1. クリティカル・システムを取り巻く環境の変化

航空宇宙産業や軍事産業では 1960 年代から，使用するソフトウェアが起こす問題をどのように押さえ込むか，また，事故に至らないようにするにはどうしたらよいのか，について研究されてきました．医療機器業界では 1980 年代から，ソフトウェアによる放射線治療器の事故をきっかけに，米国 FDA（食品医薬品局）が安全なソフトウェアを構築するためには何をすればよいかの研究を始め，1990 年代に医療機器製造業者に対して「General Principles of Software Validation（ソフトウェア・バリデーションの一般原則）」というガイダンスを作りました(1)．自動車業界では 2011 年に，ISO 26262:2011「Road vehicles -- Functional safety --（自動車 -- 機能安全 --）」が発行されました．自動車産業は巨大な市場であるため，ISO 26262 に関連するトレーニング，コンサルテーション，第三者認証などのさまざまなサービス産業が，自動車メーカ，および自動車メーカに ECU（電子制御ユニット）を納入するサプライヤに売り込みをかけています．

この流れは，ソフトウェアに対する安全対策の要求が，ソフトウェアを搭載する対象機器に問題が発生したときの人的被害/経済的損失が極めて高い産業から，一般の人々が使用する機器へと広がりつつあることを意味しています．

●安全は絶対的な概念ではない

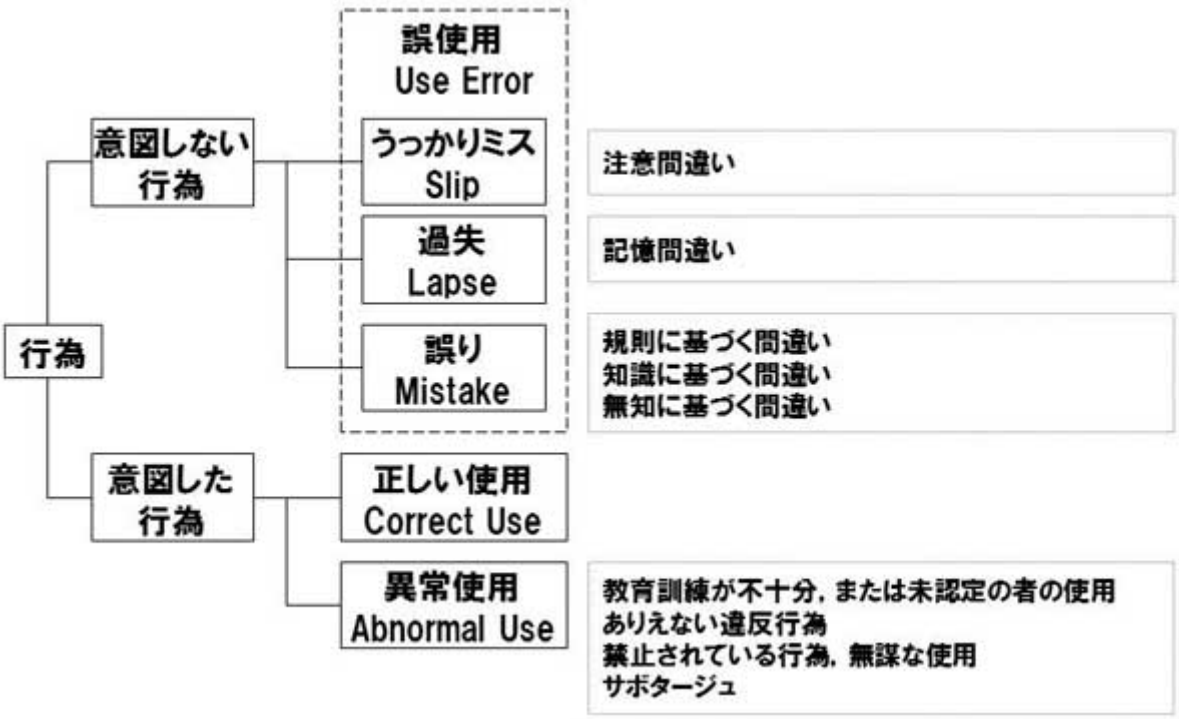
機器に搭載されるソフトウェアが増加し，使いやすさ（ユーザビリティ）が増したのと同時に，機器の機能が爆発的に増え，開発者が想定していなかったような使い方をユーザが行ってしまうケースも増加しています．そして，ユーザの行為が危害に結びついてしまうこともあります．

図 1 はユーザの機器に対する行為を分類したものです．ユーザが意図しない行為は誤使用（Use Error）となり，うっかりミス（Slip），過失（Lapse），誤り（Mistake）の三つに分類されます．うっかりミス（Slip）は不注意によるもので，過失（Lapse）は記憶違いによるもの，誤り（Mistake）は規則に基づく間違い，知識に基づく間違い，無知に基づく間違いのいずれかです．

見本

図 1　機器の使用者(ユーザ)の行為を分類する

IEC 62366:2007 Medical devices -- Application of usability engineering to medical devices Figure B.1 – Categories of foreseeable USER action （医療機器のユーザビリティ・エンジニアリング　図 B.1） より.



ユーザが意図した行為は，正しい使用(Correct Use)と異常使用(Abnormal Use)の二つに分類されます．すべてのユーザが開発者の想定した正しい使用を行ってくれればよいのですが，意図的に異常使用を行うケースもあります．プロフェッショナル・ユースの機器の場合，教育訓練を十分に受けていない使用者が，正しいと信じて誤った使用を行ってしまうこともあるでしょう．そのような状況を防ぐために機器の使用にライセンスを与える場合もあります．自動車の場合，運転免許がそれに当たります．

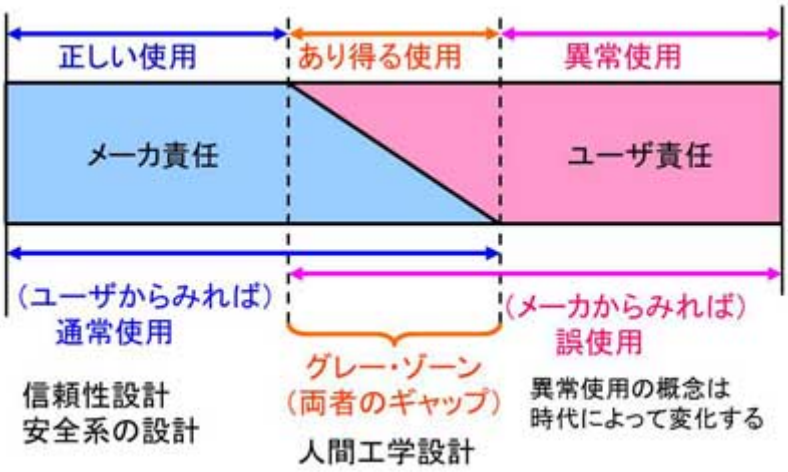
なお，正しい使用と異常使用の概念は絶対的なものではありません．正しい使用における問題はメーカの責任となりますが，異常使用における問題は基本的にはユーザ責任です[注1]．そして，正しい使用と異常使用の間にはグレー・ゾーンが存在します(図 2)．この「あり得る使用」は機器のメーカから見れば誤使用であり，ユーザから見れば通常使用です．このように，メーカから見たときの誤使用とユーザから見たときの通常使用にはオーバラップしている部分があります．この範囲は両者の認識のギャップであり，機器メーカはこのギャップをできるだけ埋めなければなりません．

見本

注 1. コンピュータ・ウイルスなどのセキュリティに関連する問題は，ユーザ以外が介入する悪意のある異常使用と考えられる.

図 2　時代とともに変わる正しい使用と異常使用の概念

早稲田大学　理工学術院　経営システム工学科　人間生活工学研究室　小松原　明哲教授の講演資料より.



安全の水準は時とともに変化しています．機器メーカが安全対策を機器に施し，それが普及すると，安全対策のない機器は危険な機器と判断されるようになります．安全は絶対的な概念ではなく，時代とともに変化する相対的な概念なのです．そして，社会の成熟度が高まるにつれ，安全に対する要求や，機器メーカが越えなければならないハードルは徐々に高くなっていきます．

●ソフトウェアに関連するリスク

ソフトウェアはよく「中身がどうなっているのか見えない」と言われます．従って，ソフトウェアの構造や品質を可視化する取り組みや研究が長年続けられていました．ソフトウェア・システムの中身が見えにくいからといって安全や信頼に対する対応を放置していたら，ソフトウェアはブラック・ボックスのままであり，重大な事故を起こすリスクを排除することができません．重大なリスクとは，医療機器で言えば，患者さんに重傷を負わせるような巨大なエネルギーを出力してしまうリスク，自動車で言えば，ブレーキをかけたいときにブレーキ制御が効かない，エアバックが動作しない，といったリスクです．

では，このようなソフトウェアに起因するリスクが具現化し障害に至らないようにするためには，何をすればよいのでしょうか．各産業はどんな取り組みを行ってきたのでしょうか．それを理解するためには，ソフトウェア品質論の歴史的背景を知る必要があります[注2]．

見本

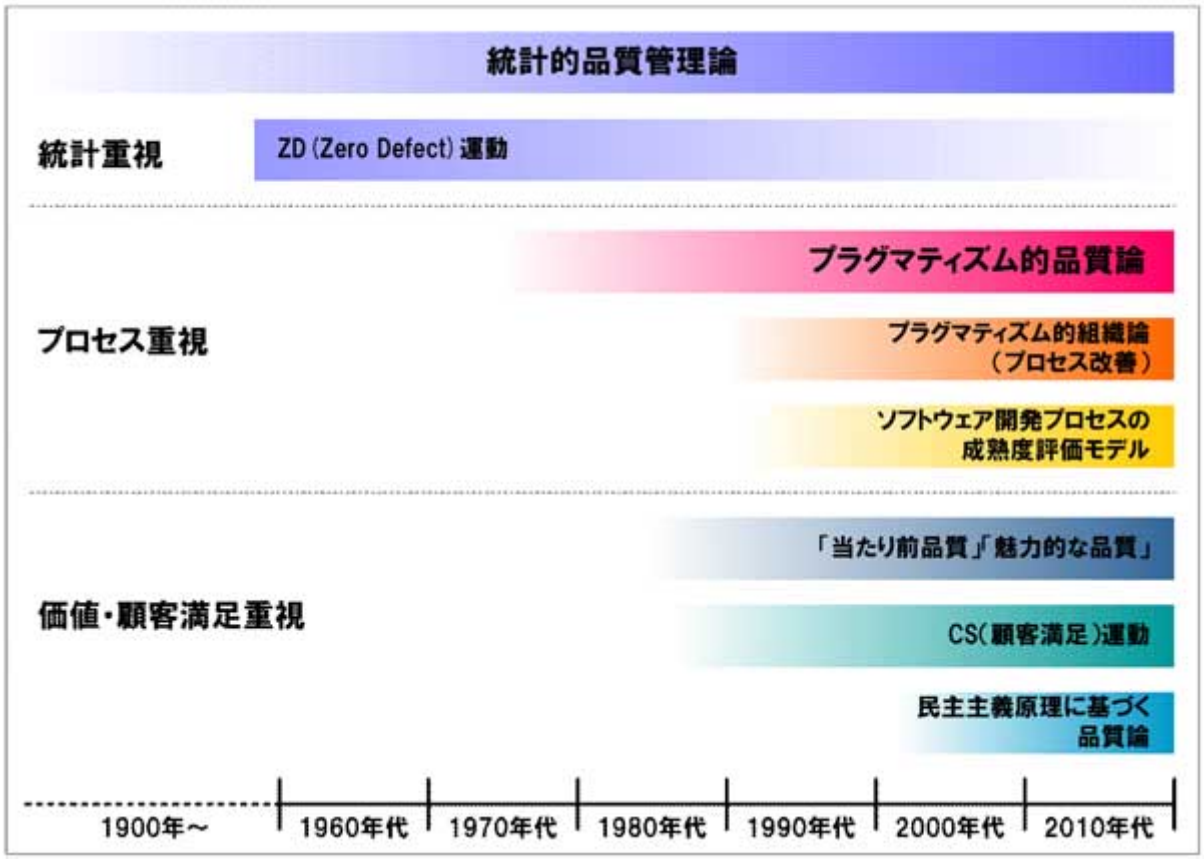
注 2：ソフトウェア品質の向上とシステムの安全確保は同義ではないが，幅広く関連している．

●統計重視からプロセス重視，顧客満足重視へ

ソフトウェア品質論の歴史的推移を図 3 に示します(2*)．1960 年代以前から現在に至るまで，見えないソフトウェアの品質を確保する考え方として常に主導的な役割を果たしてきたのが，統計的品質管理論です．統計的品質管理論は，ソフトウェアのみならず，ハードウェアも含めた品質管理の基礎となっています．

図 3　ソフトウェア品質論の歴史的推移

広島市立大学　教授　大場　充氏の講演「ソフトウェア品質って何だ～民主主義原理に基づくソフトウェア品質論入門～」を参考に作成した(2*)．



1960 年代にはソフトウェアの品質管理として ZD 運動（Zero Defect 運動；欠陥をゼロにする運動）が行われていましたが，ソフトウェアで欠陥（すなわちバグ）をゼロにすることは不可能であることが次第に明らかになり，ソフトウェアの世界では ZD 運動は衰退していきました．その後，

見本

台頭してきたのが，プラグマティズム品質論，すなわちソフトウェアの開発プロセスに着目してソフトウェア品質を高める考え方です．プロセス重視の品質論，プロセス改善の組織論，組織の成熟度モデルの研究は 1990 年代から欧米を中心に盛んに行われるようになり，現在でもソフトウェア品質に対する絶対的な地位を築いています．

ソフトウェア開発プロセス重視の品質論が時間の経過とともに，詳細かつ強固に補強されていく一方で，ソフトウェアの開発現場では小さいプロセスを機敏に適用するアジャイル開発を支持するプロジェクトも増えてきています．プロセス重視の品質論の基本は，ウォータ・フォールや V 字のプロセス・モデルを想定し，各工程においてインプットとアウトプットをきっちり確認しながら進めていくという考え方です．一方，下流工程であっても大量の変更が発生する現実に直面しているソフトウェア開発プロジェクトの中には，プロセス管理がソフトウェアの品質向上に貢献するという考え方に異を唱え，アジャイル開発の方が結果的に品質が高まる，と主張するグループもいます．

プロセス管理重視のソフトウェア品質論が台頭してきた一方で，日本では「当たり前品質」，「魅力的な品質」といった，商品の価値に焦点を当てた考え方があります．また，顧客満足を高める活動，いわゆる CS（Customer Satisfaction）運動がソフトウェアの品質向上に貢献する，という視点も追加されつつあります．

このように，現代のソフトウェア品質論は「統計的品質管理論」，「ソフトウェア開発プロセス管理」，「商品の価値・顧客満足重視」という三つの大きな考え方が共存していると考えられます．

●求められる安全要求に応える，という発想

要求される安全レベルの指標として，IEC 61508（機能安全規格）では SIL（Safety Integrity Level：安全度水準）が，ISO 26262（自動車の機能安全規格）では ASIL（Automotive Safety Integrity Level：自動車安全度水準）が定義されています．また，医療機器のプロセス規格である IEC 62304（医療機器ソフトウェア ― ソフトウェア・ライフサイクル・プロセス）では，ソフトウェア安全クラスが定義されています．

これらの指標はそれぞれ，ソフトウェアを搭載したシステムに対して，どれくらいのレベルの安全が要求されるのかをクラス分類し，そのクラスによってクリティカルなソフトウェア・システムに対する要求を変えようとしています．当然のことながら，安全度水準やソフトウェア安全クラスが高いと判定された場合の要求は，クラスが低いときよりも広く深くなります．IEC 62304 において，ソフトウェア安全クラスが A，B，C のうち一番高い C であった場合，要求されるプロセスとプロセス中に実施するアクティビティやタスクの種類が最も多くなります．

見本

この考え方は，図 3（ソフトウェア品質論の歴史的推移）に照らし合わせると，「ソフトウェア開発プロセス管理」と「商品の価値・顧客満足重視」の融合ととらえることができるでしょう．

安全度水準やソフトウェア安全クラスといった視点は，損なわれるとユーザが危害を被る「当たり前品質」の度合いと見ることができます．損なうと顧客満足を著しく低減させる商品の価値を安全度水準やソフトウェア安全クラスで格付けし，その度合いに応じて，適用するプロセス，アクティビティ，タスクの割り当てを変えていることから，ソフトウェア品質論の「ソフトウェア開発プロセス管理」と「商品の価値・顧客満足重視」を融合させていると考えることができます．この考え方は，2000 年代のソフトウェア品質論の主流となるかもしれません．

参考・引用*文献

（1）U.S. Food and Drug Administration；General Principles of Software Validation, Version 1.1, dated June 9, 1997. ※なお，2002 年 1 月に本ガイダンスの最終版が策定されている．この最終版は，米国 FDA の Web サイトで参照できる．

（2*）大場 充；ソフトウェア品質って何だ～民主主義原理に基づくソフトウェア品質論入門～，ソフトウェア品質シンポジウム 2010（SQiP2010）企画セッション 4-3 論文，2010 年 9 月．

見本

2. ISO 26262 の適合証明と米国 FDA の妥当性確認

自動車の機能安全規格である「ISO 26262」の「Part 8: Supporting processes（支援プロセス）」には，「ソフトウェア・ツールの使用への信頼」という章があります．この章にはソフトウェア・ツールに対する信頼の要求水準を決めるための基準が書かれているのですが，この基準は，「そのソフトウェア・ツールを使用することが適切である」という証拠を作成するための認定手段として使えることをも目的としている，と書かれています．近年のソフトウェア安全関連の国際規格で，商業目的につながることが予想できるツール種別の推奨やツール使用の条件のような内容を規格の本文に書いているのは，あまり見たことがありません．

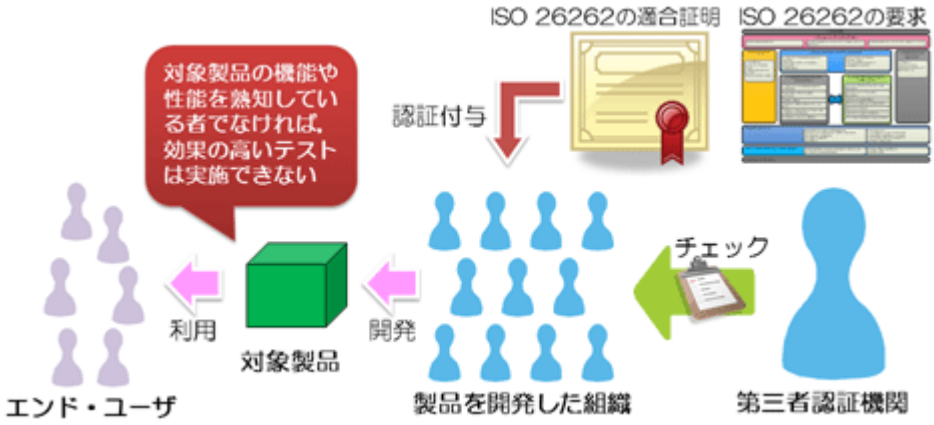
米国 FAA（連邦航空局）では，当初，航空機開発を行う際に使用するツール認証を一般的なソフトウェア開発プロセスの判断基準で行っていました．しかし，さまざまなツールが使用されている実態に対して，このツール認証の制約は合わないことが分かったため，今では止めているそうです．

●ISO 26262 の適合証明は組織やドキュメントを監査したものにすぎない

ISO 26262 における適合証明の位置付けを**図 1** に示します．ISO 26262 の第三者認証機関は，ISO 26262 の要求に基づいて，対象製品が要求を満たしているかどうかをチェックし，要求を満たしていることが確認できると ISO 26262 の適合証明書を発行します．これは ISO 9001（品質マネジメント・システム）の監査方法と同じです．

見本

図 1　第三者認証と開発組織，対象製品の関係



しかし，気をつけなければいけないのは，この方法は対象製品を直接検証しているわけではないという点です．第三者認証機関は，対象となる組織やプロジェクトの活動状況，そしてプロジェクトが作成したドキュメントを監査しているにすぎません．また，ISO 26262 では，ツール・ベンダが適合証明を取得することになりますが，ツール・ベンダは，ツールやシステムが使われる場面，用途を必ずしも熟知しているわけではありません．

医薬品や医療機器の世界では，米国 FDA や GAMP5(GAMP: Good Automated Manufacturing Practice)が，コンピュータ・システムに関する妥当性確認（CSV；Computerized System Validation）を製造業者に求めています[(1)]（図 2）．CSV の目的は，クリティカルな製品を開発・製造する際に，ツールや自動工程で使うソフトウェアの妥当性を評価することです．簡単に言えば，「危ないものを作るのに何らかのソフトウェアを使っているのであれば，それが製品の安全を脅かしていないことを確認して根拠を示せ」という要求です．この際に，根拠を示さなければいけないのは製造業者側であり，ツール・ベンダではありません（ツール・ベンダやサプライヤと協力して根拠を示すことはある）．

見本

図 2　CSV（Computerized System Validation）の流れ

出典：『GAMP ガイド：コンピュータ化システムの GxP 適合へのリスクベースアプローチ（GAMP5）』[2*]．GAMP（Good Automated Manufacturing Practice）ガイドとは，主に医薬品製造メーカを対象とした，CSV を行うための実用的なガイダンスであり，ISPE（International Society for Pharmaceutical Engineering；国際製薬技術協会）が発行している．最新版である GAMP5 は，英語版が 2008 年 2 月に，日本語版が 2009 年 3 月に発行された．

CSV では，対象とするソフトウェアのカテゴリごとに，確認するべき事柄を定義しています（**表 1**）．OS やミドルウェア，データベースなどの基盤ソフトウェア（カテゴリ 1）や市販ソフトウェア（カテゴリ 3）など，買ってきてそのまま使う（または，設定を少々変更する程度の）ソフトウェア製品であっても，妥当性確認の計画や報告，運用時の点検などは，ソフトウェア製品を使う側である製造業者が行わなければいけません．

見本

表 1　CSV のカテゴリ分類（GAMP5 による）

CSV の対象となるソフトウェア・カテゴリ		ソフトウェアの例	一般的なアプローチ
1	基盤ソフトウェア	OS，データベース，コンパイラ，ミドルウェア，ラダー・ロジック・インタプリタ，統計プログラミング・ツール	バージョン番号を記録し，承認済みのインストール手順に従って正しくインストールされていることを検証する
2	（現在このカテゴリは使用しない）		
3	構成を設定していない製品	業務目的に使用される市販製品で，デフォルトの構成のまま使用するもの	1 に加え，リスク・ベースのテストを実施する．また，適合性を維持するための手順を実施する
4	構成を設定している製品	ユーザのビジネス・プロセスのニーズに合わせた構成設定が可能なソフトウェア製品	1 と 3 に加え，ライフサイクル・アプローチおよびサプライヤの品質監査，データ管理のための手順を実施する
5	カスタム・アプリケーション	規制対象企業に特有のニーズを満たすために開発されるソフトウェア	1 と 3 と 4 に加え，厳密なサプライヤ検査，完全なライフサイクル文書の保管，デザイン・レビュー，ソース・コード・レビューを実施する

●「安全の証明」には保証が伴う

品質を評価することと安全を証明することは，似ているようで異なります．最大の相違点は「安全の証明」には保証が伴うということです．安全の向こう側には，重要度の差はあるものの必ず軽減するべきリスクが存在します．リスクが具現化することを防いでいるからこそ安全が確保されるのであって，安全を表明するためにはリスクが受容できるまでに軽減できていることを示す根拠が必要です．

見本

図 1 で第三者認証機関が発行する ISO 26262 の適合証明は，対象製品を開発した組織が ISO 26262 の要求を満たしていることを証明しているのであり，ユーザに対して対象製品そのものの安全性を保証しているわけではありません．

なお，医療機器の場合，電気的安全性については独立した試験機関が規格（IEC 60601-1）に基づいて，対象製品そのものを試験し，検査結果の証明書を発行します．検査機関自身も ISO 17025（試験所および校正機関の能力に関する一般要求事項）の認証を受けていれば，CB スキーム(Certification Body Scheme)[注 1] のもとでその試験データを広く規制当局が求める認証に使うことができます．

注 1：CB スキームとは，複数ヵ国に対する製品認証試験の手続きを簡単にするためのしくみである．この制度の加盟国にある試験所に試験を依頼し，CB 試験レポートと CB 証明書を発行してもらうことで，他の加盟国の認証機関において，認証に必要な試験を省略できる．

ソフトウェアの品質は，電気的安全性の試験で「何ボルト±10%以内に入っていれば合格」といったように，一般化された検査基準を確立することが困難です．だからこそ，前章で述べたように，ソフトウェアの開発プロセスがソフトウェア品質論の中心になっているわけです．

しかし，「ソフトウェア開発プロセスが要求通りにできているから，そのプロセスを使って作った製品は安全である」と言い切れるでしょうか？ ソフトウェア・プロジェクトのプロセス管理能力が高いと，そのプロジェクトがアウトプットするソフトウェア製品の品質が高いという相関はあるかもしれませんが，このことは対象製品の安全を保証しているわけではない，ということを忘れてはいけません．

第三者認証のすべてが保証を伴わないかと言えば，そんなことはありません．例えば，日本では国（厚生労働省）が医薬品に対して許認可を与えていますが，認可した医薬品に重大な問題が起これば，国もその責任の一端を担います．実際，薬害エイズ訴訟，B 型肝炎訴訟，C 型肝炎訴訟では国が責任の一部を認め，被害者に賠償を行っています．第三者認証には保証というエンド・ユーザに対する責任を負う場合と，要求への適合を確認するだけでエンド・ユーザに対する責任を負わない場合があることを認識することが重要です．

セーフティ・クリティカル・ソフトウェアを開発する組織は，第三者による規格適合証明を「安全のお墨付き」と考えてはいけません．対象となるソフトウェアが見えにくいから，評価が難しいから，という理由で，対象製品そのものの検証から目をそらしていては，安全の証明に近づくことはできないのです．安全を確保するためには，安全の反対側にある具体的なリスクを想定する

見本

ことが必要であり，それらのリスクを具現化させないリスク・コントロール手段の設計と検証，妥当性確認を行わなければ，安全の証明にはならないのです．

安全の証明には必ず責任が伴います．安全に対して責任を持つ者は，どのような根拠で安全を証明するのか説明できるようにしておく必要があります．国際規格の適合証明を根拠の一つに使うのであれば，それが対象製品を使うユーザの安全にどのように寄与しているのか説明できるようにしておく必要があります．適合証明を過大に評価していると，実質的な製品安全からかえって遠ざかる危険さえあります．このことを，セーフティ・クリティカル・ソフトウェアの開発組織は認識しなければいけません．

見本

3. 市販ソフトウェアに求められるリスク分析と検証の実態

市販されている既製のソフトウェア製品をシステムに組み込む場合，検証が必要なソフトウェアという意味で，OTS（off-the-shelf）ソフトウェア，またはCOTS（Commercial off-the-shelf）ソフトウェアと呼びます．OS（Windowsや各種リアルタイムOSなど）やTCP/IPプロトコル・スタック，SDカード・ドライバ，ファイル・システムなどの市販ソフトウェアが，これに相当します．また，Linuxなどの無償のオープン・ソース・ソフトウェアも，検証が必要なソフトウェアという意味では市販ソフトウェアと同じです．

機器の安全を確保する際に市販ソフトウェアに着目する必要がある理由は，市販ソフトウェアがクリティカル・デバイスに使用されることを目的として開発・販売されていないからです．できるものならば，クリティカル・デバイスに使用されることを前提として専用のソフトウェアを毎回作ることが望ましいのですが，それは現実的ではありません．打ち上げが失敗すれば莫大な損出を被る宇宙事業でさえ，市販ソフトウェアは使われています．

●SOUP（Software of Unknown Provenance）とは

市販ソフトウェア（OTSソフトウェア）のほかに，SOUP（Software of Unknown Provenance；スープと発音）という概念があります．SOUPは「由来の分からないソフトウェア」という意味で，市販ソフトウェア以外でも，自組織で以前に開発したものでも，開発プロセスや検証記録などが十分でないというものも含みます．かつてクリティカル・デバイスに組み込むことを目的に開発されたソフトウェアであっても，必要とされる技術水準で開発されていないソフトウェアはSOUPと見なされ，市販ソフトウェアと同じようにそれがクリティカル・デバイスに組み込まれても大丈夫かどうかを確認する必要があるのです．

●市販ソフトウェアに爆弾が含まれていたら

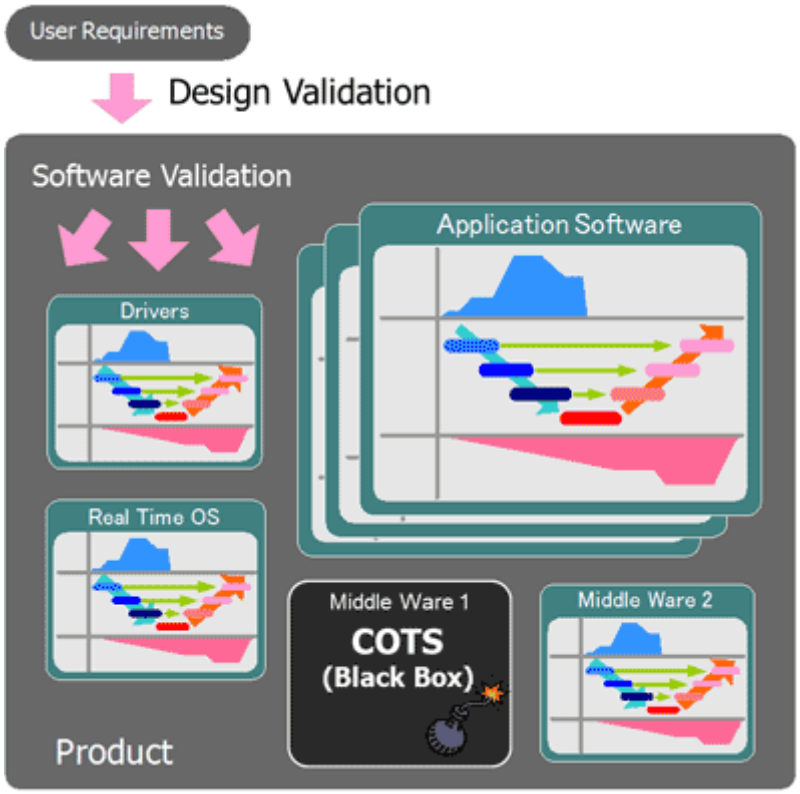
製品に使うために購入した市販ソフトウェアの中身は，多くの場合ブラック・ボックスです．ソース・コードが公開されていても検証結果が何もなければ，それはブラック・ボックスであるのと同じです．市販ソフトウェアには爆弾が含まれていないとは限りません．市販ソフトウェアの開

見本

発者側には些細なバグであっても，それを利用しているクリティカル・システムにとっては重大な障害に結びつくことがあります．

市販ソフトウェアに含まれる爆弾は，ソフトウェア妥当性確認（Software Validation）や設計の妥当性確認（Design Validation），ソフトウェア検証（Software Verification）によって取り除きます（図 1）．具体的には市販ソフトウェアに対して機能テストを行ったり，サブシステム同士の結合テストや製品の機能テストを行ったりして，爆弾を見つけます．見つけた爆弾は，ソース・コードが提供されていない場合は市販ソフトウェアの供給者に取り除いてもらうしかありません．これは社内で過去に作成した中身の分からないサブシステム（レガシ・ソフトウェア）でも同じです．

図 1　市販ソフトウェアに爆弾が含まれていたら



そして，検証が十分に行われ，市場で長い時間使用されたサブシステム・ソフトウェアは妥当性確認を経て再利用できます．市販ソフトウェアの検証責任は，基本的には市販ソフトウェア

見本

を組み込む(使う)側にあります. 検証の一部を市販ソフトウェアの提供者に委託することはできますが, 最終的な検証責任は, 市販ソフトウェアを使う側が負わなければなりません.

●市販ソフトウェアの信頼性証明に必要な情報

市販ソフトウェアをクリティカル・システムに使用するにあたり, どのようなドキュメントを作成するべきでしょうか？ 市販ソフトウェアを検証した記録は当然必要ですが, それだけでは不十分です. 検証記録以上に重要なのは, 対象の機器やシステムの中で市販ソフトウェアがどのような役割を果たすのかを分析したレポートです.

米国 FDA が発行している, 医療機器に市販ソフトウェアを使用する場合のガイダンス「Guidance for Industry, FDA Reviewers and Compliance on Off-The-Shelf Software Use in Medical Devices」の目次を, **表 2** に示します.

表 2 市販ソフトウェアの信頼性証明に必要な情報の例

市販ソフトウェア・レポート	市販ソフトウェアの検証記録（※）
● 市販ソフトウェアの説明 ● 市販ソフトウェアの使用条件 ● 市販ソフトウェアが機器内で提供する機能 ● 市販ソフトウェアの懸念レベル ● 市販ソフトウェアの管理方法 ● 市販ソフトウェアのハザード分析 ● 市販ソフトウェアの検証結果と判定（詳細は右記※）	● テスト計画 ● テスト仕様書 ● テスト成績書 ● テスト結果サマリ - バージョン - 試験者, 試験場所, 試験日時など

それでは, 市販ソフトウェア・レポートの具体的な内容を見てみましょう. まず, 市販ソフトウェアの説明と使用条件は, その市販ソフトウェアがそもそもどのようなものであるか, どのような用途を想定して作られたものかを示すものです. 特に使用条件は, そのソフトウェアの開発元が示したものなので, 必ず記録し, その条件を満たしていることを確認する必要があります.

見本