【P.18】

注記 1：リスクマネジメントファイルを構成する記録及びその他の文書は、例えば製造業者の品質マネジメントシステムによって要求される他の文書やファイルの一部とすることができる。リスクマネジメントファイルは、すべての記録及びその他の文書を物理的に含んでいる必要はない。ただし、少なくともすべての要求書類に対する参照表記あるいは指示表記を含んでいることが望ましい。製造業者は、リスクマネジメントファイルで参照表記する情報をタイムリーにまとめることができるのが望ましい。

注記 2：リスクマネジメントファイルの書式や媒体の種類は問わない。

3.5.1 一般

ISO 14971:2007 リスクマネジメントファイルの要求事項の重要な要素として強調すべきものは、リスク分析、リスク評価、及びリスクコントロール手段の実装と検証に対して、特定した各ハザードのトレーサビリティを提供する必要性である。このトレーサビリティには、ハザードのソフトウェアによる原因、使用者のためのソフトウェアリスクコントロール手段、ハザードのハードウェア又はソフトウェアによる原因、リスクコントロール手段を実行するソフトウェア、及び適切な機能を検証するソフトウェア試験を含めるのが望ましいのは明らかである。

ソフトウェアプロセスでは、ソフトウェア関連のハザード及びソフトウェアリスクコントロール手段から対応するソフトウェアアイテム、安全性に関係するソフトウェアの要求事項及びその検証に至るまで、このトレーサビリティを可能にするシステムを構築しなければならない（IEC 62304:2006 箇条 7.3.3 *文書のトレーサビリティ* (IEC 62304:2006 Clause 7.3.3 Document Traceability)参照）。

またこのトレーサビリティは、分析したソフトウェアの判明したバージョンの文書化も考慮に入れることが望ましい。この時点以降にソフトウェアに対して行われるメンテナンスでは、リスクコントロールが確実に維持されるようにする必要がある。

IEC 62304:2006 箇条 4 でも、ソフトウェア安全クラス（A、B、又は C）をリスクコントロール手段の実装に寄与する各ソフトウェアシステムに割り当てること、及びこの割り当てをリスクマネジメントファイルに文書化することを製造業者に求めている。なお、ソフトウェアシステムがソフトウェアアイテムに分解されそれらを別々に分類できるように製造業者が分離している場合は、その細分類の根拠を文書化し、リスクマネジメントファイルに入れるのが望ましい。

次の表に、ISO 14971:2007 の要求事項に追加して、リスクマネジメントファイルに含めるべき文書の IEC 62304:2006 要求事項を掲げる。

IEC 62304:2006 箇条番号	リスクマネジメントファイルに含める文書内容
4.3c)	各ソフトウェアシステムに割当てたソフトウェア安全クラス
4.3f)	安全関連機能を実装しないソフトウェアシステムのソフトウェアアイテムに上位のソフトウェア安全クラス（ひいてはソフトウェアシステム）を使用する根拠
7.1.4	ソフトウェアアイテムがハザード状態の一因となる場合の潜在的原因
7.1.5	IEC 62304:2006 箇条 7.1.2 に明示されているハザード状態を招くおそれのあるイベントシーケンス
7.2.1	ハザード状態の一要因となるソフトウェアアイテムの潜在的原因のそれぞれについて定めたリスクコントロール手段
7.3.2	リスクコントロール手段をソフトウェアアイテムとして実装する場合、製造業者はそのリスクコントロール手段を評価して、ハザード状態を招くおそれのある新しいイベントシーケンスを明らかにし文書化すること。

【P.19】

9.5	製造業者は問題報告及びその解決策（検証を含む）の記録を保守すること。製造業者はリスクマネジメントファイルを適宜更新すること。

4 リスク分析

4.1 リスク分析プロセス

ISO 14971:2007から抜粋

4 リスク分析

4.1 リスク分析プロセス

リスク分析は、4.2 から 4.4 までに記載する特定の医療機器について実行すること。計画したリスク分析活動の実施及びリスク分析の結果は、リスクマネジメントファイルに記録すること。

注記 1：リスク分析又はその他関連情報が類似の医療機器について利用可能な場合、その分析又は情報を新しい分析の出発点として使用できる。関連性の度合いは機器間の差によって、及びその差が新しいハザードを招くのかあるいはアウトプット、特性、性能、若しくは結果に対して重大な差をもたらすのかによって、異なる。また既存の分析結果をどの程度使用するかは、ハザード状態の進展に当該変更が及ぼす影響の体系的評価に基づく。
注記 2：リスク分析手法のいくつかを、付属書 G に記載する。
注記 3：生体外診断医療機器のリスク分析手法に関する新たな指針を、付属書 H に示す。
注記 4：毒物学的ハザードのリスク分析手法に関する新たな指針を、付属書 I に示す。4.2 から 4.4 の中で要求している記録に加え、リスク分析の実施及び結果の文書記録には、少なくとも以下を含めること：

a) 分析した医療機器についての説明及び識別情報
b) リスク分析を実行した者及び団体についての識別情報
c) リスク分析の範囲及び実施日

注 5：リスク分析の範囲は、（製造業者がほとんど又は全く経験がない新しい機器の開発については）非常に広範にわたる可能性もあれば、（製造業者のファイルに多くの情報が既に存在する既存の機器への変更の分析については）非常に限定される可能性もある。

適合性は、リスクマネジメントファイルの検査によって確認する。

リスク分析を様々な観点から検討することは有効である。その中には、ハザードとその考えられる発生方法を特定し、ハザードシナリオとなるイベントシーケンスを、根本原因にたどり着くまで下っていくというトップダウンアプローチが含まれる。このトップダウンアプローチの一般的な手法として、フォルトツリー解析（故障の木解析）がある（IEC 61025[3]参照）。この手法は、ハザードシナリオにつながる具体的事象の特定に役立つ。表 B.1（直接的原因の例）では、フォルトツリー解析アプローチを使用して特定可能な具体的事象の例をいくつか示す。

リスクの特定に役立つアプローチには、起こりうる故障を特定することろから始め、次にその故障の結果の特定と評価を行うというアプローチもある。この「ボトムアップ」アプローチは、故障モード影響解析法（FMEA）を使用して行われることが多い（IEC 60812[2]）。ソフトウェアについては、このアプローチは予測できない挙動から発生するハザードシナリオの分析に役立つ場合がある。表 C.1（疎結合原因の例）に、予測できない挙動を招くおそれのあるソフトウェア欠陥の例を示す。

【P.20】

ハザード状態の特定に役立つ可能性のある第 3 のアプローチは、機器の動作を分析し、その動作が意図する使用から逸脱する度合いに着目するというアプローチである。このアプローチに使用される手法には、ハザード動作性調査（HAZOP）がある（IEC 61882[4]参照）。

これらの手法を使用してハザード確率を定量的に評価する手順は開発されているが、ソフトウェアの分析ではそれに頼らないことが重要である。ソフトウェア故障率を評価するのは困難なため、リスク評価の決定に必要な定量分析結果の信頼性水準が得られない。

医療機器のソフトウェアの説明では、以下の情報を提供するのが望ましい：

- 医療機器におけるソフトウェアの用途。ハザードシナリオの特定においてソフトウェアを含めるべき時点を知るために必要である。

- リスク分析に含まれるソフトウェアの境界。根本的原因が分析中のソフトウェアの外で発生する可能性がある一方で、ソフトウェアに実装されるリスクコントロール手段はすべて、リスク分析に含まれるソフトウェアシステムの内部に実装されている。分析対象のソフトウェアシステムがソフトウェアシステムの境界外にあるハードウェア／ソフトウェアプラットフォームを利用する場合などのように、環境が変化する可能性がある場合、この点は特に重要である。

- 分析したソフトウェアの判明したバージョン。リスク分析はソフトウェアの実装前に開始するのが望ましいが、リスクマネジメント報告書には判明したバージョンを明瞭に記載する必要がある。この時点以降にソフトウェアに対して行うメンテナンスは、リスクコントロールが確実に維持されるようにするものでなければならない。

ISO 14971:2007 に記載されているように、リスク分析は 3 つの別個の活動を包括して使用される用語である。

- 意図する使用の特定

- 既知の又は予見可能なハザード（及びその原因）の特定

- 各ハザードのリスクの推定

リスク分析が有効であるためには、1 つ又は 2 つの離散事象としてではなく、ソフトウェア開発プロセス全体の不可欠な一部として実行しなければならないということを認識しておくことが非常に大切である。というのも、ハザード及び故障モードに関する情報は、ソフトウェア開発ライフサイクルプロセス全体にわたって発生し、設計の各段階で考慮する必要があるためである。

ハザード状態を招くおそれのあるソフトウェア欠陥の確率は定量的に評価できないこと、及び損傷が原因で使用中にソフトウェア故障がランダムに発生することはないことから、リスク分析のソフトウェア面については確率の評価でなくハザード状態を招くおそれのある潜在的なソフトウェアの機能性及び欠陥の特定に主眼を置くべきである。

リスク分析においてソフトウェアが十分に検討されるようにするため、計画又は手順がソフトウェアリスクの評価及びコントロール特有な側面に対応していることが望ましい。これには、ソフトウェア開発ライフサイクルにおけるソフトウェアマネジメントプロセスへの対処方法も含めるとよい。ソフトウェアがその要因である最悪の場合のハザードの重大性は、リスクマネジメントプロセス及びソフトウェア開発プロセスの厳格度を決める上での最重要インプットである（IEC 62304:2006 第 4.3 項参照）。以下の細分箇条で提供する情報は、有効なリスクマネジメントプロセスのソフトウェアに特有な側面の特定に役立つように意図されたものである。加えて、リスク分析のソフトウェア面は、結果を記録した文書において確認できること、そしてハードウェア故障のリスクコントロール手段の実装に使用するソフトウェア並びにハザードのソフトウェア原因とそれに関わるリスクコントロール手段の両方を含んでいること、が望ましい。

【P.21】

4.2 意図する使用及び医療機器の安全に関する特質の明確化

ISO 14971:2007から抜粋
4.2 意図する使用及び医療機器の安全に関する特質の明確化

検討中の特定の医療機器について、製造業者は意図する使用及び合理的に予見可能な誤使用を文書化すること。製造業者は医療機器の安全性に影響を及ぼすおそれのある定性的及び定量的特質並びに、該当する場合は、その規定限界を明確にして文書化すること。この文書は、リスクマネジメントファイルで保守すること。

注記 1：この文脈で誤使用とは、医療機器の誤った又は不適切な使用を意味する。

注記 2：付属書 Cには、安全性に影響を与えるおそれのある医療機器の特質を明確にする際に指針として利用できる、使用上の疑問などをまとめたものが含まれている。

適合性は、リスクマネジメントファイルの検査によって確認する。

4.2.1 一般

医療機器は、多様なインプットに対して複雑な挙動を示すよう要求されることが多い。このような挙動は、通常はソフトウェアが提供する。

リスクマネジメントは、医療機器による危害を防止するために又はその程度を軽減するために、医療機器がどのように挙動すればいいか又どのように挙動してはならないかの両方を決めるために必要である。医療機器の意図する使用及び予見可能な誤使用に関する知識は、リスクマネジメントが完全かつ効果的に行われるようにするために必要である。

ソフトウェアの故障が危害を招いてはならないのは明らかである。ソフトウェアは、臨床上の機能性の提供に加え、わずかな不具合を検知するその能力並びに不具合が発生した際に危害を防止又は軽減すべく柔軟に挙動できる利点を生かして、リスクコントロール手段としても使用できる。この点を覚えておくとよい。多くの場合、ソフトウェアは故障時に以下の機能を実行する。意図する使用及び合理的に予見可能な誤使用を明確にする際にはこれらについても考慮するのが望ましい：
・故障状態を検知する；
・シャットダウンして安全な状態にすることにより、故障が危害を招くことを防止する；
・危害を防止又は軽減しつつ、故障状態から復帰する。
・故障状態を検知後、フルモード又は縮退モードで機能動作を継続する。

システム要求事項作成の責任者は、以下を実行するのが望ましい：

・上記機能をソフトウェア要求事項として文書化する；
・その情報をソフトウェア設計者に伝達する；
・意図する使用及び予見可能な誤使用をソフトウェア設計者に伝達して、彼らが要求事項の背景及び理由を理解できるようにする。
・ソフトウェア要求事項、特に安全に関する要求事項を実装する能力について、ソフトウェア設計者と協議する。

ソフトウェア設計者は、以下を実施するのが望ましい：
・ソフトウェア要求事項、特に安全に関する要求事項の実現可能性を確認する；
・実現不可能なソフトウェア要求事項について、代替案を提示する。

各医療機器は意図する使用を持つが、誤使用（意図的か否かを問わず）の可能性も考慮しておくことが望ましい。これはソフトウェア特有の懸念事項ではないが、以下の理由によりソフトウェアの使用が誤使用のリスク増大につながる可能性がある：
・医療機器の挙動が（他に比べて）複雑なため、習得や理解がそれだけ難しい；
・使用者が、その限度を理解することなく、ソフトウェアに過度に依存するようになる可能性がある；

【P.22】

・医療機器は設定変更可能な場合があり、使用者が現在の設定を理解していない場合がある。

・医療機器製造業者が詳細について予想できないような方法で、医療機器が他の医療機器及び非医療機器と通信する場合がある。

ソフトウェアは、より柔軟なユーザーインターフェースの設計を可能にするが、それが使用者の挙動に影響を及ぼし、ひいては新しい形態の予見可能な誤使用につながる場合がある。起こりがちな誤使用は、過度に複雑なユーザーインターフェースに対する誤解や、エラーや非安全状態の回避についてのソフトウェアへの過度の依存から発生する。このような誤使用を予期し、できる限り回避するために設計変更を行うことが重要である。

ISO 14971:2007 を補完するユーザビリティプロセスについては、IEC 62366[5]を参照のこと。

システム要求事項作成の責任者及びソフトウェア設計者は、ソフトウェアを含むシステムの意図する使用を、安全性及び安全使用に関連するすべてのシステム／ソフトウェア要求事項とともにリスクマネジメントファイルに記録することについて、共同責任を負う。ソフトウェア設計者は、システムレベルでは識別できないほど微妙な意図する使用の特徴を明らかにすることについて、特に責任を負う。

4.2.2 医療機器の相互接続

医療機器でソフトウェアを使用することにより、医療機器と非医療機器の間での広範な相互接続及び相互通信が可能になる。このような接続及び通信は、医療機器や相互接続された機器で構成されるシステムの新しい使用（及び誤使用）を生み出す可能性が高い。これらの新しい使用や誤使用が起こりうることは容易に予見できるが、相互接続及び相互通信に制限がない場合、医療機器製造業者がこれらの使用や誤使用をすべて特定することは容易ではない。

したがって製造業者は、医療機器の通信インターフェースについて意図する使用の限られたセットを明らかにして、相互接続及び相互通信を安全なものに制限するインターフェースをできる限り設計することが重要である。これが不可能な場合、製造業者は医療機器との接続を行う人向けに、該当する場合は使用者との法的拘束力を持つ契約に裏打ちされた、安全な接続及び通信を推奨する附属文書を作成することが望ましい。

例えば、医用機器の内蔵インターフェースを使用して作成された治療データについて、使用者や患者の身元及びデータ作成の背景に基づき、一貫性及び合理性をチェックする場合がある。データが別の場所で作成されネットワーク接続を使用して医療機器にインポートされた場合は、これと同じチェックが適用できない可能性がある。そのような場合製造業者は、ネットワークアプリケーションとしてネットワーク使用者にチェック可能とさせること、及び／又はデータのインポートを信頼できる供給元に制限すること、及び臨床環境のネットワーク接続責任者のための包括的なマニュアルを作成することを検討するかもしれない。

い。

少なくとも、この例における製造業者は、ネットワーク接続からのデータのインポートと、内蔵インターフェースを使用したデータの作成を、それぞれが独自のリスクを持つ二つの別個の意図する使用として考えることが望ましい。

IEC 80001[6]は、臨床環境における医療機器の IT ネットワークへの統合を扱っている。特に、製造業者及び医療機器の IT ネットワークへの統合の実施者の責任を定めている。

4.3 ハザードの特定

ISO 14971:2007 から抜粋

4.3 ハザードの特定

製造業者は、正常状況及び不具合状況の両方における医療機器に関連する既知及び予見可能なハザードについて文書をまとめること。この文書は、リスクマネジメントファイルで保守すること。

【P.23】

注記： E.2 及び H.2.4 に示す起こりうるハザードの例は、製造業者がハザードの特定を開始する上での指針として使用できる。

適合性は、リスクマネジメントファイルの検査によって確認する。

4.3.1 一般

ハザード特定の目的は、すべての予見可能なハザードの分析、並びに有効なリスクコントロール手段の設計及び実施を可能にすることにある。

熱や電気エネルギー、懸垂部分とは違って、ソフトウェア自体はハザード源（危害の潜在的発生源）ではない。例えばソフトウェアとの接触によって負傷するおそれはない。しかし、ソフトウェアによって人がハザードにさらされる可能性はある。言い換えれば、ソフトウェアがハザード状態の原因になる可能性がある。

このように、ソフトウェアは新しいハザードを招くことはめったにないものの、ハザード状態を変化させることはよくある。製造業者にとってさらに重要なことは、ハザード状態回避の責任が使用者から製造業者に転嫁される可能性があることである。

例えば、外科用メスは明らかに切傷ハザードを有している。しかし、製造業者は従来から人間工学設計を超えた範囲でこのハザードの責任を負うことはしなかった。これは、このハザードが完全に外科医の管理下にあると想定されるためである。一方、外科用メスが遠隔手術システムの一部である場合にも同じハザードは存在するが、切傷ハザードを回避する責任の多くは、今やシステムにソフトウェアを組み込んだ製造業者にあるとされている。

つまり製造業者は、ソフトウェアが存在しない場合には自身の責任の範囲外であったハザードの一部について、医療機器でソフトウェアを使用するときには、それらを明らかにすることが今や求められているということである。

重要なケースは、データの誤処理による誤った治療のハザードである。これは常にハザードであったが、データが臨床医のデスクで処理された場合は、製造業者の責任外であった。現在、多くの医療機器はデータの生成、格納、操作、又は使用にソフトウェアを使用している。この結果、ハザードの責任の一部が製造業者に発生する。

ソフトウェアは以下を含むさまざまな形で、ハザード状態の原因になりうる（附属書 B 及び附属書 C も参照）。

- ソフトウェアが非安全システム要求事項を正確に実装しており、実際の危害が発生するまで危険な性質が認識されない挙動につながる可能性がある。

- ソフトウェア仕様がシステム要求事項を正確に実装していないため、ソフトウェア仕様上は正しいが望ましくない挙動につながる可能性がある。

- ソフトウェアの設計及び実装に不具合があり、ソフトウェア仕様に矛盾した挙動につながる可能性がある。明らかな不具合は、ソフトウェア仕様の誤解や、仕様をコード化する際のエラーから生じるかもしれない。分かりにくい不具合は、ソフトウェアアイテム間及び（ハードウェアやオペレーティングシステムを含む）インフラとソフトウェア間の、予見不能な相互作用から生じる可能性がある。

ソフトウェアを組込んだ医療機器では、注意深く包括的にハザードを特定することにより、（リスクマネジメントプロセスの後の段階において）特に以下の重要な成果を得られる可能性がある。

– ソフトウェアによる危害の発生を防止するハードウェアリスクコントロール手段

– 危害を及ぼす可能性のあるソフトウェア機能の仕様からの除去

【P.24】

- 危害の防止にソフトウェアを使用するリスクコントロール手段（IEC 62304:2006 箇条 5.2.3）

- 低欠陥密度で実装しなければならないソフトウェアの部分、及び特別試験の対象としなければならないソフトウェア仕様の部分、の特定（IEC 62304:2006 箇条 4.3）

- 予期せぬ悪影響から生じる危害の防止のため他の部分と分離しなければならないソフトウェアの部分の特定（IEC 62304:2006 箇条 4.3 及び箇条 5.3.5 参照）

ハザードを十分に明らかにするには、機器の臨床用途をよく理解しなければならない。また、ソフトウェアにはユーザーインターフェースが複雑であるなどの複雑性という特別な難しさがある。したがって、ソフトウェアハザードの特定を分離して行うことはできない。これは、臨床専門家、ソフトウェア設計者、システム設計者、使用性／人間工学の専門家を含む学際的チームによって、システムレベルで行わなければならない。

ハザードの特定では、医療機器の性質から起こりうる危害（例えば患者の切り傷、被ばく、あるいは感電）、さらにはソフトウェアの使用に関連して発生する新たなハザードを考慮するのが望ましい。後者の例として、以下が挙げられる：

- 患者の取り違い（医療機器に患者の詳細又は処方の情報が保存されている場合）

- ソフトウェアの故障による治療の遅れ又は拒否

特定するハザードは、仕様に従って動作しているソフトウェアに関連するハザードと、ソフトウェア欠陥に関するハザードの両方であることが望ましい（この文書の第 6.1 項も参照）。

多くの場合、医療機器のユーザーインターフェースはソフトウェアによってさらに複雑になっている。特に、ソフトウェアを組込んだ医療機器は情報を扱うことが多い。これは患者利益の点から正当化される場合もあるが、例えば次のような、誤った情報又は誤って使用された情報に関して新たなハザードを考慮することが望ましい：

- 誤ったデータ入力

- 使用者による表示誤読

- 使用者による警報の誤解又は無視

- 過剰なデータ又は過剰な警報数による使用者への過負荷（IEC 62366[5]参照）

IEC 62304:2006 細分箇条 4.3 において、ソフトウェアアイテムは、ソフトウェア故障から発生しうる危害の重大性をもとに 3 つのソフトウェア安全クラスに分類されている。IEC 62304:2006 は、ソフトウェアアイテムが複数のソフトウェア安全クラスに分かれる場合は、ソフトウェアアイテム間が十分に隔離されていることを実証するよう、製造業者に要求している。その目的は、下位のソフトウェア安全クラスのソフトウェアアイテムが上位のソフトウェア安全クラスのソフトウェアに及ぼす悪影響を防止することである。この点に関する詳細な検討については、この文書の第 6.2.2.5 項を参照すること。

各ハザード状態のリスク推定

ISO 14971:2007 から抜粋

4.4 各ハザード状態に関するリスクの推定

ハザード状態を招くおそれのある合理的に予見可能なイベントシーケンス又はイベントの組合せについて検討し、結果として生じたハザード状態を記録すること。

注記 1：過去に認識されていないハザード状態を明らかにするには、特定の状況を網羅する体系的手法を使用できる（附属書 G 参照）。

注記 2：ハザード状態の例は、H.2.4.5 及び E.4 で示している。

【P.25】

注記 3：ハザード状態はスリップ（動作のミス）、ラプス（記憶のミス）、及びミステイク（確認のミス）から生じうる。

特定した各ハザード状態について、利用可能な情報又はデータを使用してその関連リスクを推定する。危害の発生確率が推定不能なハザード状態については、リスク評価及びリスクコントロールで使用するために、考えられる結果を列挙する。これらの活動の結果は、リスクマネジメントファイルに記録する。

危害の発生確率又は危害の重大性を定性的又は定量的に分類するために使用するシステムすべてを、リスクマネジメントファイルに記録する。

注記 4：リスクの推定には、発生確率及び結果の分析が盛込まれている。用途によっては、リスク推定プロセスの一定要素だけが検討対象となるかもしれない。例えば、初期ハザードと結果の分析だけで十分な場合もある。D.3 も参照。

注記 5：リスクの推定は、定量的又は定性的に行うことができる。システム上の欠陥から生じるリスクの推定も含めたリスク推定手法は、附属書 D で示している。附属書 H では、生体外診断医療機器に関するリスクの推定に役立つ情報を提供している。

注記 6：リスクの推定に用いる情報又はデータは、例えば次から取得できる：

a) 公表規格

b) 科学技術データ

c) 公に報告されている事故を含む、既に使用中の類似医療機器の実地データ

d) 典型的な使用者を使ったユーザビリティ試験

e) 臨床上の証拠（エビデンス）

f) 適切な調査の結果

g) 専門家の意見

h) 外部品質評価スキーム

適合性は、リスクマネジメントファイルの検査によって確認する。

4.3.2 一般

リスクを推定するためには、まずソフトウェアを含むハザード状態を特定することが必要である。ソフトウェアは、ハザード状態の原因となるイベントシーケンスの根本的原因になる場合もあれば、ハードウェア故障の検出を意図したソフトウェアのように、シーケンスの他の原因になる場合もある。ソフトウェアには、SOUP コンポーネントや過去に開発したコンポーネントの再利用品が含まれる可能性もある。

リスク推定は、特定した各ハザード状態から生じる危害の確率及び危害の重大性に基づいて行われる。ソフトウェア故障はランダム故障ではなく決定論的原因（系統的）故障のため（箇条 4.3.2.3 参照）、危害の原因となるイベントシーケンスにおけるソフトウェア故障などハザード状態のリスク推定の際に発生するソフトウェア故障の確率を使用するときには、注意が必要である。

4.3.2.1 特定方法

ハザード状態におけるソフトウェアの潜在的役割の特定には、さまざまな手法が使用できる。これらの手法はアプローチが異なり、ソフトウェア開発の様々な段階で役立つ可能性がある。唯一の正しい手法というものは存在しない。

フォルトツリー解析（FTA）は、医療機器システムについて最初に使われることが多い、伝統的なトップダウン手法である（箇条 4.1 の FTA に関する適正事項を参照）。FTA は、主としてハザードの原因の分析に使用される。この手法では、ハザード状態が発生したと想定し、それがどのように発生しうるものかを特定するためにさかのぼっていく。フォルトツリーの各レベルは、ハザード状態が存在するために発生したに違いないイベントについてのより詳細な情報を提供する。FTA は、ハザード状態の原因となるイベントシーケンスに関与しているソフトウェアアイテムの特定に使用できる。

【P.26】

故障モード影響解析（FMEA）は、コンポーネントやサブシステム（ソフトウェアの場合はソフトウェアアイテム）から始め、「この要素が故障したとするとどのような結果になるか?」という問い掛けを行う、伝統的なボトムアップアプローチである。ここでは、ソフトウェア故障の種類は既知であり、結果として生じるソフトウェアの挙動が、ハザード状態の原因となるイベントシーケンスにどう寄与しているかについて分析することができる。箇条 4.1 の FMEA に関する追加審議事項を参照すること。

4.3.2.2 直接結合原因 対 疎結合原因

ハザード原因を特定するときには、機器の基本的性能及び関連ハザードの個別の要因[7]に直接結び付いているソフトウェア（例：血糖値を計算するアルゴリズム）に焦点を当てるのが最も簡単である。これらのソフトウェア原因は、FTA などのトップダウン手法を使用して特定できる。しかしながら、軽微な故障モードを招くおそれがありしたがって一つ以上の機器ハザードを発生させる可能性のあるソフトウェアエラーを考慮することも重要である。このように直接結合原因の例には、初期化されていないポインタ、メモリ破損、スタックオーバーフロー、競合状態などがある（その他の例については、附属書 B を参照）。FMEA などのボトムアップ手法の使用は、その予測が非常に難しい場合があるため、これらのソフトウェア故障の影響判断に必要である。

疎結合のイベントシーケンスのリスクコントロール手段を特定するときには、考えられるあらゆるハザード状態を個々の疎結合イベントまで追跡しようとするのは概して生産的ではない。一般的には、疎結合のシーケンスの種類を特定し、次にシーケンスが万一起きた場合のリスクを最小化するための個々のイベントに対する検知メカニズム及びリスクコントロール手段（例：重大データのデータ破損を検知するためのチェックサム及び安全シャットダウン、又はそのようなイベントの使用者への通知）を特定するのがより良い方法である。

4.3.2.3 確率

ISO 14971:2007 の附属書 D では、「ランダム」故障と「決定論的原因」（系統的）故障を区別し、ソフトウェア故障は「決定論的原因」故障であると断定している。決定論的原因故障の確率の推定値は、非常に計算が難しい。ハードウェア故障には決定論的原因故障の性質を持つものもあるが（誤った定格のヒューズや不適切な原材料の使用など）、大半はそうではない（通常の消耗や損傷／寿命による故障など）。ソフトウェア不具合に対処するときには、一般にその確率を正確に推定することができず、許容できる推定方法についての合意がほとんど得られていない決定論的原因故障に排他的に対処することになる。これは、コンポーネントパーツの平均故障間隔（MTBF）の分析及び確率の配分に主眼を置くことが多いハードウェア不具合への対処とは対照的である。

ソフトウェア故障発生確率の定量的推定手法について、見解の一致は存在しない。ハザード状態の原因となるイベントシーケンス内にソフトウェアが存在する場合、ハザード状態に関するリスクの推定においてソフトウェア故障発生確率を考慮することはできない。このような場合、最悪のケースの確率を考えるのが適切であり、ソフトウェア故障発生確率は 1 とすることが望ましい。シーケンスの残留イベントの定量的推定が可能な場合は、イベントシーケンス全体の発生確率について定量値を推定できる。しかしながら、多くの場合これは不可能かもしれず、リスクを危害の性質だけに基づいて評価することが望ましい。そのような場合のリスクの推定は、ハザード状態から生じるハザードの重大性に焦点を合わせるのがよい。臨床医によって発見される可能性が高い故障と、発見されずに危害の原因になる可能性がより高い故障を見分けるため、臨床知識に基づいて主観的な確率ランキングを定めることもできる。

7 固有の要因は、その機能性が機器の臨床機能と明らかに関係があり、機器ハザードの原因の一つになっているソフトウェアの欠陥である。例として、試験結果計算アルゴリズムの欠陥が挙げられる。

【P.27】

決定論的原因（非ランダム）ソフトウェア故障の確率に主観的推定値を使用することは推奨できない。しかしながら、ソフトウェア故障の確率が定量的に推定できない一方で、リスクコントロール手段の多くはハザード状態を招くソフトウェア故障の可能性を完全には排除していない。中には、そのような故障がハザード状態の原因となる確率を単に許容水準まで引き下げるだけのものもある[8]。したがって確率は、代替的リスクコントロール手段を評価するとき又はリスクコントロール手段実施前／実施後リスク評価のときに、相対的な値として使用することができる。

リスク配分手法の中には、確率と重大性の積（又は他の方法では加重値）がリスクレーティング（評点）となるように確率を扱っているものもある。リスクコントロール手法を考慮する必要がないリスク評点以下に対して、切捨てポイントが選択されることもよくある。このようなスキームを使用する場合で、確率配分が不適切な場合には、リスクコントロールが不十分な結果に終わりかねない。これを防ぐには、特に決定論的原因ソフトウェア故障がランダムハードウェア故障と同じ分析結果に含まれる場合には、重大性優先の評点を設定できる。潜在的故障の重大性が指定レベルより高い場合には、確率及びそこから導き出されるリスクレーティングに関係なく、リスクコントロールを検討しなければならない。これは、許容可能なリスクコントロールの決定において、確率推定手法へ過度に依存することを防止するのに役立つ。一般に医療機器ソフトウェアの場合は、ソフトウェアリスクコントロール手段の考慮を促し（設計の早期段階で考慮すれば非常にシンプルで安価になりうるものもあるため）、またできるだけ安全なソフトウェアの設計への意識を最大化するために、重大性優先レーティングを非常に低く設定することが望ましい。

つまり、ソフトウェアのリスク推定では、起こりうるソフトウェア故障の確率を推定しようと試みるのではなく、ハザードの重大性及び万一故障が起きた場合の危害の相対的確率に主眼を置くことが望ましい。[9]

4.3.2.4 重大性

一般に、ソフトウェアに使用する重大性の尺度は、医療機器全体について選ばれた尺度にかかわらず、事前に決められている（IEC 62304:2006 箇条 4.3 参照）。通常この尺度は、機器にとって十分なものである場合は、ソフトウェアにとっても十分なものである。しかしながら、ソフトウェアの関係者が、リスクマネジメント及びソフトウェア開発プロセス全般を通してその臨床的観点及び対象母集団を承知しておくことは重要である

なお、ソフトウェア故障の最悪の場合の重大性を決定する際には、機器におけるソフトウェアの役割が、単に人が機器全体の意図する使用を見ただけで予想したものに比べてはるかに危険性が低いソフトウェアになる可能性があることに注意されたい。例えば、外科用レーザーは深刻な危害をもたらすおそれがあるかもしれないが、装置内のソフトウェアは実際にレーザー出力を制御していないかもしれないし、安全レベルを超えた出力を行うことができないようになっているかもしれない。この場合、イベントシーケンス内にソフトウェアを含むハザード状態は存在しない。

IEC 62304:2006 箇条 4.3 は、重大性を使用して安全性に関するソフトウェアの分類を行っている。3つのソフトウェア安全クラスは、次のように分類されている：

- クラス A: 傷害又は健康被害が生じる可能性がない
- クラス B: 重篤でない傷害が生じる可能性がある
- クラス C: 死亡又は重篤な傷害が生じる可能性がある

ソフトウェア安全クラスは、最初にソフトウェアを含むハザード状態の最も深刻な結果に基づき、ソフトウェアシステム全体に割当てられる。しかしながら、重大性がもっと低いハザード状態にだけ当該ソフトウェアアイテムが含まれることを製造業者が実証可能な場合は、そのソフトウェアアイテムの安全クラスを下げてもよい。クラス A は、ソフトウェアアイテムがいかなるハザードシナリオにも含まれていない場合に限り割当てられる。

8 例えば、メモリ破損の場合のチェックサム又は通信プロトコルの一部としてのチェックサムは、起こりうるあらゆる破損が検出されることを保証するものではない。むしろ、そのような破損の大多数を検出することで、そのリスクを許容水準まで低減しうるものである。

9 これは、同じ重大性を持つハザードを見分けるのに役立ち、それによって実際の危害の確率がより高いハザードにもっと注力できるようになる。

【P.28】

ソフトウェアシステム内で、より下位の重大性がソフトウェアアイテムに与えられていない場合（フラットな分類）は、もっと分類を進めるよう検討するのが望ましい。これは、ソフトウェアシステム内において、リスクコントロール手段の目的上若しくは臨床上の効果又は基本的性能に対して、より「重要」でもっと大きな注意を必要とするソフトウェアアイテムが存在するためである。

重大性は、その故障が重も深刻なハザードの潜在的原因となるなりソフトウェアアイテムに焦点を当てながら、正常状態、異常状態及びストレス状態という最も広い範囲下で最も厳格に行う試験計画を作成するための指針の一つとして、利用することもできる。
また、ソフトウェアに変更が加えられる場合の回帰試験の焦点を決定するガイドとしても使用できる。

ソフトウェア安全クラスの決定にさいしては、リスクマネジメントの発展性及び反復性を認識しておく必要があるという点に注意されたい。ハードウェアリスクコントロール手段を使用する場合（プロジェクトの開始時には即座に決定されないことがある）、ソフトウェアの重大性は低くなる可能性がある。というのも、もはやソフトウェア故障がハザード状態の原因にはならず、ソフトウェア安全クラスを下げることができるからである。これは、設計前／設計後リスク及びリスクコントロールを特定することによって説明できる。なお、たとえ説明可能な場合でも、ソフトウェアのリスクを低減するために使用するハードウェア設計及びリスクコントロール手段が、以降の設計段階で又は製品メンテナンスやコスト削減の一環として、誤って排除されないよう保証することが望ましい。

5 リスク評価

ISO 14971:2007 から抜粋

5 リスク評価

特定した各ハザード状態について、製造業者はリスクマネジメント計画で定める基準を使用してリスク軽減が必要か否かを判断する。リスク軽減が必要でない場合、6.2 から 6.6 までに示す要求事項はこのハザード状態に適用しない（つまり 6.7 まで進む）。このリスク評価の結果は、リスクマネジメントファイルに記録する。

注記 1：リスク許容性の決定に関する指針は、D.4 で示す。

注記 2：医療機器設計基準の一部として関係する規格を適用すると、リスクコントロール活動を構成し、6.3 から 6.6 に示す要求事項を満たす場合がある。

適合性は、リスクマネジメントファイルの検査によって確認する。

5.1.1 一般

ソフトウェアリスクコントロールの決定の前に、特定したハザードのそれぞれについて、リスクコントロール手段の必要性を評価する。これは軽減前段階である。このリスク評価から、リスクコントロールが必要なハザード及び不要なハザードを特定する [10]。計画したリスクコントロール手段が十分か否かを特定するのではない [11]。この時点で誤りを犯すと、ハザードの原因になるおそれのある領域でリスクコントロール手段が考慮されないという結果になる可能性があることを理解されたい。この段階は、品質保証スタッフやシステムエンジニアがソフトウェアの想定に積極的に取組む好機である。

10 ただし、設計の早期段階でリスクコントロール手段が特定されれば簡単で安価になる場合があるため、ハザードがいかに軽微なものであってもすべてのハザードについてリスクコントロール手段の特定を図ることが望ましいと考える人が多い。

11 なお、ある種のリスクはソフトウェアでは十分にコントロールできず、医療機器レベルの変更（例：ハードウェアに冗長性や動作制限機能を付加する）が必要なことがこの時点で明らかな場合がある。

【P.29】

リスク番号の割当てにリスクレーティングの仕組みを使用した場合には、リスク全体（ハードウェア故障の確率及び重大性）とリスクコントロールを必要とする重大性優先レーティングの組合せを記した表を作成するのが一般的である。

文書に記録したハザードについて、潜在的ソフトウェア原因を持つものと持たないものを特定すること。ソフトウェアのリスクマネジメントは、ソフトウェアが寄与因子になりえないハザードのリスクコントロール手段を講じる必要はなく、ハードウェアに対して何らかのリスクコントロール手段を実施する必要もない。ソフトウェアが寄与因子でないハザードはよくある。これらのハザードは、機器のリスクマネジメント活動全体の他の部分で対処する。その場合でも、ソフトウェアが原因となりえないハザードに関して、将来の参考のためにその根拠を文書に記録することは、ソフトウェアリスクコントロールの欠落につながるおそれのあるシステムレベルで加えられる将来の変更を防止するのに有効となりうる。ハザードの残りのサブセットは、ソフトウェアリスクコントロール手段を特定する出発点となる。

例えば、モーター制御ハードウェアの設計が本質的に安全だとすれば（例：自己制御式電源）、そのモーター制御に関与するソフトウェアはリスクコントロールを必要としないかもしれない。なぜなら、動作を適切に制御する若しくはスピード又はトルクを制限するソフトウェアが故障したとしても、ハザード状態にはならず機器の有効性に悪影響を及ぼすこともないからである。

この初期段階のリスク評価には、臨床使用、考えられる誤使用、及びシステムレベルの医療機器ハザードに関する知識が必要であり、ソフトウェアだけに重点的に取組んでいたのでは効果的に実施できない。臨床／使用の専門家による実際的に独立したレビューの活用を検討することが望ましい。また、リスク評価の程度は潜在的な危害の重大性に対応することが望ましい。

6 リスクコントロール

6.1 リスク軽減

ISO 14971:2007から抜粋
6 リスクコントロール

6.1 リスク軽減

リスク軽減が必要な場合、6.2から6.7までに記載するリスクコントロール活動を実施する。

6.1.1 一般

リスクを軽減するには、ハザード状態の発生確率を下げる、又はハザード状態から生じる危害の重大性を下げることが必要である。イベントシーケンスの発生確率を下げるには、各イベントの発生確率を下げる必要はなく、そのシーケンス全体の発生確率を下げればよい。イベントシーケンスのどのポイントでリスクコントロール手段を実施すべきかを検討することによって、最も有効で、最もシンプル又は最も費用のかからないリスクコントロール手段が特定できる。多くの場合、コスト効率の高いリスクコントロール手段をイベントシーケンスの最初又は最後で実装して、そのシーケンスを中断することができる。

図2に、ハザード状態の原因となるイベントシーケンスを図解する。この図では、それぞれの丸がイベントを表している。左上のイベントは、ハザード状態が複数の連続によって起こりうることを示している。連続の最後のイベントでハザード状態の防止に十分なリスクコントロール手段が取られれば、シーケンス内の個々のイベントは別々のリスクコントロールを必要としないかもしれない。例えば、シーケンス内の複数のイベントが、システムの他の部分の不良データから生じたエラー状態を表すものであるかもしれない。だが、シーケンスの最後でデータの完全性を保証できるとすれば、考えられる各中間エラー状態を保護的状況に置く実利的な理由はないといえる。

【P.30】

右下のイベントシーケンスは、異なるハザード状態の原因となるイベントシーケンスについて考えられる数多くのバリエーションの別の例を示している。この例では、複数のエラー状態が不具合状態を起こしている。ここでリスクコントロール手段は、イベントシーケンスの最初で不具合を捕捉しリスクを軽減する例外処理状態であるかもしれない。イベントシーケンスのどこでリスクコントロールを適用するかを検討することによって、最も有効かつシンプルで費用のかからないリスクコントロール手段の決定の際に相応のメリットがあるかもしれない。多くの場合、コスト効率の高いリスクコントロール手段を危険なイベントシーケンスの最初又は最後に実装して、そのシーケンスを中断することができる。

（ハードウェア又はシステムの中の）ソフトウェア外のリスクコントロール手段をイベントシーケンスの終わり付近に実装することによって確率又は重大性を下げることが可能かどうかを検討することは重要である。医療機器の設計変更によって、ソフトウェアの変更なしにイベントシーケンスの最後で重大性又は確率を下げられることがある。

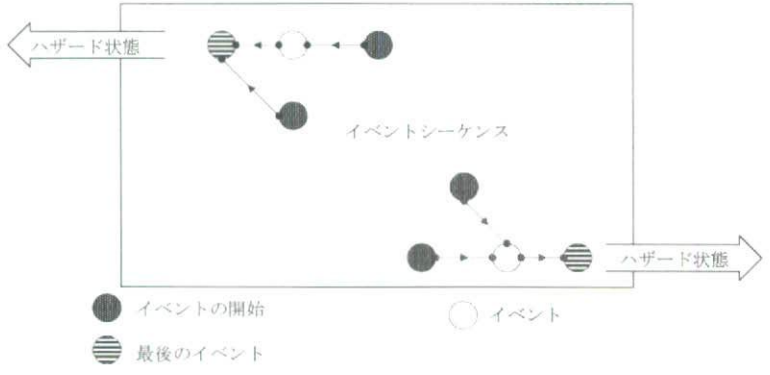

図 I – イベントシーケンスの最初のイベントと最後のイベント

ソフトウェアの最後のイベントに実装したリスクコントロール手段若しくはハードウェア又はラベリングを通してソフトウェア外に実装したリスクコントロール手段は、多くの場合リスクを最小化する上で重要である。これは、イベントシーケンスを通してハザード状態を招くおそれのあるソフトウェアの欠陥又は故障モードの一部が特定されておらず、十分な軽減もされていない場合はなおさらである。

非常に有効なリスクコントロール手段をイベントシーケンスの最後のイベントに定義できる場合には、考えられるすべての原因又はソフトウェアイベントを特定・軽減したことを実証しようとするよりも、ソフトウェアの安全性及び有効性を保護することの方がより簡単な場合がある。また、そのようなリスクコントロール手段は、考えられる寄与イベントごとに固有のリスクコントロール手段を実装するよりもコスト

効率も高い場合がある。さらに、イベントシーケンスの最後のイベントにリスクコントロール手段を適用することが、予測不能な影響を持ち、考えられるすべてのシーケンスを完全に分析することが実現不可能な場合の、疎結合の原因に対処するための最善の方策となる場合もある。

流体の圧力を制御する医療機器について考えたい。圧力が一定の限度を超えると、傷害が発生する可能性がある。ソフトウェアは、この限度を超える結果を招くおそれのあるソフトウェア又はハードウェアの考えられる不具合に対して、さまざまなエラーチェックやクロスチェックの機能を含め複雑な制御アルゴリズムを実装している。さらに、（場合によっては別の CPU カード上又はメモリ空間上にある）別のソフトウェアが、患者に最も近いポイントで圧力センサーを監視する。圧力が限度を超えると、ソフトウェアはポンプを閉める（ポンプを閉めることはハザード状態を表すものではないと想定している）。このような手段は、考えられるすべての原因及びイベントシーケンスが軽減されたか否かにかかわらず、ソフトウェア制御の最後のポイントで傷害のリスクに歯止めをかける。

【P31】

イベントシーケンスの最後のイベントに実装したリスクコントロール手段は非常に有効なものとなりうるが、常に絶対確実なものというわけではない。一般的には、イベントシーケンスの最後のイベントだけでなく、イベントシーケンスの中の一つ以上の寄与イベントに対してもリスクコントロール手段を実装することが最善である。イベントシーケンスの最後のイベントに対するリスクコントロール手段は、ハザードの潜在的原因の一部が見逃された場合、又は既存のリスクコントロール手段が失敗に終わった場合にリスクを軽減するセーフティネットとして機能する一方、その正確な実施が保証されていない場合でも、イベントシーケンスの中のその他のイベントに対するリスクコントロール手段の実施によって、そのリスクは軽減される。

イベントシーケンスの最後のイベントでのリスク軽減に加えて、最初のイベントに注意することも重要である。最初のイベントでリスクコントロール手段を実装すれば、ハザード状態を招く潜在的な原因の組合せの数を制限できる。イベントシーケンスの最初のイベントでのリスクコントロールは、結果としてハザード状態を招くおそれがある誤った情報の防止と特に関係がある。例えば、試験結果を判定する上で正確かつ適切に動作したアルゴリズムでも、センサーを通して取得したさまざまな生データに誤りがある場合や、使用者がパラメータやオプションを不適切に入力した場合は、その結果はやはり誤りである可能性がある。最初のイベントでのリスクコントロールの例としては、センサー情報に関する一貫性、不具合、又は範囲のチェックや、操作者が入力した情報に関するユーザーインターフェースエラーのチェックが挙げられる[12]。状況によって、リスクコントロール手段はイベントシーケンスの最後に実装した方が有効な場合もあれば、最初に実装した方が有効な場合もある。ときには、最初と最後、そして中間のイベントに、リスクコントロール手段の組合せを実装することが必要になる場合もある。適切なリスクコントロール手段は、機器の意図する使用、利用可能な技術、及びソフトウェア設計の制約によって異なる。

コントロールの最初や最後のポイントという概念が有効な一方で、通常はこれらのポイントでリスク分析やリスクコントロールを行うだけでは不十分である。これらのポイントでのリスクコントロール手段が十分であるように思えたとしても、その他のリスク手段もリスクをさらに軽減する上で、又は妥当性が完全でない場合を想定した「石橋をたたいて渡る」アプローチとして、価値があるかもしれない。

リスクコントロール手段を定めるときには、リスクコントロール手段によって複雑さが増す可能性があるということを認識しておくことが重要である。つまり、リスクコントロールの機能数が多いほど、多くの結果に欠陥が起こりやすくなる可能性がある。リスクコントロール手段を確立するときには、設計の複雑性の増大という結果について慎重に考慮することが望ましい。

6.1.2 リスクコントロールとしてのプロセス

イベントシーケンスにソフトウェアの欠陥から生じるイベントが含まれていると、特定のソフトウェアのリスクコントロール手段を実装できない場合があり、リスクコントロール手段をシーケンスの最後に適用することが現実的ではなくなる。この場合の最善のソリューションは、設計により本質的安全を確保してソフトウェアの欠陥がハザード状態を招かないようにすることである。これが不可能な場合は、効果

的なソフトウェア開発プロセスを使ってソフトウェア欠陥の発生確率を下げるようにしてもよい。ソフトウェア故障の唯一のリスクコントロール手段として試験を列挙するという好ましくない傾向が従来から見られる点を特に考慮して、このトピックについては多くの議論が交わされている。ただし、別のタイプのリスクコントロール手段との組合せを検討していてその詳細まで定義されているプロセス手段であれば有益であるという点では、強い合意が存在する。

意図する機能の確実な実行について、ソフトウェアの信頼性が非常に高いこと、そしてソフトウェアに不具合がないことについての信頼性が非常に高いことを立証できるなら、そのソフトウェアを完全性の高いコンポーネントとして扱うことができる。この高レベルの信頼性を達成するには、製造業者はソフトウェア開発プロセスが信頼性の高く不具合のないソフトウェアを予測通り生み出せることを実証しなければならない。そうすれば、そのようなプロセスを使用しているのでソフトウェア欠陥の発生確率を抑えられると主張できる。

12 考えられる操作者エラーのすべてが検出可能であると主張することを意図したものではない。

【P.32】

ソフトウェア開発プロセスの厳格度を上げることで、ソフトウェア欠陥の数を減らせるという主張は広く認められている。ただし、医療機器とソフトウェアの試験がソフトウェア欠陥を防止するわけではないことに注意されたい。医療機器とソフトウェアの試験はむしろ、ハザード状態を招くおそれのある欠陥の検出のために行う。ソフトウェアは非常に複雑で徹底的に試験することができないため、厳格な試験はハザード状態の確率を小さくする手法として見なすことができるに過ぎない。しかしながら、試験だけでは、ソフトウェアが完全性の高いコンポーネントして取扱うのに足る信頼性を持っていると立証することはできない。

厳格なソフトウェア開発プロセスを定める上での出発点は、IEC 62304:2006 で定める活動と作業をプロセスに含めることだと言えるだろう。自分の団体のために厳格なソフトウェア開発プロセスを開発する際に考慮すべきその他の事項としては、次が挙げられる：

- スタッフの能力－技能、資格、経験、及び教育（誰がソフトウェアを開発するのか？）
- 手法－仕様、設計、コーディング、及び試験方法の適切性（どのような開発プロセスか？）
- 厳格度、形式性、及びレビューと検査の範囲（静的分析をどの程度実行するか？）
- ツール－コンパイラやコンフィギュレーション管理ツールなどのツールの質（ソフトウェア開発中にどのようなツールを使用するか？）

完全性の高いソフトウェアの開発についての予測可能性は、一貫して実施できる反復可能なプロセスが存在するか否かによって異なる。

完全性の高いソフトウェアを作り出すのに十分に厳格と考えられるプロセスを使用する場合でも、その結果を検証しなければならない。このプロセスで生み出されたソフトウェアが、高い信頼性を備え不具合のないものであることを実証する必要がある。この実証ではおそらく、ソフトウェアの使用に関するデータや、ソフトウェア開発プロセスが完了した後に発見される異常の収集が必要となる。完全性の高いソフトウェアであるなら、繰返し使用した後でも異常はほとんどないはずである。

ISO 14971:2007 から抜粋

6.2 リスクコントロールオプション分析

製造業者は、リスクを許容可能な水準まで軽減するのに適切なリスクコントロール手段を特定する。

製造業者は、次のリスクコントロールオプションの一つ以上を、次に掲げる優先順位で使用する：

a) 本質安全設計

b) 医療機器自体又は製造工程の保護手段

c) 安全性情報

注記 1：オプション b) 又は c) の実施について、製造業者はリスクが許容できるか否かを判断する前に、合理的に実施可能なリスクコントロール手段が考慮され適切なリスクの軽減を提供するオプションが選択されているプロセスに従うことができる。

注記 2：リスクコントロール手段は、危害の重大性を下げること、危害の発生確率を下げること、又はその両方が可能である。

注記 3：多くの規格が、医療機器の本質安全設計、保護手段及び安全性情報について定めている。また、他の多くの医療機器規格が、リスクマネジメントプロセスを統合した要素を備えている（例：電磁両立性、ユーザビリティ、生体適合性）。関連規格は、リスクコントロールオプション分析の一部として適用することが望ましい。

注記 4：危害の発生確率を推定できないリスクについては、D.3.2.3 を参照。

【P.33】

注記 5：附属書 J で提供する安全性情報に関する指針

選択したリスクコントロール手段は、リスクマネジメントファイルに記録する。

リスクコントロールオプション分析中に、要求されるリスク軽減の実施が不可能と製造業者が判断した場合、製造業者は残留リスクのリスク／効用分析を行う（6.5 へ進む）。

適合性は、リスクマネジメントファイルの検査によって確認する。

6.2.1 一般

ハザードやハザード状態の原因となるイベントシーケンスを特定すると、次にリスクコントロール手段を特定することができる。ここで、医療機器のリスクを軽減するリスクコントロール手段の特定を簡単にするという、リスク分析の真価が明らかになる。

本質安全設計とは、設計上の決定がハザード状態の発生確率をゼロにする又は小さくするようになされたことを意味する。保護手段は、潜在的なハザード状態を検出し、それらが危害を招くことを防止する、又は危害に至る確率を小さくする設計上の決定である。安全性情報は、ラベル表示、ユーザーマニュアル、オンスクリーンディスプレイ、又は教育を通して使用者に伝達される情報である。これは、使用者による機器との正しい対話及び機器が表示する臨床データの正しい使用により、ハザード状態を回避するのに役立つ。安全性情報の例には、視聴可能なエラーフラグや重大なアクションについての使用者への確認要求などがある。

ハードウェアに実装されたリスクコントロール手段よりもソフトウェアに実装されたものを使用すべきか否かの決定は、多くの場合費用対効果の問題である。ソフトウェアは複雑なため、ハードウェアに実装されたリスクコントロール手段の検証の方が容易な場合もある。しかし、ハードウェアの再設計や製造工程の変更に要するコストは、ソフトウェアにリスクコントロール手段を実装するコストよりはるかに高くなる可能性がある。リスクコントロール手段のソフトウェアへの実装を性急に決めると、望ましい水準のリスク軽減が得られない場合もあるため、この両面のバランスを徹底的に調査することが重要である。

リスクコントロール手段のハードウェアへの実装に要するコストは、開発が進むにつれて高くなる。リスクマネジメントをどこに実装するかの決定が設計のより早い段階で行われるほど、有効性がその決定の主要検討事項になる可能性が高くなる。このため、システムレベルのリスク分析及びリスクコントロールオプション分析は、製品開発プロセスのできる限り早い段階で行うことが望ましい。

リスクコントロール手段は、危害の潜在的原因の重大性を小さくすること、及び／又はハザード状態の発生確率を小さくして、許容可能なリスク水準を満足することが望ましい。追加的なコントロール手段は、

それぞれが設計エラーの可能性を高め、新たなハザードを招く又は新たなハザード状態を生み出す可能性がある。したがって、リスクコントロール手段はできる限りシンプルでストレートなものにし、常に新しいリスク評価にかけることが望ましい。

6.2.1.1 ソフトウェア欠陥のためのリスクコントロール手段

製造業者は、リスクコントロールオプションの違いを理解し、それぞれの場合で可能な限り最善のタイプのリスクコントロール手段を適用するよう努力することが重要である。これを効果的に行うために、ソフトウェアリスクマネジメントを機器開発プロセス及びソフトウェア開発ライフサイクルの早い段階で実施しなければならない。

次の例は、ソフトウェア欠陥を含むイベントシーケンスからハザード状態が生じる場合について、ISO 14971:2007 で定められている 3 つのリスクコントロールオプションのそれぞれを説明するものである。2 つの例を下に示すとともに、表でそれぞれのリスクコントロールオプションに対するリスクコントロール手段の例を掲げる。

【P.34】

最初の例では、ソフトウェアが機器カバーを閉じるモーターを制御している。潜在的なハザードの一つとして、患者の指がカバーで挟まれ、その結果患者の指に物理的な危害が及ぶことが考えられる。

二番目の例では、ソフトウェアが放射線治療の被ばく量を制御している。ハザードは過剰照射であり、その結果患者に危害が及ぶ。過剰照射の潜在的原因は、動的メモリ割当ての失敗として特定されている。

表1－ ソフトウェアイベントのためのリスクコントロール手段の例

	本質安全設計	保護手段	安全性情報
例1	機器のカバーを閉じたときに最大トルクになっても患者の指にケガを負わせないようなモーターサイズにする。	モーターの出力を監視し、指定のリミットを超えた場合にモーターを逆回転させる又は止めるようなセンサー入力及びソフトウェアアルゴリズム。	「動作中に触れないこと」と書いたラベルをカバーに貼る。
例2	動的メモリ管理の使用自体を止め、静的データ構造だけを使用する。	動的メモリ割当て失敗のエラー検出及び放射線出力終了のためのソフトウェア。	各治療セッションの前に機器の再起動を行わせる指示。

ハザード状態を招くの原因となりうるソフトウェアの考えられる欠陥又はコードすべてについて、個別のリスクコントロール手段を特定するのは不可能なことも多く、その場合は上位の抽象的レベルで作業し、欠陥のタイプを検討して最も重大なコンポーネントに最大の注意を払うことが必要になる。

ソフトウェアのリスクマネジメントについては、次の細分箇条で述べるリスクコントロール手段の分類を検討することが役立つこともある。各カテゴリーについて検討することによって、特定したコントロール手段の包括性が確保され、安全性にかかわるソフトウェアコンポーネントの適切な特定を確実にする。

6.2.1.2 ハードウェア故障のためのソフトウェアリスクコントロール手段

このカテゴリーには、対策が取られなければ危害を招くおそれのあるハードウェア故障が起きた場合に安全対応又はシャットダウンを確実にするための、ハードウェア故障を監視又はこれに対応する（保護手段）ソフトウェアが含まれる。

例としては、針の位置を監視してハードウェアセンサー故障を検出した場合に動作を停止する又は針を退避させるソフトウェアが挙げられる。

6.2.1.3 ユーザーエラーのためのソフトウェアリスクコントロール手段

このカテゴリーには、妥当性確認のためにユーザー入力をチェックし、無効又は疑わしい入力についてユーザーエラーメッセージを表示する又は追加確認を求める（保護手段）ソフトウェアが含まれる。

例としては、有効期限日又は患者の年齢についての単純なデータ入力エラーチェックや、使用者がテストストリップを適切に挿入したかどうかチェックするソフトウェアが挙げられる。

6.2.1.4 医療機器相互接続のためのソフトウェアリスクコントロール手段

このカテゴリーには、通信ネットワークを介して送信中の医療機器データの一貫性及び正確性をチェックする（保護手段）ソフトウェアが含まれる。

例としては、インポートを信頼できるデータ元のみからとする制限や、データ転送の成功を検証するチェックサムの使用が挙げられる。

【P35】

6.2.2 ソフトウェア開発中のリスク軽減

ソフトウェアに対して適切なリスクコントロール手段を効率よく実装するには、製品開発及びソフトウェアライフサイクルに関する理解について、慎重に検討しなければならない。リスクコントロール手段の種類によっては、設計の初期段階であれば非常に簡単に実装できるが、開発の後の段階になると実装が不可能になるものもある。製品開発プロセスの早期段階でソフトウェアをリスクマネジメントの観点から慎重に検討しないと、医療機器の安全性について、適切なソフトウェア操作への依存が期せずして過度になってしまうようなハードウェアの決定がなされる可能性がある。

極めて重大なソフトウェアアイテム（例：欠陥があれば死に至るおそれのあるアイテム）と安全性に影響を及ぼすおそれがないソフトウェアアイテムを区別するために、ソフトウェアアイテムを分離してソフトウェア安全クラスを割当てることは、有効な手法となりうる。ソフトウェア安全クラスについては、IEC 62304:2006 細分箇条 4.3 を参照すること。ソフトウェア安全クラスの割り当ては、より重大なソフトウェアアイテムの検証及びコンフィギュレーション管理活動で、厳格度をより上げ焦点をさらに絞るための基礎として役立つ。これを行う場合は、悪影響を注意深く検討する必要があり、重大性の低いソフトウェアアイテムでも、それがより重大性の高いソフトウェアアイテムに影響を及ぼすおそれがある場合はそれと同じレーティングを与えなければならない。

なお、最初にソフトウェアアイテムを安全性に関係するアイテムに分類し、一定のリスクコントロール手段や設計選択を行った後により低い重要性にして扱うこともできる。リスクマネジメントを正しく実施すれば、安全性に関係するソフトウェアは分離と本質安全設計により最小限のサブセットに減らすことができる。

ソフトウェアの安全性を確保するには、製品開発ライフサイクルを通してさまざまな活動が必要になる。故障分析の正式手法などの信頼性に関する技術には、完全なリスク分析手法は含んでいない。また、信頼性と安全性は関連していることが多いものの、まったく同じ属性ではないことを理解しておくことも重要である。信頼性に重点を置いたライフサイクルプロセスでは、安全性を十分に達成できない場合がある。

6.2.2.1 ソフトウェアアーキテクチャの設計

ソフトウェアアーキテクチャを使用して、本質安全設計によりリスクをコントロールできる。また、ソフトウェアアーキテクチャは、リスクを低減する保護手段のメカニズムを提供することもできる。

6.2.2.2 アーキテクチャの機能による本質安全設計

ソフトウェア制御機能にかかわるハザードは、例えばその機能の実装にハードウェアを使用するなどの手段によって回避できるかもしれない（同様に、ハードウェア機能にかかわるハザード（消耗、疲弊）は、ソフトウェアの使用によって回避できるかもしれない）。

時には、上位レベルでの設計に関する決定によって、ハザードを完全に回避できる場合もある。例えば、ハードウェアの観点からは、AC 電源の代わりにバッテリーを使用することによって感電リスクを排除できる。また、ハザードを招くおそれのあるプログラミングエラーのクラス全体を、上位レベルでの設計によって排除可能な場合もある。例えばメモリリークは、静的データ構造だけを使用することによって回避できる。

6.2.2.3 フォールトトレラント（耐故障）アーキテクチャ

医療機器によっては、基本性能を提供できないと患者や使用者にハザードが及ぶものもある。フォールトトレラントアーキテクチャによって、機器の故障にかかわるリスクを制限して機器の安全性を高められる場合がある。

フォールトトレラント設計は、機器の信頼性を高めるアプローチとして非常に一般的である（ソフトウェアエンジニアリング実施担当者のための参考資料としては、Pullum[7] や Banatre[8]などがある）。このアプローチは、設計に何らかの形の冗長性を採用することによって医療機器故障の確率を小さくしそれによってリスクを最小限にするために、本質安全設計及び保護手段を取り入れている。フォールトトレラント設計の狙いは、コンポーネントに欠陥がある状況においても、安全関連システムが確実に動作し続けるようにすることである。

冗長性を実現するための手法は数多くある。例えば、一般的な故障モードの確率を最小限に抑えるため、冗長要素を交換可能にする又は設計に多様性を採り入れる場合がある（例：重要センサーの入力をクロスチェックするためのアルゴリズムとコードを単独で開発する）。他の検討事項としては、冗長要素が機器の正常動作に積極的に関与するのか、あるいは必要とされるまで待機するのか、という点がある。冗長性スキームの中には、多数決又は同様の組合せ演算を採用して冗長要素からの出力を選択するものもあれば、冗長性要素を待機状態に置き、主要素が故障した場合にだけ介入又は作動するようにする医療機器アーキテクチャもある。

【P.36】

6.2.2.4 保護手段

多くの場合、本質安全設計ですべてのハザードを回避すること及びすべての潜在的故障に対してフォールトトレラント設計を実装することは現実的ではない。それらのケースでは保護手段が、潜在的ハザードに対処するための次善のアプローチとなる。一般に保護手段は、潜在的なハザード状態を検出することによって作動し、結果を軽減するために自動介入するか、又は警報を発生させて使用者に介入を促す。

例えば治療用X線システムは、施設のドアが開けられた場合にX線発生装置をシャットダウンするソフトウェアロジック又はハードウェアを使用したインターロックシステムを備えている場合がある。そのインターロック機能は、治療上の役割を一切担っていない。その唯一の目的は、意図しない放射線被ばくのハザードを軽減することにある。

時には（つまり、機器の機能性の喪失がハザードを招かない場合）、役割を転嫁にして安全性が達成される場合がある。例えばラボ用血液分析装置の場合、結果を出力できなくてもハザードは招かないが、誤った結果を出力すればハザードにつながりかねない場合がある。この例の場合、防御的プログラミングチェックで予期せぬ不具合が示されたときには、分析装置の動作を続けるのではなくシャットダウンすることでリスクを軽減する。フェイルセーフアーキテクチャでは、システム又はコンポーネントの不具合若しくはその他のハザード状態は、機能の喪失につながる可能性はあるが操作者及び患者の安全は守られる。フェイルオペレーションシステムでは、システムは安全に動作を継続できるが性能は低下する（例：キャパシティが下がる、応答時間が遅くなるなど）。これらの手法は、複雑な機器では機能単位で適用することもでき、これは適切な低均化（graceful degradation）の概念につながる。

保護手段の中には、その保護機能の実施に実行時間を要するものがある（例：ランタイムチェック、ビルトインテスト、警告）。組込みシステムによく見られるビルトインテストの例としては、ROM CRC（チェックサム試験）、重要変数の冗長格納及びクロスチェック、スタックチェック、プログラムチェックなどが挙げられる。他の保護手段はアーキテクチャの属性と見なすことができ、パーティショニング、カプセル化、静的データの格納、未使用 ROM を既知の値で満たすなどがある。これらの手段は、その保護機能の実施に実行時間を必要としない。論理グルーピングやコントロール（ボタン）のカラーコーディングなど、多くのユーザーインターフェース設計手法も、保護手段の構成要素となっている。

6.2.2.5 疎結合原因によるリスクの軽減のためのパーティショニング（分割）

疎結合原因による故障から生じるイベントの影響を評価することが非常に難しいので（疎結合原因の例については、附属書 C を参照）、ソフトウェアのパーティショニングを利用して安全性が重要なソフトウェアをこれらのイベントから隔離することができる（IEC 62304:2006 箇条 4.3 参照）。これによって、安全性に関係しないソフトウェアのソフトウェア安全クラスを下げ、安全性に関係するパーティションに対するリスク軽減に注意を集中させることができる。

この手法を有効にするためには、パーティショニングによって安全性に関係するソフトウェアアイテムを安全性に関係しないソフトウェアアイテムから十分に隔離し、安全性に関係しないソフトウェアアイテムがいかなる形でも安全性に関係するソフトウェアアイテムの動作に干渉できないようにしなければならない。また、パーティショニングにより、ソフトウェアアイテムによるリソース（物理的又は時間）の適切な使用を実証するのが望ましい。

物理的リソースというのは、実質的には、ソフトウェア担当者のハードウェアのモデルである。バスなどの機能は、それを制御するレジスタと見なされる。物理的リソースは、プロセッサレジスタ、メモリ割当て、I/O、及びその他の特殊なレジスタで構成される。

時間などのリソースは、割込み、システムタイマー／カウンターの使用、同期アクションなどの関連するハードウェアアクションを有する不連続のタイミングイベントである。

【P.37】

ラボ環境でのシステム試験では所与のテストケースについて十分な性能を示すものの、現場でアプリケーションをロードする又は実行環境（同じボックスで他のプロセスが実行されている）にすると、ソフトウェア（又はその未達性能）が危害を招くような挙動を示す場合がある。

一方、ラボにおけるテストケースで、性能は低いがソフトウェアのスピードアップを性急に図るために妥当でない手段が取られているということが示された場合、その手段が設計を台無しにし目に見えない悪影響による他の新たなリスクを生む可能性がある。

効果的なパーティショニングは、正常動作環境で次のことを実証することが望ましい：

1) データフロー破損が防止されている：安全性に関係しないソフトウェアアイテムは、安全関連データを変更できない。

2) コントロールフロー破損が防止されている：
   - 安全関連機能は、安全性に関係しないソフトウェアアイテムのアクションに影響されることなく、常に適時実行可能である。
   - 安全性に関係しないソフトウェアアイテムは、安全性に関係するソフトウェアアイテムを変更できない。

3) 実行環境の破損が防止されている：
   安全性に関係するソフトウェアアイテム及び安全性に関係しないソフトウェアアイテムの両方が使用するソフトウェアシステムの部分（例：プロセッサレジスタ、デバイスレジスタ、メモリアクセス特権）の破損は起こりえない。

上記のいずれかに反するイベント（例：ハードウェア故障）が検出され、安全性の継続の確保に必要な措置がシステムによって実施されるようにすることが望ましい。

6.2.2.6 開発経路が未知のソフトウェア（SOUP）に関する考慮事項

システム設計中に、SOUP コンポーネントの使用が決定される場合がある。医療機器の潜在的ハザードが高いほど、SOUP コンポーネントの潜在的故障モードをより慎重に検査し、リスクコントロール手段を特定しなければならない。一般に、SOUP コンポーネントは、リスクコントロール手段を組込むために変更を加えることができない。また、SOUP コンポーネントが招くすべての潜在的ハザードを特定するために十分な内部設計情報を入手することもできない。したがって、SOUP コンポーネントの故障によるハザードの発生を防ぐために、SOUP コンポーネントを監視又は隔離するために必要なリスクコントロール手段が提供されるよう、システム及びソフトウェアのアーキテクチャを設計しなければならない。

機器に SOUP コンポーネントが含まれている場合、機器の安全性が損なわれないように注意を払わなければならない。そのような状況では、「ラッパー」、つまりミドルウェアアーキテクチャが必要となる場合がある。このミドルウェアは、a) 使用を望まない SOUP の機能の使用を阻止する、b) SOUP と機器ソフトウェアの間で正しい情報が転送されるようにするための論理チェックを実行する、又は c) 機器が必要とする追加情報を提供する、ことができる。

SOUP に関連するその他の問題に、市販のオペレーティングシステムや通信システムの使用がある。徹底したリスクアセスメントに基づき、安全性を損なうことなくソフトウェアプラットフォームへの変更（例：安定性、セキュリティ）を許可するアーキテクチャを構築することが望ましい。このようなリスクアセスメントには、ネットワークセキュリティパッチのインストールなど、機器安全性の完全性を確保するために必要な変更の頻度分析を含めることが望ましい。

6.3 リスクコントロール手段の実装

ISO 14971:2007 から抜粋

6.3 リスクコントロール手段の実装

製造業者は 6.2 で選択したリスクコントロール手段を実装すること。各リスクコントロール手段の実装を検証すること。検証結果をリスクマネジメントファイルに記録すること。