Case Study Using the Guidelines Framework," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.51-62.

Wright, C.L. and Zawilski, A.J., "Existing and Emerging Standards for Software Safety," The Fourth Software Engineering Standards Application Workshop(SESAW '91), San Diego, May 1991, pp.112-118.

Zave, P., "An Operational Approach to Requirements Specifications for Embedded Systems," IEEE Transactions on Software Engineering, Vol.SE-8, No.3, pp.250-269, (May 1982).

浦野, 中村, "フォールトトレラントソフトウェア技術," 電子情報通信学会, Vol.73, No.5, pp.475-480, (May 1990).

当麻(監修), 向殿(編集), コンピュータシステムの高信頼化技術入門：フォールトトレランスシステムの基礎, 日本規格協会, 1988年3月, pp.249.

2．発行年別（1980年～1993年）

[1980]

Heninger, K., "Specifying Software Requirements for Complex Systems: New Techniques and their Applications," IEEE Transactions on Software Engineering, Vol.SE-6, No.1, pp.2-13, (Jan. 1980).

[1981]

Rouse, W.B., "Human-Computer Interaction in the Control of Dynamic Systems," ACM Computing Surveys, Vol.13, No.1, pp.71-99, (Mar.1981).

[1982]

Zave, P., "An Operational Approach to Requirements Specifications for Embedded Systems," IEEE Transactions on Software Engineering, Vol.SE-8, No.3, pp.250-269, (May 1982).

[1983]

Leveson, N.G. and Harvey, P.R., "Analyzing Software Safety," IEEE Transactions on Software Engineering, Vol.SE-9, No.5, pp.569-579, (Sept. 1983).

Leveson, N.G. and Stolzy, J.L., "Safety Analysis of Ada Programs Using Fault Trees," IEEE Transactions on Reliability, Vol.R-32, No.5, pp.479-484, (Dec. 1983).

Norman, D.A., "Design Rules Based on Analyses of Human Error," Communications of the ACM, Vol.26, No.4, pp.254-258, (Apr. 1983).

[1984]

MIL-STD-882B: Military Standard System Safety Program Requirements, Department of Defense, Washington, D.C., 30 Mar. 1984.

Avizienis, A. and Kelly, J.P.J., "Fault Tolerance by Design Diversity: Concepts and Experiments," IEEE Computer, Vol.17, No.8, pp.67-80, (Aug. 1984).

Cristian, F., "Correct and Robust Programs," IEEE Transactions on Software Engineering, Vol.SE-10, No.2, pp.163-174, (Mar. 1984).

Duran, J.W. and Ntafos, S.C., "An Evaluation of Random Testing," IEEE Transactions on Software Engineering, Vol.SE-10, No.4, pp.438-444, (July 1984).

Leveson, N.G., "Software Safety in Computer-Controlled Systems," IEEE Computer, Vol.17, No.2, pp.48-55, (Feb. 1984).

Ramamoorthy, C.V., Prakash, A., Tsai, W-T., and Usuda, Y., "Software Engineering:

Problems and Perspectives," IEEE Computer, Vol.17, No.10, pp.191-209, (Oct. 1984).

Shaw, M., "Abstraction Techniques in Modern Programming Languages," IEEE Software, Vol.1, No.4, pp.10-26, (July 1984).（毛利，林訳，「プログラミング言語と抽象化」，IEEEソフトウェア '88，岩波書店，1988年8月，pp.47-62.）

[1985]

Parnas, D.L. and Weiss, D.M., "Active Design Reviews: Principles and Practices," ICSE-8, London, Aug. 1985, pp.132-136.

Fryer, M.O., "Risk Assessment of Computer Controlled Systems," IEEE Transactions on Software Engineering, Vol.SE-11, No.1, pp.125-129, (Jan. 1985).

Rapps, S. and Weyuker, E.J., "Selecting Software Test Data Using Data Flow Information," IEEE Transactions on Software Engineering, Vol.SE-11, No.4, pp.367-375, (Apr. 1985).

Eckhardt, D.E. and Lee, L.D., "A Theoretical Basis for the Analysis of Multiversion Software Subject to Coincident Errors," IEEE Transactions on Software Engineering, Vol.SE-11, No.12, pp.1511-1517, (Dec. 1985).

Meyer, B., "On Formalism in Specifications," IEEE Software, Vol.2, No.1, pp.6-26, (Jan. 1985).（羅，佐伯訳，「仕様をきちんと書くには」，IEEEソフトウェア '88，岩波書店，1988年8月，pp.139-154.）

Brown, M.L., "Software Safety for Complex Systems," Proceedings of the Seventh Annual Conference of the IEEE/Engineering in Medicine and Biology Society, Sept. 1985, pp.210-216.

MacMillan, K.L., "Software Safety Analysis," Proceedings of the Seventh Annual Conference of the IEEE/Engineering in Medicine and Biology Society, Sept. 1985, pp.1222-1229.

Petroski, H., To Engineer is Human: The Role of Failure in Successful Design, New York: St. Martin's Press, 1985.（北村訳，人はだれでもエンジニア：失敗はいかに成功のもとになるか，鹿島出版会，1988年4月，pp.290.）

Parnas, D.L., "Software Aspects of Strategic Defense Systems," American Scientist, Sept./Oct. 1985, pp.432-440.（高榎訳，「ＳＤＩは不可能である - ソフトウェア工学からの視点 - 」，世界，No.489, pp.298-319, (1986年6月号)，岩波書店.）

van Baal, J.B.J., "Hardware/Software FMEA Applied to Airplane Safety," Proceedings of the Annual Reliability and Maintainability Symposium 1985, 1985, pp.250-255.

MIL-STD-1521B(USAF): Military Standard Technical Reviews and Audits for Systems, Equipments, and Computer Software, Department of Defense, Washington, D.C.,

4 June 1985.

AFISC SSH 1-1: System Safety Handbook Software System Safety, Department of the Air Force, Headquarters Air Force Inspection and Safety Center, 5 September 1985.

[1986]

Leveson, N.G., "Software Safety: Why, What, and How," ACM Computing Surveys, Vol.18, No.2, pp.125-163, (June 1986). (金岡訳,「ソフトウェアの安全性：なぜ, 何か, いかに」, bit別冊 ACM COMPUTING SURVEYS '86 コンピュータ・サイエンス, 共立出版, 1988年5月号別冊, pp.21-54.)

Currit, P.A., Dyer, M., and Mills, H.D., "Certifying the Reliability of Software," IEEE Transactions on Software Engineering, Vol.SE-12, No.1, pp.3-11, (Jan. 1986). (Currit, P.A., Dyer, M., and Mills, H.D., "Correction to 'Certifying the Reliability of Software'," IEEE Transactions on Software Engineering, Vol.SE-15, No.3, p.362, (Mar. 1989)).

Knight, J.C. and Leveson, N.G., "An Experimental Evaluation of the Assumption of Independence in Multi-Version Programming," IEEE Transactions on Software Engineering, Vol.SE-12, No.1, pp.96-109, (Jan. 1986).

Dunham, J.R., "Experiments in Software Reliability: Life-Critical Applications," IEEE Transactions on Software Engineering, Vol.SE-12, No.1, pp.110-123, (Jan. 1986).

Parnas, D.L. and Clements, P.C., "A Rational Design Process: How and Why to Fake It," IEEE Transactions on Software Engineering, Vol.SE-12, No.2, pp.251-257, (Feb. 1986).

Jahanian, F. and Moke, A.K., "Safety Analysis of Timing Properties in Real-Time Systems," IEEE Transactions on Software Engineering, Vol.SE-12, No.9, pp.890-904, (Sept. 1986).

Neumann, P.G., "On Hierarchical Design of Computer Systems for Critical Applications," IEEE Transactions on Software Engineering, Vol.SE-12, No.9, pp.905-920, (Sept. 1986).

Bishop, P.G., Esp, D.G., Barnes, M., Humphreys, P., Dahll, G., and Lahti, J., "PODS - A Project on Diverse Software," IEEE Transactions on Software Engineering, Vol.SE-12, No.9, pp.929-940, (Sept. 1986).

Abdel-Ghaly, A.A., Chan, P.Y., and Littlewood, B., "Evaluation of Competing Software Reliability Predictions," IEEE Transactions on Software Engineering, Vol.SE-12, No.9, pp.950-967, (Sept. 1986).

Mills, H.D., "Structured Programming: Retrospect and Prospect," IEEE Software, Vol.3, No.6, pp.58-66, (Nov. 1986). (近藤, 前沢訳,「構造化プログラミング － 回想と展望」, IEEEソフトウェア '88, 岩波書店, 1988年8月, pp.93-103.)

Leveson, N.G., "Software Hazard Analysis Techniques," in Software System Design Methods: The Challenge of Advanced Computing Technology, ed. by Skwirzynski, J.K., Berlin: Springer-Verlag, Reading, 1986, pp.681-699.

Daniels, B.K.(ed), Safety and Reliability of Programmable Electronic Systems: Proceedings of the Programmable Electronic Systems Safety Symposium(PES-3) held May 28-30, 1986 in Channel Islands, UK, London: Elsevier Applied Science, Reading, 1986, pp.270.

Daniels, B.K., "Guideline Framework for the Assessment of PES," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.41-50.

Wray, A.M., "Case Study Using the Guidelines Framework," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.51-62.

Greenberg, R., "Software Safety Using FTA Technique," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.86-95.

Meffert, K., "Standardisation for Computer Safety - The Current Situation in Germany," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.247-250.

Daniels, B.K., "Harmonisation of Safety Standards for PES," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.251-262.

Bell, R., "Guidance on the Use of Programmable Electronic Systems in Safety Related Applications," in Safety and Reliability of Programmable Electronic Systems, ed. by Daniels, B.K., London: Elsevier Applied Science, Reading, 1986, pp.263-270.

Leveson, N.G., "An Outline of a Program to Enhance Software Safety," IFAC SAFECOMP '86, Oct. 1986, pp.129-135.

Helps, K.A., "Some Verification Tools and Methods for Airborne Safety- Critical Software," Software Engineering Journal, Vol.1, No.6, pp.248-253, (Nov. 1986).

[1987]

Leveson, N.G. and Stolzy, J.L., "Safety Analysis Using Petri Nets," IEEE Transactions on Software Engineering, Vol.SE-13, No.3, pp.386-397, (Mar. 1987).

Mills, H.D., Dyer, M., and Linger, R.C., "Cleanroom Software Engineering," IEEE

Software, Vol.4, No.5, pp.19-25, (Sept. 1987).

Selby, R.W., Basili, V.R., and Baker, F.T., "Cleanroom Software Development: An Empirical Evaluation," IEEE Transactions on Software Engineering, Vol.SE-13, No.9, pp.1027-1037, (Sept. 1987).

Brooks, F.P., Jr., "No Silver Bullet - Essence and Accidents of Software Engineering," IEEE Computer, Vol.20, No.4, pp.10-19, (Apr. 1987).

Geary, K., "Beyond Good Practices - A Standard for Safety Critical Software," in Achieving Safety and Reliability with Computer Systems, ed. by Daniels, B.K., Proceedings of the Safety and Reliability Society Symposium(SARSS '87) held November 11-12, 1987 in Altrincham, Manchester, UK, London: Elsevier Applied Science, Reading, 1987, pp.232-241.

Leveson, N.G., "Building Safe Software," in Software Reliability: Achievement and Assessment, ed. by Littlewood, B., Oxford: Blackwell Scientific Publications, Reading, 1987, pp.1-18.

Goldberg, J., "Some Principles and Techniques for Designing Safe Systems," ACM SIGSOFT Software Engineering Notes, Vol.12, No.3, pp.17-19, (July 1987).

Jackson, T., "Practical RAMCAD," 1987 Proc. Tech. Meet. (Institute of Environmental Sciences, USA), Vol.33, pp.134-140.

DTI and NCC, The STARTS Guide - A Guide to Methods and Software Tools for the Construction of Large Real-Time Systems, Manchester: NCC Publications, 2nd ed(2 vols), 1987, pp.496.

[1988]

Brown, M.L., "Software Systems Safety and Human Errors," COMPASS '88, June/July 1988, pp.19-28.

Cha, S.S., Leveson, N.G., and Shimeall, T.J., "Safety Verification in MURPHY Using Fault Tree Analysis," ICSE-10, Singapore, Apr. 1988, pp.377-386.

Ntafos, S.C., "A Comparison of Some Structural Testing Strategies," IEEE Transactions on Software Engineering, Vol.SE-14, No.6, pp.868-874, (June 1988).

Dahll, G., Mainka, U., and Martz, J., "Tools for the Standardised Software Safety Assessment(The SOSAT Project)," IFAC SAFECOMP '88, Nov. 1988, pp.1-6.

Shore, J., "Why I Never Met a Programmer I Could Trust," Communications of the ACM, Vol.31, No.4, pp.372-375, (Apr. 1988).

Fetzer, J.H., "Program Verification: The Very Idea," Communications of the ACM, Vol.31, No.9, pp.1048-1063, (Sept. 1988).

Rushby, J., "Validation and Testing of Knowledge-Based Systems: How Bat Can It Get?," The AAAI-88 Workshop on Verification and Validation of Expert Systems, Dec. 1988, pp.1-9.

Hansen, M.D. and Watts, R.L., "Software System Safety and Reliability," Proceedings of the Annual Reliability and Maintainability Symposium 1988, Los Angeles, CA, Jan. 1988, pp.214-217.

IAEA, Manual on Quality Assurance for Computer Software Related to the Safety of Nuclear Power Plants, International Atomic Energy Agency, Technical Reports Series No.282, Vienna, 1988, pp.104.

当麻(監修), 向殿(編集), コンピュータシステムの高信頼化技術入門：フォールトトレランスシステムの基礎, 日本規格協会, 1988年3月, pp.249.

DOD-STD-2167A: Military Standard Defense System Software Development, Department of Defense, Washington, D.C., 29 Feb. 1988.

MIL-STD-480B: Military Standard Configuration Control - Engineering Changes, Deviations and Waivers, Department of Defense, Washington, D.C., 15 July 1988.

Hill, J.V., "The Development of High Reliability Software - RR&A's Experience for Safety Critical Systems," BCS/IEE Conference on Software Engineering, Liverpool, 1988, pp.169-172.

[1989]

Duke, E.L., "V&V of Flight and Mission-Critical Software," IEEE Software, Vol.6, No.3, pp.39-45, (May 1989).

Leveson, N.G., "Safety as a Software Quality," IEEE Software, Vol.6, No.3, pp.88-89, (May 1989).

Gruman, G., "Software Safety Focus of New British Standard," IEEE Software, Vol.6, No.3, pp.95-96, (May 1989).

McKinlay, A.VI, "Software Safety Handbook," COMPASS '89, June 1989, pp.14-17.

Gerhart, S., "Preliminary Summary: FM89 Assessment of Formal Methods for Trustworthy Computer Systems," Proceedings of the ACM SIGSOFT '89: The Third Symposium on Software Testing, Analysis, and Verification(TAV3), Key West, Florida, Dec. 13-15, 1989, pp.152-156. Also available in ACM SIGSOFT Software Engineering Notes, Vol.14, No.8, pp.152-155, (Dec. 1989).

Brilliant, S., Knight, J.C., and Leveson, N.G., "The Consistent Comparison Problem in N-Version Software," IEEE Transactions on Software Engineering, Vol.SE-15, No.11, pp.1481-1484, (Nov. 1989).

Schoitsch, E., "The Interaction between Practical Experience, Standardization

and the Application of Standards," IFAC SAFECOMP '89, Dec. 1989, pp.17-24.

Erb, A., "Safety Measures of the Electronic Interlocking System 'ELEKTRA'," IFAC SAFECOMP '89, Dec. 1989, pp.49-52.

Alzerra, D. and Galivel, G., "Validating the SACEM Railway Control System Using the IDAS Software Test and Debugging Tool," IFAC SAFECOMP '89, Dec. 1989, pp.59-63.

Grimm, K., "An Effective Strategy and Automation Concepts for Systematic Testing of Safety Related Software," IFAC SAFECOMP '89, Dec. 1989, pp.71-79.

Wirthumer, G., "VOTRICS - Fault Tolerance Realized in Software," IFAC SAFECOMP '89, Dec. 1989, pp.135-140.

Jaffe, M.S. and Leveson, N.G., "Completeness, Robustness, and Safety in Real-Time Software Requirements Specification," ICSE-11, Pittsburgh, Pennsylvania, May 1989, pp.302-311.

IEC Draft Standard: Software for Computers in the Application of Industrial Safety-Related Systems, IEC Reference:65A(Secretariat)94, Aug. 1989, pp.165.

IEC Draft Standard: Functional Safety of Programmable Electronic Systems: Generic Aspects; Part 1: General Requirements, IEC Reference:65A(Secretariat) 96, Sept. 1989, pp.104.

Chudleigh, M., "Developing Software for Safety-Critical Applications," The Nuclear Engineer, Vol.30, No.1, pp.14-21, (1989).

Dale, C., "Software Safety Certification in Potentially Hazardous Industries," in Software Certification, ed. by de Neumann, B., London: Elsevier Applied Science, Reading, 1989, pp.100-112.

Miller, D.R., "The Role of Statistical Modeling and Inference in Software Quality Assurance," in Software Certification, ed. by de Neumann, B., London: Elsevier Applied Science, Reading, 1989, pp.135-152.

Bloomfield, R.E. and Froome, P.K.D., "Aspects of the Licensing and Assessment of Highly Dependable Computer Systems," in Safe and Secure Computing Systems, ed. by Anderson, T., Oxford: Blackwell Scientific Publications, Reading, 1989, pp.1-39.

Leveson, N.G., "Safety-Critical Software Development," in Safe and Secure Computing Systems, ed. by Anderson, T., Oxford: Blackwell Scientific Publications, Reading, 1989, pp.155-162.

Laprie, J.-C., "Hardware-and-Software Dependability Evaluation," Proceedings of the IFIP 11th World Computer Congress, San Francisco, Calif., Aug./Sept. 1989, pp.109-114.

Knight, J.C. and Ammann, P.E., "Issues Influencing the Use of N-Version Programming," Proceedings of the IFIP 11th World Computer Congress, San Francisco, Calif., Aug./Sept. 1989, pp.217-222.

Avizienis, A., "Software Fault Tolerance," Proceedings of the IFIP 11th World Computer Congress, San Francisco, Calif., Aug./Sept. 1989, pp.491-498.

Neumann, P.G., "Safety of Computer Systems in Aerospace Applications," Proceedings of AIAA Computers in Aerospace Conference Ⅶ, Oct. 1989, pp.970-974.

DTI and NCC, The STARTS Purchasers' Handbook - Procuring Software-Based Systems, Manchester: NCC Publications, 2nd ed, May 1989.

Pearson, J., "Proposed HSE Guidelines on Emergency Shutdown Systems," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.55-58.

Johnston, I.H.A. and Wood, A.P., "Integrity Levels and Assessment Levels in Practice," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.70-74.

Pilsworth, R., "Defence Standards 00-55 & 00-56," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.103-105.

Bennett, P.A., "The March Towards Standards in Safety Related Systems," Computers and Safety: A First International Conference on the Use of Programmable Electronic Systems in Safety Related Applications, IEE, Conf. Publ. No.314, Nov. 1989, pp.106-110.

UK Interim Defence Standard 00-56/1: Requirements for the Analysis of Safety Critical Hazards, Ministry of Defence, Directorate of Standardisation, Kentigern House, 65 Brown St., Glasgow G2 8EX, May 1989.

Bennett, P.A., "Experiences Assessing High Integrity Software," IEE Colloquium on 'Control Systems Software Reliability for Industrial Applications', London, Oct. 1989, pp.2/1-2/3.

[1990]

Dunham, J.R. and Finelli, G.B., "Real-Time Software Failure Characterization," COMPASS '90, June 1990, pp.39-45.

Brown, M.J.D., "Rationale for the Development of the UK Defence Standards for Safety-Critical Computer Software," COMPASS '90, June 1990, pp.144-150.

Bell, R. and Smith, S., "An Overview of IEC Draft Standards: Functional Safety

of Programmable Electronic Systems," COMPASS '90, June 1990, pp.151-163.

Buckley, T.F., Jesty, P.H., Hobley, K., and West, M., "DRIVE-ing Standards:- A Safety Critical Matter," COMPASS '90, June 1990, pp.164-172.

Singh, A.D. and Murugesan, S., "Fault-Tolerant Systems," IEEE Computer, Vol.23, No.7, pp.15-17, (July 1990).

Nelson, V.P., "Fault-Tolerant Computing: Fundamental Concepts," IEEE Computer, Vol.23, No.7, pp.19-25, (July 1990).

Laprie, J.-C., Arlat, J., Beounes, C., and Kanoun, K., "Definition and Analysis of Hardware- and Software-Fault-Tolerant Architectures," IEEE Computer, Vol.23, No.7, pp.39-51, (July 1990).

Parnas, D.L., van Schouwen, A.J., and Kwan, S.P., "Evaluation of Safety-Critical Software," Communications of the ACM, Vol.33, No.6, pp.636-648, (June 1990).

Parnas, D.L., "Education for Computing Professionals," IEEE Computer, Vol.23, No.1, pp.17-22, (Jan. 1990).

Gerhart, S.L., "Applications of Formal Methods: Developing Virtuos Software," IEEE Software, Vol.7, No.5, pp.7-10, (Sept. 1990).

Hall, A., "Seven Myths of Formal Methods," IEEE Software, Vol.7, No.5, pp.11-19, (Sept. 1990).

Spivey, J.M., "Specifying a Real-Time Kernel," IEEE Software, Vol.7, No.5, pp.21-28, (Sept. 1990).

Cobb, R.H. and Mills, H.D., "Engineering Software under Statistical Quality Control," IEEE Software, Vol.7, No.6, pp.44-54, (Nov. 1990).

Leveson, N.G., "The Challenge of Building Process-Control Software," IEEE Software, Vol.7, No.6, pp.55-62, (Nov. 1990).

Saeed, A., Anderson, T., and Koutny, M., "A Formal Model for Safety-Critical Computing Systems," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.1-6.

Ravn, A.P., Rischel, H., and Stavridou, V., "Provably Correct Safety Critical Software," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.13-18.

Schoitsch, E., Dittrich, E., Grasegger, S., Kropfitsch, D., Erb, A., Fritz, P., and Kopp, H., "The ELEKTRA Testbed: Architecture of a Real-Time Test Environment for High Safety and Reliability Requirements," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.59-65.

Finnie, B.W. and Johnston, I.H.A., "Practical Experience in the Assessment of Existing Safety Critical Computer Based Systems," IFAC SAFECOMP '90,

Oct./Nov. 1990, pp.95-98.

Johnston, I.H.A., "The Impact of Social Factors on Acceptable Levels of Safety Integrity," IFAC SAFECOMP '90, Oct./Nov. 1990, pp.157-161.

Kersken, M, "Interaction of Man and Tools During Verification and Validation of Safety Critical Software," Proceedings of the Topical Meeting on Advances in Human Factors Research on Man/Computer Interactions: Nuclear and Beyond, June 1990, pp.47-52.

Guiho, G. and Hennebert, C., "SACEM Software Validation," ICSE-12, Nice, Mar. 1990, pp.186-191.

Laprie, J.-C., "On the Assessment of Safety-Critical Software Systems," ICSE-12, Nice, Mar. 1990, p.222.

Leveson, N.G., "Evaluation of Software Safety," ICSE-12, Nice, Mar. 1990, pp.223-224.

Pilaud, E., "Some Experiences of Critical Software Development," ICSE-12, Nice, Mar. 1990, pp.225-226.

Thomas, M., "Assessing Failure Probabilities in Safety-Critical Systems Containing Software," ICSE-12, Nice, Mar. 1990, p.227.

Craigen, D., "FM89: Assessment of Formal Methods for Trustworthy Computer Systems," ICSE-12, Nice, Mar. 1990, pp.233-235.

Bjorner, D. and Druffel, L., "Position Statement: The ICSE-12 Workshop on Industrial Experience Using Formal Methods," ICSE-12, Nice, Mar. 1990, pp.264-266.

Wing, J.M., "A Specifier's Introduction to Formal Methods," IEEE Computer, Vol.23, No.9, pp.8-24, (Sept. 1990).

Bjorner, D., Hoare, C.A.R., and Langmaack, H.(eds), VDM '90: VDM and Z - Formal Methods in Software Development; Proceedings of the Third International Symposium of VDM Europe held April 17-21, 1990 in Kiel, FRG, Berlin: Springer-Verlag, Reading, 1990, pp.579.

Brilliant, S., Knight, J.C., and Leveson, N.G., "Analysis of Faults in a N-Version Programming Experiment," IEEE Transactions on Software Engineering, Vol.SE-16, No.2, pp.238-247, (Feb. 1990).

Leveson, N.G., Cha, S.S., Knight, J.C., and Shimeall, T.J., "The Use of Self Checks and Voting in Software Error Detections: An Empirical Study," IEEE Transactions on Software Engineering, Vol.SE-16, No.4, pp.432-443, (Apr. 1990).

Craigen, D. and Summerskill, K.(eds), Formal Methods for Trustworthy Computer

Systems(FM89); Report from FM89: A Workshop on the Assessment of Formal Methods for Trustworthy Computer Systems held July 23-27, 1989 in Halifax, Canada, London: Springer-Verlag, Reading, 1990, pp.248.

Sayet, C. and Pilaud, E., "An Experience of a Critical Software Development," Digest of Papers Fault-Tolerant Computing: The Twentieth International Symposium(FTCS-20), Newcastle upon Tyne, England, June 26-28, 1990, pp.36-45.

Norman, D.A., "Commentary: Human Error and the Design of Computer Systems," Communications of the ACM, Vol.33, No.1, pp.4-5,7, (Jan. 1990).

Denning, P.J., "Human Error and the Search for Blame," Communications of the ACM, Vol.33, No.1, pp.6-7, (Jan. 1990).

Jacky, J., "Inside Risk: Risks in Medical Electronics," Communications of the ACM, Vol.33, No.12, p.138, (Dec. 1990).

Miller, L.A., "Dynamic Testing of Knowledge Bases Using the Heuristic Testing Approach," Expert Systems With Applications, Vol.1, No.3, pp.249-269, (1990).

Lee, P.A. and Anderson, T., Fault Tolerance - Principles and Practice, Second, revised edition, Wien: Springer-Verlag, Dependable Computing and Fault-Tolerant Systems, Vol.3, Reading, 1990, pp.320.

Abboytt, R.J., "Resourceful Systems for Fault Tolerance, Reliability, and Safety," ACM Computing Surveys, Vol.22, No.1, pp.35-68, (Mar. 1990).(浦野訳, 「フォールトトレランス, 信頼性, そして安全性のための"機略"システム」, bit 別冊 ACM COMPUTING SURVEYS '90 コンピュータ・サイエンス, 共立出版, 1992年7月号別冊, pp.5-36.)

浦野, 中村, "フォールトトレラントソフトウェア技術," 電子情報通信学会, Vol.73, No.5, pp.475-480, (May 1990).

Chudleigh, M., "Software and Safety: How Compatible are They?," Information and Software Technology, Vol.32, No.5, pp.323-329, (June 1990).

SEB6-A(EIA): System Safety Engineering in Software Development, Electronic Industries Association, EIA Safety Engineering Bulletin No.6-A, Apr. 1990.

Reed, W., "Safety Critical Software in Traffic Control Systems," IEE Colloquium on 'Safety Critical Software in Vehicle and Traffic Control', London, Feb. 1990, pp.2/1-2/5.

[1991]

Shimeall, T.J., McGraw, R.J., Jr., Gill, J.A., "Software Safety Analysis in Heterogeneous Multiprocessor Control Systems," Proceedings of the Annual Reliability and Maintainability Symposium 1991, Orlando, Florida, Jan. 1991, pp.290-294.

Strandberg, K., "IEC 300: The Dependability Counterpart of ISO 9000," Proceedings of the Annual Reliability and Maintainability Symposium 1991, Orlando, Florida, Jan. 1991, pp.463-467.

Leveson, N.G., "Software Safety in Embedded Computer Systems," Communications of the ACM, Vol.34, No.2, pp.34-46, (Feb. 1991).

Neumann, P.G., "Inside Risk: Certifying Professionals," Communications of the ACM, Vol.34, No.2, p.130, (Feb. 1991).

Horstmann, C., "Arguing Against Certification," Communications of the ACM, Vol.34, No.10, p.13, (Oct. 1991).

Neumann, P.G.(ed), "Risks to the Public in Computers and Related Systems: Subsection on Certification of Professionals," ACM SIGSOFT Software Engineering Notes, Vol.16, No.1, pp.24-32, (Jan. 1991).

Shimeall, T.J. and Leveson, N.G., "An Empirical Comparison of Software Fault Tolerance and Fault Elimination," IEEE Transactions on Software Engineering, Vol.17, No.2, pp.173-182, (Feb. 1991).

Jaffe, M.S., Leveson, N.G., Heimdahl, M.P.E., and Melhart, B.E., "Software Requirements Analysis for Real-Time Process-Control Systems," IEEE Transactions on Software Engineering, Vol.17, No.3, pp.241-258, (Mar. 1991).

Baker, H.G., "Factoring Redundancy," Communications of the ACM, Vol.34, No.5, pp.98-99, (May 1991).

Avizienis, A. and Laprie, J.-C.(eds), Dependable Computing for Critical Applications, Wien: Springer-Verlag, Dependable Computing and Fault-Tolerant Systems, Vol.4, Reading, 1991, pp.429.

Barnes, M., "Dependable Computing in the UK," in Dependable Computing for Critical Applications, ed. by Avizienis, A. and Laprie, J.-C., Wien: Springer-Verlag, Reading, 1991, pp.3-21.

Thevenod-Fosse, P., "Software Validation by Means of Statistical Testing: Retrospect and Future Direction," in Dependable Computing for Critical Applications, ed. by Avizienis, A. and Laprie, J.-C., Wien: Springer-Verlag, Reading, 1991, pp.23-50.

Reibman, A.L. and Veeraraghavan, M., "Reliability Modeling: An Overview for System Designers," IEEE Computer, Vol.24, No.4, pp.49-57, (Apr. 1991).

Belli, F. and Jedrzejowicz, P., "An Approach to the Reliability Optimization of Software with Redundancy," IEEE Transactions on Software Engineering, Vol.17, No.3, pp.310-312, (Mar. 1991).

Rettig, M., "Testing Made Palatable," Communications of the ACM, Vol.34, No.5, pp.25-29, (May 1991).

Lafontaine, C., Ledru, Y., and Schobbens, P.-Y., "An Experiment in Formal Software Development Using the B Theorem Prover on a VDM Case Study," Communications of the ACM, Vol.34, No.5, pp.62-87, (May 1991).

Gabrielian, A. and Franklin, M.K., "Multilevel Specification of Real-Time Systems," Communications of the ACM, Vol.34, No.5, pp.50-60, (May 1991).

Gerhart, S., "Formal Methods: An International Perspective," ICSE-13, Austin, May 1991, pp.36-37.

Craigen, D., "Tool Support for Formal Methods," ICSE-13, Austin, May 1991, pp.184-185.

Apostolakis, G.(ed), Probabilistic Safety Assessment and Management: Proceedings of the International Conference on Probabilistic Safety Assessment and Management(PSAM) held February 4-7, 1991 in Bevery Hills, California, New York: Elsevier, Reading, 1991, pp.1545.

Bowman, W.C., Archinoff, G.H., Raina, V.M., Tremaine, D.R., and Leveson, N.G., "An Application of Fault Tree Analysis to Safety Critical Software at Ontario Hydro," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.363-368.

McKinlay, A.VI., "The State of Software Safety Engineering," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.369-376.

Leveson, N.G., "Safety Assessment and Management Applied to Software," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.377-382.

Guarro, S.B., Wu, J.S., Apostolakis, G.E., and Yau, M., "Embedded System Reliability and Safety Analysis in the UCLA ESSAE Project," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.383-388.

Petrella, S., Michael, P., Bowman, W.C., and Lim, S.T., "Random Testing of Reactor Shutdown System Software," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.681-686.

Daughtrey, H.T., Levinson, S.H., and Kimmel, D.M., "Developing a Standard for the Qualification of Software for Safety System Applications," in Probabilistic Safety Assessment and Management, ed. by Apostolakis, G., New York: Elsevier, Reading, 1991, pp.687-692.

Leveson, N.G., Cha, S.S., and Shimeall, T.J., "Safety Verification of Ada Programs Using Software Fault Trees," IEEE Software, Vol.8, No.4, pp.48-59, (July 1991).

Kelly, J.P.J., McVittie, T.I., and Yamamoto, W.I., "Implementing Design Diversity to Achieve Fault Tolerance," IEEE Software, Vol.8, No.4, pp.61-71, (July 1991).

Fraser, M.D., Kumar, K., and Vaishnavi, V.K., "Informal and Formal Requirements Specification Languages: Bridging the Gap," IEEE Transactions on Software Engineering, Vol.17, No.5, pp.454-466, (May 1991).

Cooling, J.E., Software Design for Real-Time Systems, London: Chapman and Hall, Reading, 1991, pp.506.

Warwick, K. and Tham, M.T.(eds), Failsafe Control Systems: Applications and Emergency Management, London: Chapman and Hall, Reading, 1991, pp.246.

Bell, R. and Pantony, M.F., "Framework for the Design and Assessment of Safety Related Control Systems," in Failsafe Control Systems: Applications and Emergency Management, ed. by Warwick, K. and Tham, M.T., London: Chapman and Hall, Reading, 1991, pp.70-92.

McDermid, J.A., "Issues in Developing Software for Safety Critical Systems," Reliability Engineering and System Safety, Vol.32, Nos.1/2, pp.1-24, (1991). Also available in Software Reliability and Safety, ed. by Littlewood, B. and Miller, D., London: Elsevier Applied Science, Reading, 1991, pp.1-24.

Bloomfield, R.E., Froome, P.K.D., and Monahan, B.Q., "Formal Methods in the Production and Assessment of Safety Critical Software," Reliability Engineering and System Safety, Vol.32, Nos.1/2, pp.51-66, (1991). Also available in Software Reliability and Safety, ed. by Littlewood, B. and Miller, D., London: Elsevier Applied Science, Reading, 1991, pp.51-66.

Gonzalez Sanz, G., "Standardization for Safety Software: Current Status and Perspectives," The Fourth Software Engineering Standards Application Workshop(SESAW '91), San Diego, May 1991, pp.84-105.

Wright, C.L. and Zawilski, A.J., "Existing and Emerging Standards for Software Safety," The Fourth Software Engineering Standards Application Workshop(SESAW '91), San Diego, May 1991, pp.112-118.

Tripp, L.L., Struck, W.F., and Pflug, B.K., "The Application of Multiple Team Inspections on a Safety-Critical Software Standard," The Fourth Software Engineering Standards Application Workshop(SESAW '91), San Diego, May 1991, pp.106-111.

McDermid, J.A., "Safety Arguments, Software and System Reliability," Proceedings of the 1991 International Symposium on Software Reliability Engineering(ISSRE '91), Austin, Texas, May 17-18, 1991, pp.43-50.

Potocki De Montalk, J.P., "Computer Software in Civil Aircraft," COMPASS '91, June 1991, pp.10-16.

Buckley, T.F., Jesty, P.H., and West, M.M., "Some Results from DRIVE," COMPASS '91, June 1991, pp.17-26.

Taylor, J.A., "Training, Career Development and Registration for Safety Critical Software Systems Specialists," COMPASS '91, June 1991, pp.29-35.

Gove, R.A. and Heinzman, J.L., "Safety Criteria and Model for Mission-Critical Embedded Software Systems," COMPASS '91, June 1991, pp.69-73.

Saeed, A., de Lemos, R., and Anderson, T., "The Role of Formal Methods in the Requirements Analysis of Safety-Critical Systems: A Train Set Example," Digest of Papers Fault-Tolerant Computing: The Twenty-First International Symposium(FTCS-21), Montreal, Canada, June 25-27, 1991, pp.478-485.

Neumann, P.G., "Inside Risk: The Human Element," Communications of the ACM, Vol.34, No.11, p.150, (Nov. 1991).

Shagnea, A.M., Hayhurst, K.J., and Withers, B.E., "Data Collection and Descriptive Analysis: A First Step for Developing Quality Software," COMPASS '91, June 1991, pp.173-179.

McFarland, M.C., "Ethics and the Safety of Computer Systems," IEEE Computer, Vol.24, No.2, pp.72-75, (Feb. 1991).

Kriewall, T.J. and Long, J.M., "Computer-Based Medical Systems," IEEE Computer, Vol.24, No.3, pp.9-12, (Mar. 1991).

Klein, P., "The Safety-Bag Expert System in the Electronic Railway Interlocking System Elektra," Expert Systems With Applications, Vol.3, No.4, pp.499-506, (1991).

Neumann, P.G.(ed), "Risks to the Public in Computers and Related Systems: Subsection on Safety Standards," ACM SIGSOFT Software Engineering Notes, Vol.16, No.3, p.28, (July 1991).

Stavridou,V. and Ravn,A., "A Comment on the Revised UK MOD Standards 00-55/56," ACM SIGSOFT Software Engineering Notes, Vol.16, No.4, p.22, (Oct. 1991).

Smith, I.C., Wall, D.N., and Baldwin, J.A., "DARTS - An Experiment into Cost of and Diversity in Safety Critical Computer Systems," IFAC SAFECOMP '91, Oct./Nov. 1991, pp.35-40.

Dobson, J., Laprie, J.-C., and Randell, B., "Predictably Dependable Computing Systems: An ESPRIT Basic Research Action," in Software Reliability and Metrics, ed. by Fento, N. and Littlewood, B., London: Elsevier Applied Science, Reading, 1991, pp.111-131.

Ehrenberger, W., Roan, A., and Robert, P., "Software Certification Programme in Europe - SCOPE," in Software Reliability and Metrics, ed. by Fento, N. and

Littlewood, B., London: Elsevier Applied Science, Reading, 1991, pp.132-148.

Wingrove, A.A., "Software Certification," in Software Reliability and Metrics, ed. by Fento, N. and Littlewood, B., London: Elsevier Applied Science, Reading, 1991, pp.192-216.

Pyle, I.C., Developing Safety Systems: A Guide Using Ada, London: Prentice Hall International, Reading, 1991, pp.254.

ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety Related Systems, IEC Reference:65A(Secretariat)122, Nov. 1991, pp.88.

UK Interim Defence Standard 00-50: The Procurement of Safety Critical Software in Defence Equipment: Part 1: Requirements, Part 2: Guidance, Ministry of Defence, Directorate of Standardization, Kentigern House, 65 Brown Street, Glasgow G2 8EX, 5 April 1991.

Kersken, M. and Saglietti, F.(eds), Software Fault Tolerance: Achievement and Assessment Strategies, Research Reports: ESPRIT Project 300 REQUEST (Reliability and Quality of European Software Technology), Vol.1, Berlin: Springer-Verlag, Reading, 1991, pp.243.

Hill, J.V., "Software Development Methods in Practice," in Microprocessor Based Protection Systems, ed. by Churchley, A., London: Elsevier Applied Science, Reading, 1991, pp.101-118.

Ward, N.J., "The Static Analysis of Critical Software," in Microprocessor Based Protection Systems, ed. by Churchley, A., London: Elsevier Applied Science, Reading, 1991, pp.121-130.

Froome, P.K.D. and Monahan, B.Q., "Formal Methods for the Development and Assessment of Microprocessor Based Protection Systems," in Microprocessor Based Protection Systems, ed. by Churchley, A., London: Elsevier Applied Science, Reading, 1991, pp.133-152.

Gray, E.M. and Thayer, R.H., "Requirements," in Aerospace Software Engineering: A Collection of Concepts, ed. by Anderson, C. and Dorfman, M., Washington, DC: AIAA, Reading, 1991, pp.89-122.

Leveson, N.G., "Safety," in Aerospace Software Engineering: A Collection of Concepts, ed. by Anderson, C. and Dorfman, M., Washington, DC: AIAA, Reading, 1991, pp.319-337.

Jennings, N., "Aerospace Software Engineering in the United Kingdom," in Aerospace Software Engineering: A Collection of Concepts, ed. by Anderson, C. and Dorfman, M., Washington, DC: AIAA, Reading, 1991, pp.545-559.

Jesty, P.H., Lamb, D.A., Buckley, T.F., and West, M.M., "Choice of Design Methodologies for Demanding High Integrity Industrial Control Systems," The Third IEE/BCS International Conference on Software Engineering for Real Time

Systems, Cirencester, Sept. 1991, pp.16-21.

Hughes, T.S. and Cooling, J.E., "Real-Time Systems - Animation Prototyping of Formal Specifications," The Third IEE/BCS International Conference on Software Engineering for Real Time Systems, Cirencester, Sept. 1991, pp.51-56.

Cooling, J.E. and Hughes, T.S., "Animation Prototyping of Real-Time Systems Specifications," The 5th Annual European Computer Conference: Advanced Computer Technology, Reliable Systems and Applications(CompEuro '91), 1991, pp.562-566.

Probert, P.J. and Lamb, D., "Validation Methods Using Formal Techniques," The Third IEE/BCS International Conference on Software Engineering for Real Time Systems, Cirencester, Sept. 1991, pp.244-249.

Potter, B., Sinclair, J., and Till, D., An Introduction to Formal Specification and Z, London: Prentice Hall International, Reading, 1991, pp.304.

Dandanell, B., "Rigorous Development Using RAISE," ACM SIGSOFT Software Engineering Notes, Vol.16, No.5, pp.29-43, (Dec. 1991). Also available in Proceedings of the ACM SIGSOFT '91 Conference on Software for Critical Systems, New Orleans, Louisiana, December 4-6, 1991.

Butler, R.W. and Finelli, G.B., "The Infeasibility of Experimental Quantification of Life-Critical Software Reliability," ACM SIGSOFT Software Engineering Notes, Vol.16, No.5, pp.66-76, (Dec. 1991). Also available in Proceedings of the ACM SIGSOFT '91 Conference on Software for Critical Systems, New Orleans, Louisiana, December 4-6, 1991.

Bennett, P.A., "Forwards to Safety Standards," Software Engineering Journal, Vol.6, No.2, pp.37-40, (Mar. 1991).

[1992]

Toola, A., "Elicitation of Safety Requirements for Process Automation," Reliability Engineering and System Safety, Vol.35, No.3, pp.209-215, (1992), (Published in 1991).

Laprie, J.-C. and Littlewood, B., "Probabilistic Assessment of Safety- Critical Software: Why and How," Communications of the ACM, Vol.35, No.2, pp.13,16,21, (Feb. 1992).

Eagle, K.H. and Agarwala, A.S., "Redundancy Design Philosophy for Catastrophic Loss Protection," Proceedings of the Annual Reliability and Maintainability Symposium 1992, Las Vegas, Nevada, Jan. 1992, pp.1-4.

Sethy, A., "Connection Between Reliability and Signalling-Safety in Railway Technology," Proceedings of the Annual Reliability and Maintainability Symposium 1992, Las Vegas, Nevada, Jan. 1992, pp.75-79.

Keene, S.J., Jr., "Assuring Software Safety," Proceedings of the Annual Reliability and Maintainability Symposium 1992, Las Vegas, Nevada, Jan. 1992, pp.274-279.

Tripp, L.L., "What is the Future of Software Engineering Standards?," ACM SIGSOFT Software Engineering Notes, Vol.17, No.1, pp.58-61, (Jan. 1992).

Laprie, J.-C.(ed), Dependability: Basic Concepts and Terminology in English, French, German, Italian and Japanese, Wien: Springer-Verlag, Dependable Computing and Fault-Tolerant Systems, Vol.5, Reading, 1992, pp.265.

Meyer, J.F. and Schlichting, R.D.(eds), Dependable Computing for Critical Applications 2, Wien: Springer-Verlag, Dependable Computing and Fault-Tolerant Systems, Vol.6, Reading, 1992, pp.437.

Lyu, M.R., "Assuring Design Diversity in N-Version Software: A Design Paradigm for N-Version Programming," in Dependable Computing for Critical Applications 2, ed. by Meyer, J.F. and Schlichting, R.D., Wien: Springer-Verlag, Reading, 1992, pp.197-218.

Thomas, M., "Limits and Customers are Driving Change," IEEE Software, Vol.9, No.2, p.10, (Mar. 1992).

Melford, R.J., "Development to Begin on Standards of Ethical Practices for Computing Professionals," IEEE Computer, Vol.25, No.4, pp.76-78, (Apr. 1992).

Gantenbein, R.E., "An Annotated Bibliography of Dependable Distributed Computing," ACM SIGOPS Operating Systems Review, Vol.26, No.2, pp.60-81, (Apr. 1992).

Fields, B. and Elvang-Goransson, M., "A VDM Case Study in MURAL," IEEE Transactions on Software Engineering, Vol.18, No.4, pp.279-295, (Apr. 1992).

ACM, "ACM Code of Ethics and Professional Conduct," Communications of the ACM, Vol.35, No.5, pp.94-99, (May 1992).

Wiener, L., "A Trip Report on SIGSOFT '91," ACM SIGSOFT Software Engineering Notes, Vol.17, No.2, pp.23-38, (Apr. 1992).

Chokhani, S., "Trusted Products Evaluation," Communications of the ACM, Vol.35, No.7, pp.64-76, (July 1992).

Sadeghi, T. and Motamed, F., "Evaluation and Comparison of Triple and Quadruple Flight Control Architectures," IEEE AES Systems Magazine, Vol.7, No.3, pp.20-31, (Mar. 1992).

Wichmann, B.A.(ed), Software in Safety-Related Systems, Chichester: John Wiley & Sons, Reading, 1992, pp.304.

Wallace, D.R., Kuhn, D.R., and Ippolito, L.M., "An Analysis of Selected Software Safety Standards," IEEE AES Systems Magazine, Vol.7, No.8, pp.3-14, (Aug. 1992). Also available in COMPASS '92, June 1992.

Neumann, P.G., "Inside Risk: Where to Place Trust," Communications of the ACM, Vol.35, No.10, p.138, (Oct. 1992).

Deckers, J. and Schabe, H., "The Sneak Circuit Analysis - An Investigation Method for Aerospace Systems," Proceedings of the European Safety and Reliability Conference '92(ESRC '92), Copenhagen, Denmark, June 10-12, 1992, pp.607-618.

Charon, P., Kahn, P., Le Henaff, A., Mignot, J., and Nicolas, O., "Sneak Analysis Evaluation: General Statement on 'Sneak' Approach and Practical Case Studies," Proceedings of the European Safety and Reliability Conference '92(ESRC '92), Copenhagen, Denmark, June 10-12, 1992, pp.1147-1155.

Halbwachs, N., Lagnier, F., and Ratel, C., "Programming and Verifying Real-Time Systems by Means of the Synchronous Data-Flow Language LUSTRE," IEEE Transactions on Software Engineering, Vol.18, No.9, pp.785-793, (Sept. 1992).

Mahony, B.P. and Hayes, I.J., "A Case-Study in Timed Refinement: A Mine Pump," IEEE Transactions on Software Engineering, Vol.18, No.9, pp.817-826, (Sept. 1992).

Schneidewind, N.F.(ed), "Are Software Engineering Process Standards Really Necessary?," IEEE Computer, Vol.25, No.11, pp.82-84, (Nov. 1992).

Peters, J.F., III and Ramanna, S., "Verifying Command Sequences for Satellite Systems," IEEE AES Systems Magazine, Vol.7, No.10, pp.14-21, (Oct. 1992).

Sparkman, D., "Standards and Practices for Reliable Safety-Related Software Systems," Proceedings of the Third International Symposium on Software Reliability Engineering(ISSRE '92), Research Triangle Park, North Carolina, Oct. 7-10, 1992, pp.318-327.

Chelini, J.V., "A Discussion on the Ada Run-Time Environment in Safety Critical Applications," ACM SIGSOFT Software Engineering Notes, Vol.17, No.4, pp.24-27, (Oct. 1992).

Friend, J.R., "A Review of Computer Controlled Systems Safety and Quality Assurance Concerns for Acquisition Managers," COMPASS '92, June 1992, pp.105-122.

Wallace, D.R., Kuhn, D.R., and Ippolito, L.M., "An Analysis of Selected Software Safety Standards," COMPASS '92, June 1992, pp.123-136. Also available in IEEE AES Systems Magazine, Vol.7, No.8, pp.3-14, (Aug. 1992).

Davis, P.I., "Certification of Safety-Critical Software by Licensed Software Engineers," IEEE Computer, Vol.25, No.12, pp.72-73, (Dec. 1992).

Bowen, J.P. and Stavridou, V., "Formal Methods and Software Safety," IFAC SAFECOMP '92, Oct. 1992, pp.93-98.

Goddard, E.O. and Zufferey, C.H., Report by the Technical Committee: Cross-Acceptance of Vital Signalling Systems, The Institution of Railway Signal Engineers, Mar. 1992, pp.10.

Leveson, N.G., "High-Pressure Steam Engines and Computer Software," ICSE-14, Melbourne, Australia, May 1992, pp.2-14.

[1993]

Neumann, P.G., "Inside Risk: Are Dependable Systems Feasible?," Communications of the ACM, Vol.36, No.2, p.146, (Feb. 1993).

Bhansali, P.V., "Survey of Software Safety Standards Shows Diversity," IEEE Computer, Vol.26, No.1, pp.88-89, (Jan. 1993).

■

付録　ソフトウェア安全性調査研究ワーキンググループの議事内容一覧

付録　ソフトウェア安全性調査研究ワーキンググループの議事内容一覧

平成４年度はソフトウェアの安全性の調査研究を進めるに際して、以下に示す議事内容を「ソフトウェア安全性調査研究ワーキンググループ」で検討した。

(1) 第１回ＷＧ

日　時：　平成４年５月２６日（火）

議事内容：

① 太刀川氏：鉄道信号におけるソフトウェアの信頼性評価について

② ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety-Related Systems の検討

鍛冶担当：第１条～第８条

(2) 第２回ＷＧ

日　時：　平成４年６月３０日（火）

議事内容：

① ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety-Related Systems の検討

・松本氏：第９条、第１０条

・中村氏：第１１条、第１２条

(3) 第３回ＷＧ

日　時：　平成４年７月２１日（火）

議事内容：

① ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety-Related Systems の検討

・太刀川氏：第１３条、第１４条

・内　海氏：第１５条、第１６条

(4) 第４回ＷＧ

日　時：　平成４年８月２５日（火）

議事内容：

① ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety-Related Systems の検討

・松尾谷氏：第１７条、第１８条
・総合討議

(5) 第５回ＷＧ

日　時：　平成４年９月３０日（水）

議事内容：

① ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety-Related Systems の表の検討
・鍛治担当：第６、７、８条の表および引用している詳細表
・中　村氏：第１１、１２条の表および引用している詳細表
・太刀川氏：第１３、１４条の表および引用している詳細表

(6) 第６回ＷＧ

日　時：　平成４年１０月２７日（火）

議事内容：

① ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety-Related Systems の表の検討
・松　本氏：第　９、１０条の表および引用している詳細表
・太刀川氏：第１３、１４条の表および引用している詳細表
・内　海氏：第１５、１６条の表および引用している詳細表
・松尾谷氏：第１７条の表および引用している詳細表

(7) 第７回ＷＧ

日　時：　平成４年１１月２５日（水）

議事内容：

① 片山禎昭氏(東芝アドバンストシステム(株))の講演
「ソフトウェアの安全性の評価技術について」
② ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety-Related Systems の表の検討
・内　海氏：第１５、１６条の表および引用している詳細表
③ 「今後の検討の進め方」および「検討のまとめ方」について（全員）

(8) 第８回ＷＧ

日　時：　平成４年１２月２５日（金）

議事内容：

① 峰尾欽二氏（日本ユニシス(株)）の講演
「形式的ソフトウェア開発法とＲＡＩＳＥ」
② 航空宇宙分野のソフトウェア規格との比較（松本氏）
③ ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety-Related Systemsの解説
・鍛治担当：第１条から第８条

(9) 第９回ＷＧ
日　時：　平成５年１月２９日（金）
議事内容：
① 松原友夫氏（コンサルタント）の講演
「ソフトウェア工学の規格化の現状」
② ソフトウェア安全規格の目的（松尾谷氏）
③ 原子力分野のソフトウェア規格との比較（内海氏）
④ 鉄道分野のソフトウェア規格との比較（太刀川氏）
⑥ 報告書目次の検討

(10)第10回ＷＧ
日　時：　平成５年２月２６日（金）
議事内容：
① 鉄道分野のソフトウェア規格との比較（中村氏）
② ISO/IEC JTC1/SC7 N917: Software for Computers in the Application of Industrial Safety-Related Systemsの解説
・中　村氏：第１１、１２条の解説
・太刀川氏：第１３、１４条の解説
・内　海氏：第１５、１６条の解説
・松尾谷氏：第１７、１８条の解説

——禁 無 断 転 載——

平成５年３月発行

発行所　財団法人　日本情報処理開発協会
東京都港区芝公園３丁目５番８号
機 械 振 興 会 館 内
電話　０３（３４３２）９３７２

印刷所　三 協 印 刷 株 式 会 社
東京都世田谷区池尻２丁目35番７号
電話　０３（３４１０）７３０

０４－Ｒ ００５