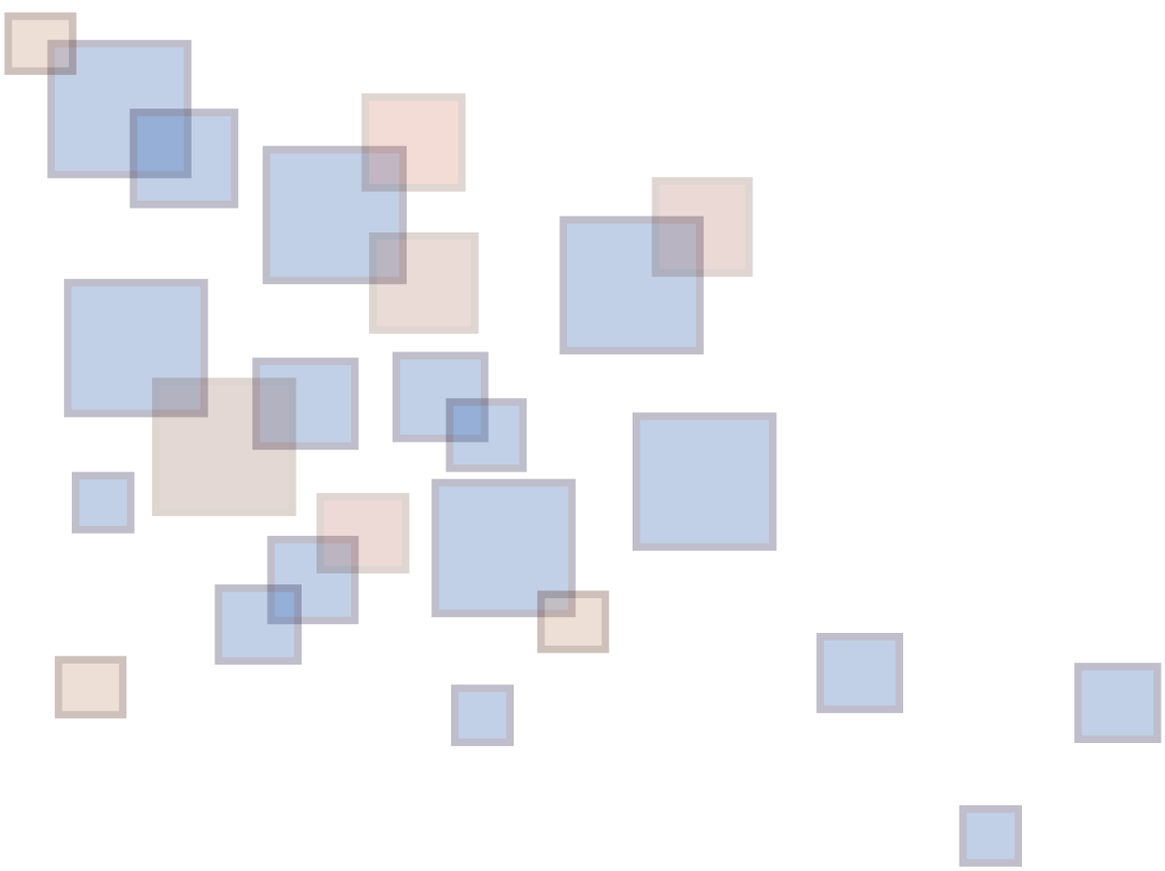
ITプロジェクトのリスク予防への実践的アプローチ

―ユーザー／ベンダー協働によるリスクへの対処―



IPA 独立行政法人 情報処理推進機構
技術本部 ソフトウェア・エンジニアリング・センター 編

まえがき

■ IT プロジェクトとリスク

情報システムは企業経営を支える重要な基盤である。企業が価値を生み出し、その価値を商品やサービスを通して顧客に届けることで顧客との関係性を構築し、そして収益を回収するといった企業活動は、今や情報システムなしには実現できない。情報システムは、企業の根幹を支える重要な存在である。

企業活動を支える情報システムは、ユーザー企業の手によって構築されることもあるが、多くの場合はベンダー企業の協力を得て構築される。ここに、ユーザー企業とベンダー企業が協働する IT プロジェクトが生まれる。そして、ユーザー企業とベンダー企業の担当者が互いの専門性を発揮しながら、経営目標の達成に必要な情報システムを具現化するために IT プロジェクトを進めていく。

しかし、IT プロジェクトの実施にあたっては、様々なリスクがある。リスクが顕在化したときに、その対処を誤ると損失が生じ、また、その損失の扱いをめぐってユーザー企業とベンダー企業ので「対立」が生じる。そして最後には、企業間の力関係のために一方的に押し切られるなどして、片方のみが損失を被るという後味の悪い決着をみることもある。または、転嫁したはずのリスクが自社の事業に損失となって現れたりする。読者が携わった IT プロジェクトでも、そのような心当たりがないだろうか。

■ 本ガイドの範囲

本ガイドでは、リスクの原因と結果の構造に着目し、リスクを顕在化させないための予防的措置に焦点を当てている。リスクの予防的措置のためには、IT プロジェクトにおいて「どのようなリスクがあるのか」をユーザー企業とベンダー企業が一緒に議論し、重点的に管理すべきリスクへの共通認識を持った上で、これを誘発する要因（ドライバー）に対して双方が働きかける必要がある。ドライバーに働きかけるためには、まず、その有無を客観的に把握する必要がある。そして、ドライバーを取り去る、またはリスク事象に至らしめないための対策を講じる。さらに、その対策が有効に機能しているかどうかを把握し、リスクを管理下に置く。この一連のプロセスにおいて定量指標（メトリクス）を適用し、客観的な把握に努める。本ガイドでは、このような範囲を扱っている。

なお、一般的なリスクマネジメントではリスクが顕在化した後の対処や、リスクの発生確率の扱いを含むが、本ガイドでは対象外としている。

また、本ガイドではユーザー企業とベンダー企業の協働を阻むリスクの例として、要件定義で発生するものに焦点を当てている。これは、ユーザー企業とベンダー企業の協働における問題の多くが、これらのプロセスに集約されているためである。

■ 本ガイドについて

本ガイドは、IPA/SEC 内に設置された定量的管理基盤 WG での議論の成果をまとめたものである。本 WG では、プロジェクト管理や品質保証などの実務経験が豊富なエキスパートを、ユーザー企業とベンダー企業の双方から集め、IT プロジェクトの現場で実際に生じている問題とその予防策について議論を重ねてきた。議論では、ユーザー企業とベンダー企業の双方の「言い分」を共有し、問題が発生するメカニズムを検討した上で、それぞれの企業で実践され成果を上げている予防策等について検討してきた。

本ガイドが、ユーザー企業とベンダー企業の協働における「対立」の未然防止に少しでも貢献できればと考えている。リスクの予防的措置に取組むことで、IT プロジェクト本来の目標達成に必要なタスクに、ユーザー企業と

ベンダー企業の双方が集中できるようになり、それが両企業それぞれの競争力の向上、また、企業の健全な成長と産業の発展につながるものと信じている。

■ 本ガイドの構成

本ガイドは、次の3部構成となっている。

・第1部（解説編）
・第2部（資料編）
・付録

第1部（解説編）では、IT プロジェクトのリスクについての一般的な事項を説明し、ユーザー企業とベンダー企業の協働における IT プロジェクトのリスクを把握することの難しさを述べる。その上で、本ガイドで用いるリスク事象予防モデルを概観し、リスク事象とその原因となるリスク事象ドライバーという因果関係の構造を示す。また、リスク事象ドライバーを客観的に把握するためにメトリクスを用いること、及びリスク事象ドライバーの予防策の有効性確認のためにメトリクスを用いることを示す。その上で、リスク事象予防モデルを適用するプロセスについて詳しく説明する。

第2部（資料編）では、本 WG の委員が IT プロジェクトの実務においてしばしば遭遇したリスク事象や、予防策の適用により上手く回避できたリスク事象について、事例を重ねて経験的に得られた知見を整理し、リスク事象ドライバー記述書の形式で整理したものを掲載する。これらの中には、「このような方法を適用すればリスク事象を回避できるだろう」という希望的観測に基づく予防策も含まれるが、多くは委員所属の企業において実践されてきた有効な予防策である。

モデルを理解するためには、資料編からまず入って具体的なイメージを持ってもらうとよい。

また、付録では、本ガイドで用いるリスク事象予防モデルの基となる標準リスクモデルの概説を掲載する。その他、本ガイドの第1部と第2部には含めていないが、WG の中で検討した事項のうち本ガイドに記載しておくことが望ましいと思われる事項を掲載する。

まえがき

第1部(解説編)

第2部（資料編）

付録

第1部（解説編）

1. IT プロジェクトのリスクへの対処

1.1 リスクとは

IT プロジェクトのリスクについて、PMBOK では「もしそれが起きれば、プロジェクトの目標にプラスやマイナスの影響を与える不確実な事象」と定義している（PMI, 2009）。リスクは、その性質として、①まだ起こっていないが、起こるかもしれない不確実な事象、②発生確率を伴う、③プラスとマイナスの影響が生じる、という3つの要素を持つ。これらの要素をより明確に表現したリスクの定義として、「発生確率を持ち、最終的に損失または利益になること」がある（岡村, 2008）。

1.2 リスク事象

一般的なリスクマネジメントでは、1.1 節で述べたとおり、リスクの発生によってもたらされるプラスとマイナスの影響と、リスクの発生確率を扱う。しかし、多くのITプロジェクトにおいてリスクマネジメントの主な対象となるのは、マイナスの影響を持つリスクである。また、リスクの発生を回避するためには、リスクの発生確率について精緻な議論を深めるよりも、そもそもリスクの原因を取り除くことに注力することが求められる。そのため、冒頭の「はじめに」で述べたとおり、本ガイドではマイナスの影響をもたらすリスクのみに着目している。また、リスクの原因を客観的に把握することに焦点を当て、リスクの発生確率の扱いについては対象外としている。

このように、本ガイドで扱うリスクの範囲は、一般的なリスクマネジメントにおけるリスクに比べて限定したものである。そのため、限定された範囲であることを明確化するときには「リスク事象」という用語を用いている。リスク事象という用語については、スミス＆メリットの著書において示されている「プロジェクトにおいて発生する可能性のある、望ましくない出来事」（スミス＆メリット, 2003）という定義に従い、本ガイドでも用いることとする。

1.3 リスク事象把握の難しさ

ITプロジェクトにかかるリスク事象を適切に把握することは容易ではない。その理由として、少なくとも次の3つがあると考えている。

第一に、リスク事象をステークホルダー間で共有する難しさがある。例えば、顧客や上司に対して「あなたが要件をコロコロ変えることが、重要なリスクの要因です」、もしくは「あなたが交渉に弱腰であることで、要求が曖昧なままプロジェクトが進行して危機に瀕しています」などと、常に直言できるとは限らない。

第二に、リスク事象の評価基準を定めることの難しさがある。ステークホルダーによって、リスク事象に対する感度や識別スキルに大きな差がある。そのため、リスク事象を把握するときに誰の意見が正しいか判断するのは容易ではなく、誰かがリスク事象であるように感じていても、「それがリスクです」と明言するには根拠が必要である。

第三に、ステークホルダーの立場によってリスク事象の捉え方が異なる点が挙げられる。ユーザー企業、またはベンダー企業のいずれかの立場から、自社の事業への影響範囲としてリスク事象を捉えてしまいがちになると、双方の協働による「IT プロジェクト」全体のリスク事象に目を向けることが容易ではなくなるため、企業の枠を超えて、全体的な視点でリスク事象を捉える必要がある。

これらの理由が互いに絡み合うと、リスク事象の把握はさらに難しくなる。結果として、リスク事象に関する情報の共有が遅れ、対応のタイミングを失い、大きな影響に繋がるということもしばしば起こると考えられる。

1.4　リスク事象の予防に向けて

1.3 節で挙げた点を踏まえて、リスク事象の予防を実現するために本ガイドでは次のアプローチをとる。

① リスクを原因と結果の構造で捉え、原因に相当する「リスク事象ドライバー」に着目する

② 原因であるリスク事象ドライバーに対して予防策を講じ、回避と軽減を図る

③ 定量指標（メトリクス）を用いることで、リスク事象ドライバーの有無を客観的に把握するとともに、リスク事象ドライバーの予防策の有効性を確認する

④ リスク事象、リスク事象ドライバー、予防策、メトリクスの具体的な例を提示し、資料編としてまとめ、IT プロジェクトリスク予防方法のイメージを読者に分かりやすく示す

2. リスク事象の予防モデル

2.1 モデルの全体像

本ガイドでは、図 2-1 に示したリスク事象予防モデルを参照モデルとして用いている。これは、スミス＆メリットの著書『実践・リスクマネジメント』(スミス＆メリット, 2008)に示された標準リスクモデルを参考にしている。標準リスクモデルの概要は付録 A に掲載した。

リスク事象予防モデルでは、IT プロジェクトにマイナスの影響をもたらすリスク事象と、その原因となるリスク事象ドライバーを、それぞれ異なる要素として区別している。これにより、リスク事象という結果と、その原因であるリスク事象ドライバーとの間にある因果関係を識別することを狙いとしている。そして、原因に相当するリスク事象ドライバーに対して、回避と軽減の二種類の予防策を考える。さらに、リスク事象ドライバーを客観的に把握するため、及びリスク事象ドライバーの予防策の有効性確認のためにメトリクスを用いている。

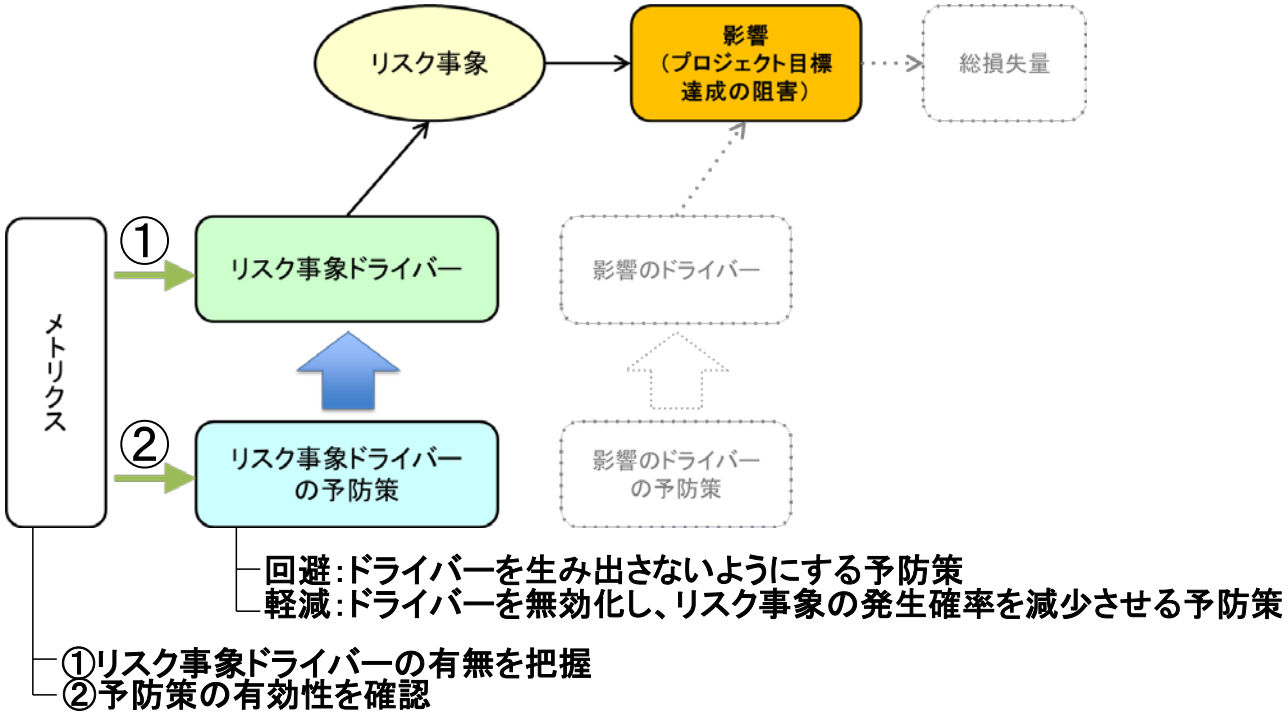


図 2-1 リスク事象予防モデル

2.2 モデル要素の説明

以下に、リスク事象予防モデルに含まれる主な要素について、それぞれ説明する。

2.2.1 リスク事象

プロジェクトにおいて発生する可能性のある、望ましくない出来事である。リスク事象が発生し、これに適切な対処をとれないまま時間が経過してしまうと、品質・コスト・納期などに関する目標を達成できないなどの影響が生じ、損失となって現れる。

2.2.2 リスク事象ドライバー

プロジェクトにおいて存在する、特定のリスク事象を誘発しうる要因である。例えば、「要件獲得の対象とすべきユーザー側の対象者が正しく特定できていない」というリスク事象ドライバーが要件定義プロセスにおいて存在した場合、「要件定義に漏れが生じる」というリスク事象が発生しうる。

プロジェクトにおいてリスク事象ドライバーが存在しているからといって、そのドライバーと因果関係にあるリスク事象が必ず発生するわけではない。

2.2.3 リスク事象ドライバーの予防策

リスク事象ドライバーの予防策には、少なくとも、回避と軽減の二種類が求められる。

回避策は、プロジェクトにおいて関心のあるリスク事象について、これをもたらすリスク事象ドライバーを存在させないようにするための措置である。これは、リスク事象とリスク事象ドライバーという因果関係において、リスク事象ドライバーを発生させないための予防措置である。

軽減策は、プロジェクトにおいて関心のあるリスク事象について、これをもたらすリスク事象ドライバーの存在が確認された場合に、そのドライバーを無効化し、リスク事象の発生確率を減少させるための措置である。これは、リスク事象とリスク事象ドライバーという因果関係において、リスク事象ドライバーが発生した場合にリスク事象が発現しないための予防措置である。

2.2.4 メトリクス

リスク事象ドライバーの有無を客観的に把握したり、リスク事象ドライバーの予防策の有効性を確認したりするために、本ガイドではメトリクスを用いている。メトリクスというと、0, 1, 2, 3, …などの量的変数で表されるものが一般的だと思われるが、質的変数で表されるものもメトリクスに含まれる。具体的には、支援体制の有無を「あり／なし」で回答するような名義尺度に基づくメトリクスや、要員のスキルを 5 段階で評価するような順序尺度に基づくメトリクスなども含まれる。もちろん、プロジェクトの要員数やレビュー実施回数などを数値で測定するような比率尺度に基づくメトリクスも利用できる。

本ガイドの資料編に具体例として提示したメトリクスは、そのほとんどが名義尺度または比率尺度に基づくメトリクスである。

2.3　リスク事象予防モデルの記述

図 2-1 に示したリスク事象予防モデルに基づいて識別された具体的な内容は、図 2-2 に示したリスク事象ドライバー記述書のテンプレートに従って記述する。本ガイドでは、リスク事象ドライバーを定量的に把握し、予防策を講じることに重点を置いているため、テンプレートではリスク事象ドライバーを基軸としている。

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
このリスク事象ドライバーによって顕在化するリスク事象	
リスク事象ドライバー有無の把握	
把握の観点	メトリクス（定性的／定量的）
リスク事象ドライバーの予防策	
対応区分	予防策の内容	予防策の有効性を確認するメトリクス

図 2-2 リスク事象ドライバー記述書のテンプレート

3. リスク事象予防モデルの使い方

3.1 リスク事象予防モデルの適用プロセス

リスク事象予防モデルを実際に使用するにあたっては、図3－1に示す適用プロセスに基づくことを想定している。適用プロセスは、リスク事象の記述、リスクの特定・予防策の検討、評価・維持の3つのアクティビティから構成される。

当該プロセスを適用することにより、リスクの可視化、リスクの共有、リスクに関連するメトリクス活用等の推進が期待される。

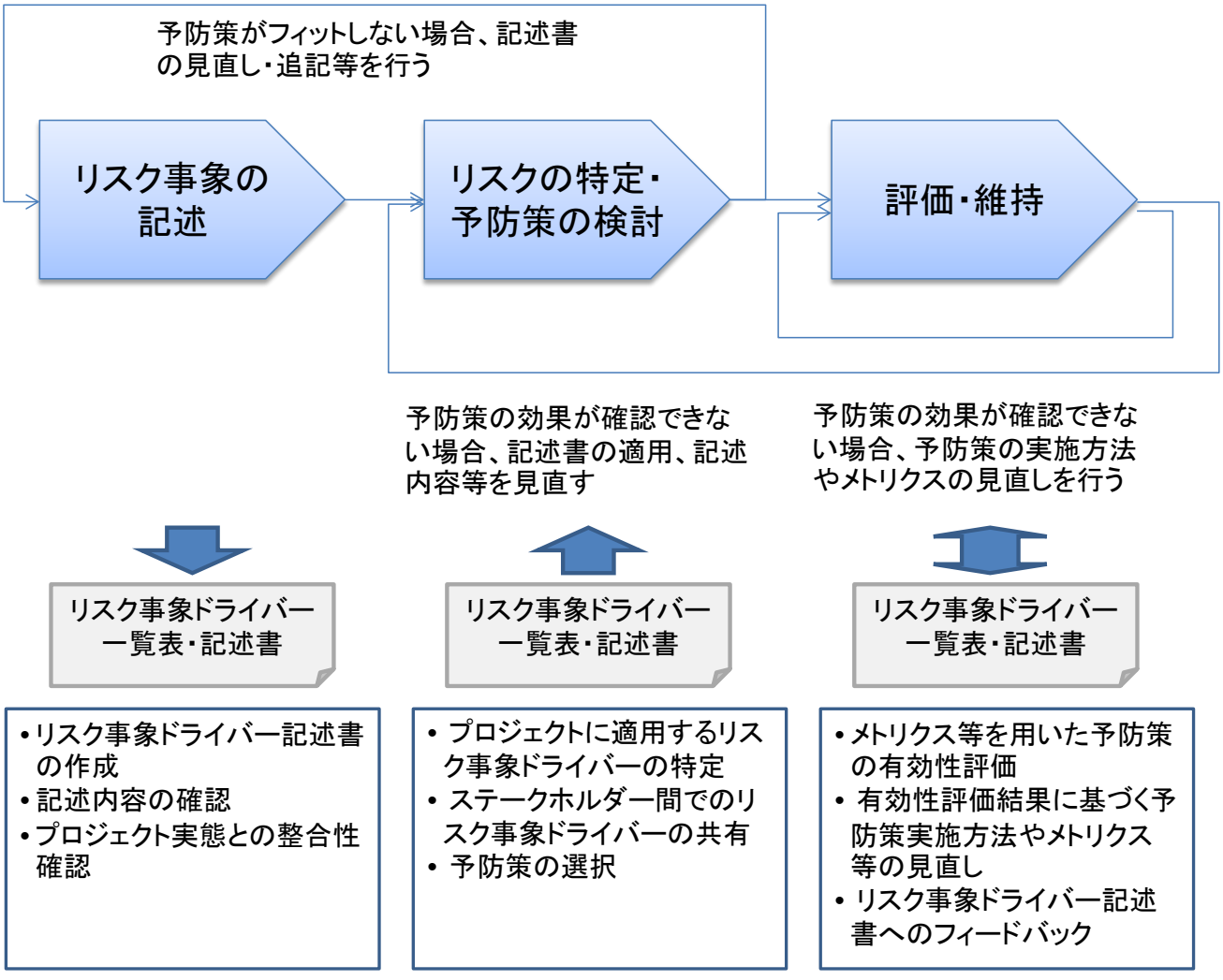


図 3-1 リスク事象予防モデルの適用プロセス

以下に適用プロセスを構成する3つのアクティビティの概要を示す。

① リスク事象の記述

このゴールは、テンプレートに基づきリスク事象ドライバー記述書を作成し、リスクを可視化・精査することである。

資料編に掲載した事例等を参考に、プロジェクトの目的、特性、環境等に基づき、検討が必要と目されるリスク事象についてリスク事象ドライバーを記述しとりまとめを行う。記述した内容については、プロジェクト

で検討すべきリスクの実態と整合をとることが重要であるため、アクティビティの中で記述書を基にした十分な議論を行い、その結果を反映して記述書の改変と追加を行う。

② リスクの特定・予防策の検討

このゴールは、リスク事象ドライバーをプロジェクトの現状に当てはめて、現実的なリスクを特定し、そのリスクに合った予防策を選択することである。

具体的には、プロジェクト内のリスク事象ドライバーを把握・特定し、プロジェクト内、及び関連するステークホルダー間でリスクに関する情報の共有を行った上で、記述書に示される予防策を基に適用する予防策を検討する。

③ 評価・維持

このゴールは、予防策の実施状況を把握し、メトリクス等を用いて有効性を評価して必要な改善策を検討することである。

これらの検討を通じて、プロジェクトのリスク低減、信頼性向上に貢献するとともに、評価の過程で得られる諸々の気付きをリスク事象ドライバー記述書に反映することにより、当該プロジェクト、及び今後のプロジェクト活動におけるリスク事象の全体像が明らかになり、予防策が有効に機能することにも繋がると期待される。

本プロセスは、アクティビティを順次遂行し予防策を実施することによりプロジェクトのリスクの低減を図るものである。しかしながら、評価の結果としてリスクの低減が明確にならない状況であっても、それぞれのアクティビティを実施しリスク事象ドライバーや予防策の確認、選択、見直し、評価等を繰り返すことにより、プロジェクトメンバーやステークホルダー間におけるリスクについての認識は進むと想定され、プロジェクト全体としてのリスク対応力は向上していることが期待できる。

次節より、各アクティビティの進め方を解説する。

3.2 リスク事象の記述

リスク事象ドライバー記述書のテンプレートに沿って、リスクを記述する。記述書を作成するにあたってのアプローチは、リスク事象ドライバー一覧表（5章参照）から、プロジェクトの特性等により注力するカテゴリーを決定し、各カテゴリーの要求に従ってリスク事象ドライバー記述書を作成していく方法、プロジェクトメンバーがそれぞれの危機意識に基づき自発的にリスク事象ドライバー記述書を作成し、それらをプロジェクト全体で議論しながら不足する領域を埋めていく方法等が考えられる。

アプローチ1（トップダウンアプローチ）:

リスク事象ドライバー一覧表を用いて、プロジェクトの目的、特性、環境等の諸条件から重点的に検討すべきリスクを挙げ、そこから注力するカテゴリーを決める。

例えば、リスク事象ドライバー一覧表では、企業システムの開発等の事例で大きなトラブルになりがちなリスクについて、WG の参加者の意見を基に、プロジェクトのかなり早期の段階で気付くべきリスク事象ドライバーの例を、作業プロセスのカテゴリー別に記述している。プロジェクト早期にこのようなリスク事象ドライバーに気付くこと

ができれば、ステークホルダーに大きな影響を与えずに手を打つことが可能なはずである。このような事例を参考として、プロジェクトで検討すべきリスクのカテゴリーを定義し、重点カテゴリーごとにリスク事象ドライバーを記述していくというアプローチが考えられる。なお、カテゴリーについては、作業プロセスに加え、必要に応じて工程等のタイミングやドライバーを発見すべきロール等を組み合わせて定義することが考えられる。

次に、一覧表に挙げたそれぞれのリスク事象ドライバーについて、リスクの重大性や対応の計画をイメージしながら、リスク事象ドライバー、リスク事象、予防策、メトリクスなどを記述していく。

アプローチ2（ボトムアップアプローチ）:

プロジェクトメンバーが、それぞれの問題意識や懸念事項に基づいて、ランダムにリスク事象ドライバー記述書を作成する。この時点では、重複や強弱の調整を過度には行わず、メンバーの意識するリスクが全て表出化されることを目標に記述書の作成収集を進める。

作成した記述書を基に、リスク事象ドライバー一覧表を作成し、これを基にプロジェクトメンバー全員でプロジェクトのリスクについて議論し、優先順位の高いリスク事象ドライバーが何であるかの合意を形成していく。ここで重複したドライバーが挙がっていることは、予防優先度の高いリスクを表している可能性があることに留意する。

ある程度の俯瞰的整理ができた段階で、優先的に予防等の対応を検討するリスク事象ドライバーを抽出する。抽出したリスク事象ドライバーについて、予防策実施に向けた担当をアサインし、リスク事象ドライバー、リスク事象、予防策、メトリクスなどの記述を実施に向けて精査する。

アプローチ1は、プロジェクトのリスクに関するアプローチが体系的になることを狙っており、リスクを俯瞰した一覧表を効率的に作成する上では有効と考えられる。対して、アプローチ2は、プロジェクトメンバーの実感や問題意識に基づいたリスク事象ドライバーの抽出が可能になり、アプローチ1に比べてとりまとめに時間は要するものの、より現実的なリスクを把握できる可能性がある。アプローチ2を行うのであれば、とりまとめを進める上で強力なファシリテーターを用意し、統制のとれたプロセス運営を行うことが重要になると想定される。

いずれのアプローチをとるにしても、リスクに関するプロジェクト内での認識の共有に至ることがポイントとなり、そのためにはリスク事象ドライバー記述書の記載事項のレベルを合わせることが重要になると想定される。例えば、リスク事象ドライバーの記述に関する粒度等に着目し、メンバーの誰かが違和感を持つようなドライバーに関しては、より共通的・汎用的な原因に変える、またはより現実的な原因を挙げるなど、十分に議論して全体のレベル感をすり合わせていくことが求められる。

3.3　リスクの特定・予防策の検討

ほぼ完成したリスク事象ドライバー一覧表とリスク事象ドライバー記述書を活用して、プロジェクトのリスク現状を把握しリスクを特定していく。リスクの特定のための活動には、リスクのカテゴリーやプロジェクトの工程ごとに様々な形態がある。例えば、要件定義工程の開始前にキックオフ的にプロジェクト関係者を集めて一覧表をベースに今後顕在化の想定されるリスクを特定してみることや、要件定義工程の終了前に、グランドレビューの前後の作業として残存リスクを特定してみることなどが考えられる。

前述したように、リスクは様々なステークホルダーの立場や捉え方によって様相を変えるものであり、議論を通じてそれらの立場の違いを認識し距離を詰めることは常に重要である。その意味で、リスク事象ドライバー一

覧表やリスク事象ドライバー記述書もこれで完成とは考えず、リスクに関する議論の過程で更なる見直しが発生すると考え続けることが重要であり、リスク対応力の点からは健全であると言える。

リスクの特定では、一覧表、記述書、並びに現状を比較してリスク事象ドライバーを決定することについて主に議論が行われると想定しているが、当然のように予防策の実効性にも議論が及び、同時に精査が進むことを期待している。

予防策の選択にあたっては、記述書の対応区分（回避・軽減）に注意を払う必要がある。回避策は、それを十分に行った場合、リスクの根本的な原因に作用することを意識しており、大きな変革を生むが効果が大きい対応をイメージしている。他方、軽減策では、現状に配慮しながら実施する比較的穏やかな対応をイメージしている。通常の対応では、対応の効果を見ながらこれらを併用することになると想定している。

テンプレートの情報から有効な予防策が見出せない場合は、現場担当者の意見等を参考に有効と思われる予防策の提案を受け、それらを加えた予防策の検討を行う。検討結果は、後発のプロジェクトへの適用も視野に入れ適宜テンプレートに反映する。

それぞれのリスク事象ドライバーについて十分な議論を行い、精査された予防策の中からプロジェクトとして実施する対応を選択する。なお、選択した対応とその根拠になった記述書の内容については、予防策を実施する担当への伝達を確実にすることが望ましい。これにより、実施担当による予防策実施の評価に向けて、リスクを捉える観点や有効性を評価するメトリクスについての認識が高まることが期待される。

3.4　評価・維持

予防策の実施を受けてしかるべきタイミングで、対応の効果＝リスクの低減状況について評価を行う。一義的には予防策の決定時に記述書を基に定めたメトリクスを基に評価を行うが、その他にリスクの低減を見極める有効な情報・方法があれば、それらについても評価のインプットに加えることが望ましい。例えば、経験に基づく知見については、定性的な情報であっても参考にする価値があると思われる。評価には、原則としてリスク事象の記述に関わったメンバーを始め、なるべく多くのステークホルダーが参加することが望ましい。直接的関与が難しい場合には、評価の議事内容に関するコメントを収集するような方法でも関与を引き出すことが、プロジェクト全体の意識を合わせる上で重要である。

評価の運営では、それぞれのリスクについて評価を行い、関係者の総意として十分にリスクが低減したという合意に達したものは評価の対象から外していく。また、プロジェクトの総合的な状況から優先順位が下がったリスク事象ドライバーについても、関係者の合意が得られれば評価対象から外していく。その他のリスク事象ドライバーについてはリスクのレベルに応じて評価を継続することとし、これ以降の評価予定を定めることになる。また、従来前提と捉えていた事象（プロジェクトの目的、特性、環境等）が変化した場合には、新たなリスクに対してリスク事象の記述のアクティビティを部分的にでも再実行することが必要になる場合も想定される。

基本的にはリスクが全面的に消滅することはないと考えた方がよく、プロジェクトの存続中は評価を繰り返すような運営を継続するのがよい。

リスクそのものが変動しやすいことを意識すれば、プロジェクトの工程の節目や一定の期間をもって、リスク事象ドライバーに関する一覧表・記述書の妥当性を確保するため、それぞれの記述項目について総合的な見直しを実施することが望ましい。こうした活動を通じて、有効なリスク事象ドライバーに関する一覧表・記述書が整備されていくことに加え、リスクに関する企業、組織、プロジェクト及び関連するメンバーやステークホルダーの理解の向上が期待される。

参考文献:

プレストン・G・スミス、ガイ・M・メリット（澤田美樹子訳）『実践・リスクマネジメント』生産性出版（2003）
『ISO/IEC 12207：2008　ソフトウェアライフサイクルプロセス（JISX0160:2012）』
独立行政法人情報処理推進機構ソフトウェア・エンジニアリング・センター『共通フレーム2013』
独立行政法人情報処理推進機構ソフトウェア・エンジニアリング・センター『IT プロジェクトの「見える化」』
PMI『プロジェクトマネジメント知識体系ガイド（PMBOK®ガイド）』第4版、PMI（2009）
岡村正司『IT プロジェクトを成功に導く リスクマネジメント大全』日経 BP 社（2008）

第 2 部（資料編）

4. 本ガイドのリスク事象ドライバーについて

第2部は WG 委員がそれぞれの職場で顕在化する可能性のあるリスク事象とリスク事象ドライバーについてリスク事象ドライバー記述書を用いて、整理した結果を資料編としてまとめている。

それぞれの立場（ユーザー、ベンダー、プロジェクトマネージャー、情報システム部門、品質管理部門など）で重要なもの、頻出するものを経験に基づく言葉で記述している。

整理にあたって、重視したことは次の点である。

✓ **要件定義に着目**

要件定義（要件定義プロセスと開発プロセスのシステム要件定義、ソフトウェア要件定義まで）はユーザー、ベンダーの協調作業において接点が多いプロセスなので特にここに重点をおいて検討した。

注）プロセス等の名称は、共通フレームより参照。

✓ **経験上、頻出した事象を選定**

検討するにあたって、各委員がユーザー、ベンダー間に関する事柄で、頻繁に発生し、困ったことに関して事例を挙げてもらった。

✓ **プロセスに着目した区分**

様々な経験を収集したときに、分かりやすい軸としてプロセスに着目して区分した。

- プロジェクト主プロセスで対処すべきリスク事象ドライバー

  要件定義・設計・実装などのメインプロセスにおいて、対処すべきリスク事象ドライバーである。これらはプロジェクト固有の品質に関わる。ベンダー、ユーザーでは分けていない。

- プロジェクト支援・管理プロセスで対処すべきリスク事象ドライバー

  プロジェクトの主プロセスが円滑に実行されるように対処すべきリスク事象ドライバーである。これらはプロジェクトに依らずに実施すべきことで、複数のプロジェクトの品質に関わる。

- 組織的に対処すべきリスク事象ドライバー

  プロジェクト単位ではなく、組織として対処すべきリスク事象ドライバーである。これらは組織全体の品質に関わる。

✓ **リスク事象ドライバー名の統一**

リスク事象ドライバー名の記述ルールを下記のように定めた。

（誰が）	何について	～していない、～できていない

5. リスク事象ドライバー一覧

表5－1に本ガイドで扱う 24 個のリスク事象ドライバーの一覧を示す。

それぞれの詳細説明は6章に示す。

表 5-1 本ガイドで扱うリスク事象ドライバー一覧

区分	NO.	リスク事象ドライバー	ページ
主プロセス	1	システム化の目的が明確でない	16
	2	現行機能の調査・確認が不足している	19
	3	現行システムとそのドキュメントが整合していない	22
	4	パッケージ選定に関する検討が十分でない	25
	5	性能の検討が十分でない	28
	6	可用性の検討が十分でない	30
	7	運用要件の検討が十分でない	32
	8	運用に向けての制約条件が明確でない	34
	9	要件を獲得すべきステークホルダーが網羅されていない	37
	10	システム部門による要件とりまとめが十分でない	40
支援・管理プロセス	11	ドキュメントの更新が管理されていない	42
	12	仕様の変更管理ができていない	44
	13	ユーザーによる仕様の確認が十分でない	47
	14	要求の優先度が曖昧になっている	50
	15	業務要件の網羅性が検証できていない	52
	16	設計と実業務の整合性が検証できていない	55
組織的プロセス	17	経営層によるプロジェクト運営の関与が十分でない	58
	18	経営層によるスコープ決定への関与が十分でない	61
	19	経営層がパッケージ導入の意図・目的を明示していない	64
	20	ステークホルダー間の力関係がアンバランスである	66
	21	高次の調整・決定機関が機能していない	68
	22	十分なコミュニケーションがとれていない	71
	23	業務用語が共有されていない	73
	24	業務知識が不足している	75

6　個別リスク事象ドライバー記述書

6.1　システム化の目的が明確でない

表 6-1 リスク事象ドライバー(1)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
システム化の目的が明確でない	プロジェクト主プロセス
このリスク事象ドライバーによって顕在化するリスク事象	
要件の増大、もしくは絞り込みの不全が発生し、プロジェクトが統制できなくなる。	
リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的／定量的)
システム化目的が文書化されているか	・されている／いない
各目的に対する責任部門が明確になっているか	・明確になっている／いない
各目的の達成指標が定義されているか	・KPI が定義されている／定性的な定義はある／定義されていない
要求や仕様と目的との紐付けが分かる仕組みはあるか	・仕組みがある／ない
財務的目標、もしくはKPI目標、定性的な目標といった各システム化目的に対して、それぞれ対応する達成指標が、要件定義書や基本計画書に記載されているか	・記述率 → 100%か
システム化目的の優先順位は決定しているか	・決定している／いない
各プロジェクト目標に対する責任部門が決定しているか	・決定している／いない
各プロジェクト目標のカットオーバー後のトレース方法は決定しているか	・決定している／いない
上記は全ての組織階層(意思決定者～PM)で共有されているか	・組織階層の責任者数＝承認者数 ・共有したことを確認した人数／共有すべき人数＝100%
リスク事象ドライバーの予防策	
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	企画プロセスの結果に基づき、システム化の目的を具体的なレベルで文書化し、複数の目的がある場合は優先度を明確にする	・システム化の目的がプロジェクト企画書へ記載されている／いない
回避	各システム化目的の責任部署を明記	・プロジェクト企画書が適切なレベルで承認されている／いない ・目的・目標のトレース責任部署の設定率 ⇒ 100%か
軽減	仕様が膨らんだ場合(コストや納期の超過が見込まれる)のエスカレーションルールと判断基準を定める	・プロジェクト計画書にエスカレーションルールと判断基準が記載されている／いない

軽減	コンフリクト発生時の調整機関を設置し、システム化目的とプロジェクト目標のトレース責任部署全体の調整を図る	・設置されている／いない

＜リスク事象ドライバー＞

企画プロセスの結果が反映されていない、あるいは企画そのものが曖昧である。

例えば、システム化の目的がトッププライオリティ以外にも複数存在し、企画プロセスで優先順位が整理しきれていないような場合、ステークホルダーによって、もしくはエンドユーザー部門とシステム部門間で優先事項や達成レベルの認識が異なっているケースが存在する。

＜リスク事象＞

本来のシステム化の目的に沿わない要件が混在し、要件定義フェーズで要件を絞り込む段階でも、システム化目的が共有されていないため、本来の目的と合致しない判断が行われるなど要件のコントロールができなくなる。

＜リスク事象ドライバーの有無の把握＞

下記が満たされない場合は、要求間の整合性、プライオリティの確認がされていない可能性があり、ドライバーが存在しているおそれがある。

- システム化の目的について、達成目標が具体的に明記されている。
- 複数列挙されている目的に対して、それぞれ、どのようにシステム的に実現するのか、どの目的をどのレベルで実現するのかについて、要件定義（もしくは基本計画書）に明記され、それが関係者間で共有されている。
- 各目的に対する達成責任者が決定している。

＜リスク事象ドライバーの予防策＞

まずシステム化目的、プロジェクト目標を網羅的かつ具体的に文書に記述する。

例えば、「システム老朽化に対応した再構築」というのは目的かも知れないが、これでは再構築に伴ってどこまで現場の要求を受け入れ、機能改善するのかが明確になっていない。明確な幾つかの目的を具体的に掲げ、それを最優先にすること、期間や予算を超過する場合は、その目的以外の要求を削減対象にすることが判断基準として周知されることが望ましい。

例えば、「トランザクションが増加傾向にある〇〇業務について、現在より 10%以上作業時間を短縮する。その達成については〇〇部門が責任を担う」というように、この目的の達成に寄与する仕様が優先されることを明確にする。

また、ステークホルダーが多く、コンフリクト発生時の調整が困難であると識別できる場合は、目標達成の責任部署を巻き込んだ意思決定のため、ユーザーサイドの PM レベルからの調整機関設置の提言も必要である。

＜解説＞

多くの場合、システム化の目的は稟議書・決裁文書などの意思決定を示す文書に明記されており、基本的に不明確であることはあり得ないが、それが開発現場で提示されていない、あるいはプライオリティがはっきりしないケースでは、プロジェクトメンバーからは目的が不明確であるように見えることもありうる。

予算も期間も制限があるプロジェクトにおいては、システムの仕様を目的に沿って絞る必要があるが、肝心の達成すべき目的が不明確であったり、ステークホルダー間で共有されていなかったりするケースが少なくない。

例えば、事務系のシステム改善で、事務ミスのリスクを減らしつつ効率化による人員減も考慮する、また、営業系のシステムでは、顧客サービスを向上させながら、収益の獲得も見込む、といったケースである。さらにシステム基盤の強化による可用性の強化なども盛り込まれているケースもある。

高次の実施判断の時点（経営判断、部門間折衝など）では、何らかの要素（リスク回避、効果など）が決め手となって実施判断がなされるケースでも、プライオリティが文書化されないことは多いと考えられる。この場合、開発現場に降りてきた際には複数のシステム化目的が併存したまま、要件としてまとめられる。

それら全てがプロジェクトの予算と納期に納まらない場合の優先順を決めておかないと、要件のコントロール不全に陥る可能性がある。

さらに、現場には、IT 化によって業務を効率化したい要望や、現行システムへの不満などが溢れている。目的が曖昧だと、我先にと要望が出され収拾がつかなくなり、同様に要件のコントロール不全に陥り、プロジェクトが統制できなくなる。

この状態では要件の絞り込み時に、システム化目的の優先順位を踏まえない判断が起案部門担当者、情報システム部門担当者、ベンダーPM の間でなされるケースも多い。

このように、システム化の本来の目的が共有化されないことで、プライオリティや実装方針が迷走したり、目的からかい離したりするケースがある。

現場の PM はユーザー、ベンダー側双方とも、また、エンドユーザー側担当者も当初の経緯や判断過程を知り得ないケースも多く、要件の絞り込みや詳細の決定で、当初の目的を充足しない決定がなされる可能性もある。

この結果として、例えばシステムのリスク回避の部分の機能が低下した形で実装されていて、後日リスクインシデントが発生する、もしくは監査にて指摘されたりする、などの問題に発展することもあり得る。

これを防止するためには、システム化目的と優先順位を、財務目標、KPI 目標などの形で明確化した上で、カットオーバー後のトラッキング責任を誰が負うかといったユーザー側の体制整備も必要である。

これらがなされていない場合には PM 等の現場レベルから、明文化や調整機関の設置を求めることも別途必要である。

6.2　現行機能の調査確認が不足している

表 6-2 リスク事象ドライバー(2)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
現行機能の調査・確認が不足している	プロジェクト主プロセス

このリスク事象ドライバーによって顕在化するリスク事象
受け入れテスト時に不具合指摘が多発しプロジェクトの中断または大量の仕様変更が発生してコストが超過

リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的／定量的)
ベンダーに業務知識がない	・同一業務分野での開発・保守の経験がある／ない
ユーザーに現行システム開発時に要件定義作業の携わったメンバーがいない ※業務要求と現行業務仕様の関係を理解し、説明できるメンバー	・メンバーがいる／いない
現行システムの設計書がメンテナンスされていない	・最新化されている／いない
要件定義書レビューへのユーザー参画率が低い	・ユーザーのレビュー密度(ページ／時間) ・エラー指摘密度(指摘件数／ページ)
ユーザー辞書は作成したか	・作成した／しない
要件定義書の記述レベルは適切か	・要件定義書からテストシナリオの抽出が可能／不可能
外部設計書のウォークスルーをユーザーと参加のもとで実施したか	・実施した／しない
「現行どおり」という記載はないか	・存在する／しない
システムの開発目的は明確になっているか	・明確かつ具体的に記述されている／いない
継承すべき機能と新規に追加する機能が明確に識別されているか	・識別されている／いない

リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
軽減	業務に精通した人を事前にアサインする	・精通した人の参加の有無
軽減	要件定義書(主に業務プロセスフロー図)を基に、業務とシステムの関係を再レビューする	・ユーザーのレビュー密度(ページ／時間) ・エラー指摘密度(指摘件数／ページ)
軽減	現行の設計書がメンテナンスされているのかを確認するフェーズを設ける	・確認フェーズの有無
軽減	業務要件の提示等、ユーザーが責任を持って行うべき作業について、責任分担・役割分担を明確にする	・責任分担・役割分担が明確に行われているか否か

軽減	ユーザーのレビュー参加について事前に定める	・ユーザーのレビュー参加工数割合（%） ・レビュー密度（ページ／時間） ・エラー指摘密度（指摘件数／ページ）
軽減	ユーザー辞書を作成する	・ユーザー辞書の有無
軽減	要件定義書が適切な記述レベルまで詳細化されているか	・要件定義書からテストシナリオが作成できるか否か
軽減	「現行どおり」という記載を調査する	・現行機能の調査フェーズの有無 ・現行システムのドキュメントが最新状態にない場合は、現行システムのソースの確認作業の有無
軽減	システムの開発目的は明確になっているか	・プロジェクト企画への記載の有無（プロジェクト企画段階で明確になっているか）
軽減	現行から継承する機能、現行に修正を加える機能、新規に追加する機能を明確にする	・明確に識別されているのか

＜リスク事象ドライバー＞

プロジェクトメンバーに現行業務仕様の知識がない、現行システムの設計書がメンテナンスされていない等のため、現行機能の調査・確認が十分行えない。

＜リスク事象＞

現行機能の調査・確認の不足から、受け入れテスト時に不具合指摘が多発し、プロジェクトの中断または大量の仕様変更が発生してコストが超過となる可能性がある。

＜リスク事象ドライバー有無の把握＞

＜リスク事象ドライバーの予防策＞

現行機能が継承されていないというリスク事象ドライバーの予防策の立案では、プロジェクトメンバーに現行業務仕様の知識があるか、またはそれを知る方法があるのかにかかっている。現行業務仕様を知る方法として現行システムのドキュメントの存在があるが、これが確実にメンテナンスされているかどうかの確認が必要になる。

メンテナンスされていなければ、現行システムのソースを読み解くことが必要になる。また、プログラムから業務仕様を確認する作業は大変な労力がかかるので、現行機能から継承する業務の重要性や複雑度、手戻りが発生した場合の影響を考慮し、ソースを読み解く部分を選択することも必要になる。この場合、ソース解読による確認を実施しない部分については、確認を行った部分に比べ継承漏れによる仕様変更が発生する可能性が高くなる。したがって、未確認部分から発生する仕様変更については受容したリスクとしてプロジェクト計画やリスク計画に反映し管理することも必要になる。

現行業務仕様の確認では、ユーザーの関与は必須となることから開発を着手する前にユーザー側の体制・責任分担・役割を明確にすることが重要となる。

＜解説＞

業務システムを再開発するときに、現行システムのドキュメントが存在しない、存在したとしても正確にメンテナンスされていないということがよくある。また、ユーザーは日々の業務は分かっているが、業務流れとシステムの処理の紐付けができていないケースが多い。この場合、ユーザーは現行のシステムを意識せず要件を定義し、要件定義書には「現行どおり」と記載しその後の設計書の確認を省略しがちで、受け入れテスト時に仕様の解釈の相違などによる不具合指摘が発生する。ベンダーには、仕様変更に近い要求がリリース直前に大量に突きつけられ、突貫工事をするはめになる。結果として、プロジェクトのコスト超過、リリースの延期、最悪の場合にはプロジェクトが中断する事態に発展することもある。

現行システムとソースコードがあれば100%現行機能が継承されるというわけではない。ソースコードはシステムの要件や業務の流れ、業務データやデータに対する処理などが分かりやすく表現されてはいないので、これらを伝えるには現行システムにある業務であっても、改めて要件定義書へ記載しなければならない。さらに、複雑な業務については、業務フローやユースケース、画面・帳票定義など様々な方法で可視化する必要がある。また、ソース解読による確認は、あくまで継承する業務が作成された要件定義書に反映されているかどうかを確認するための作業であることを理解すべきである。

特に気を付けなければならないのは、業務システムの再開発の際に、業務改革を併せて行う場合の体制である。新システムの設計思想が現行システムと異なることから、現行システムの設計思想が引き継がれることを嫌い、新システムのプロジェクトメンバーに現行システムの開発・維持管理のメンバーがアサインされていないケースがある。こうなると現行機能の継承は諦めなくてはならない。

6.3　現行システムとそのドキュメントが整合していない

表 6-3 リスク事象ドライバー(3)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
現行システムとそのドキュメントが整合していない	プロジェクト主プロセス

このリスク事象ドライバーによって顕在化するリスク事象
現行システムから引き継ぐべき要件に漏れが発生する

リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的／定量的)
現行システム概説書の不存在	・存在する／しない
現行システム機能一覧および機能書の不存在	・存在する／しない
現行プログラム一覧および仕様書の不存在	・存在する／しない
現行プログラムモジュールとの整合性	・プログラムモジュールの作成日付とドキュメントの改定履歴の整合性がある／ない
各ドキュメントの網羅性不足	・記載ページ数／保有ステップ数(ドキュメントごと)
各ドキュメントの更新不足	・最終更新後経過期間、既知の更新漏れ回数
各ドキュメントの記載ミス、矛盾	・ミス／矛盾数(サンプルチェック)
非稼働ドキュメント・モジュールの不明	・稼働不明率(ヒアリングによる推定)
構成管理との整合性	・構成管理の更新日付とドキュメントの改定履歴の整合性がある／ない ・構成破綻率(サンプルチェック・ヒアリングによる推定)
記載難解度	・読解難易度(サンプルチェック) ・前提解説不備率(サンプルチェック) ・作成後経過年数
参照容易性	・電子化率(サンプルチェック) ・版形統一率(サンプルチェック) ・保管場所へのアクセス時間 (サンプルチェック) ・相互参照先の数(サンプルチェック)
作成者在籍率	・作成者など問合せ先の有無 ・回答精度(サンプルチェック)
保守部門での改定状況	・保守部門でドキュメントを改定していると言っている／いない

リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
軽減	ドキュメント整備作業の要・不要を判断する基準を合意する。(最低限のドキュメントに絞って見直す)	・判断基準を合意している／いない

軽減	現行システムドキュメント可読性・正確性・充分性の調査。	・調査完了サブシステム・機能数／全体サブシステム・機能数
回避	現行システムの動作を調査しドキュメントを最新化する（リバース、現行ソースコード解読含む）	・現行システムの開発経緯を知っている人の確認 ・整備率 ・キャッチアップ時点（YMD）
軽減	現行システムや業務に精通したメンバーをプロジェクトに参画させる	・現行システムや業務に精通したメンバー数÷プロジェクトメンバー数
軽減	現行システム問合せ先の設定	・問合せ先設定済みサブシステム・機能数／全体サブシステム・機能数
軽減	現行システムに携わっている人間から最新状況を取得し把握する	・現行関連組織とのレビュー
回避	現行の影響を受けないためスクラップアンドビルドに切り替える	・システムオーナーの承認
軽減	現状の担当者、利用部門や業務の有識者が参加した要件レビューを行い、要件の正確度を担保する	・レビュー参加構成

＜リスク事象ドライバー＞

現行システムのプログラム改修が優先されることでドキュメントのメンテナンスが滞り、要件定義のインプット情報が不完全になる。

＜リスク事象＞

現行システムのドキュメントが最新化されていないため、設計ポリシーやバックグラウンドでの処理が不明確となり、引き継ぐべき要件に漏れが発生する。その結果、品質未達、遅延、コスト超過のリスクが懸念される。

＜リスク事象ドライバー有無の把握＞

現行システムのドキュメントが最新のものかを、保守部門へのヒアリングや、プログラムの改修履歴とドキュメントの改定の整合性から把握する。

＜リスク事象ドライバーの予防策＞

現行機能保証をしないのが最善であるが、何らかの理由で現行機能保証をした場合、現行システムドキュメントの整備・キャッチアップを終えておくことがほぼ唯一の予防策となる。少なくとも現行システムドキュメントの可読性・正確性・充分性を調査し、不全が判明した場合には、一部または全部を現行保証範囲から外し、要件定義の進め方や検証方法についての合意を形成しておく必要がある。

ドキュメントを最新化する方法としては、①現行システムの動作からドキュメントを見直す、②現状の画面・帳票とプログラム（ソースコード）からドキュメントを見直す、③スクラップアンドビルドでドキュメントを作り直す方法がある。①の予防策は、バックグラウンドの処理（導出や他システムとの連携）や画面のオペレーション（裏技のようなもの）を網羅することが難しいので、予防策としては回避までは至らない。

現状の担当者や利用部門、有識者が要件定義工程（検討、レビュー）に参加し、要件を正確に把握し定義することが、後続工程での要件漏れによるトラブルの発生を予防するポイントとなる。

＜解説＞

このリスク事象ドライバーは、現行システム構築後のシステム改定の管理プロセスが適正に行われていないことが原因となっており、多くの開発現場で発生しているリスク事象ドライバーである。

ドキュメント不全の現行システムについて、要件定義をやり直すことなしに再構築を行うことはほぼ不可能である。ドキュメントが不全のまま保守してきたということは、要件や機能について充分な検証をしないまま、業務優先でプログラムの改修が重ねられ、ドキュメントのメンテナンスが後回し（それどころかメンテナンスさえされない）の状態を意味する。仕様は現状に合致していないだけでなく、矛盾や誤り、使われていないロジックやプログラムを内包していると思わなければならない。開発・改訂時の有識者も今や不在で、「何が正しいのか」を判断できるメンバーがいない場合も多い。

このような状態で現在動いているシステムと同様の仕様でリプレースを行なおうとすると、バックグラウンド処理が分からず、演算結果を表示しているフィールドの計算式の仕様が分からない、全ての画面の入力フィールドの遷移が把握できない、他システムとのインタフェイスがはっきりしない、などの様々な問題が発生し、仕様書レベルの整備が限界となり、業務の流れや機能、設計の背景や理由を再現することが困難となる。

このような場合の現実解は、現状を正しく認識し共有することである。再構築する以上、現行システムの機能を完全に保証することは不可能で、一部または全部について要件定義をやり直したり、現行との比較検証を実施して、どうあるべきかを決め直したりするしかないであろう。最悪な場合も想定しながら、リスクや工数に正面から向き合うことが、結局は最も安全な道である。請け負ったベンダーや現行保守チームに転嫁するだけでは解決にならないと知るべきである。ベンダーもまた、安易に受注するのではなく、受注の際には現状を調査し、ユーザーに正確な情報を伝えることが必要である。

また、「現在動いているシステムと同様の仕様でシステム構築して欲しい」という要求に対しては、現行システムに関するドキュメントやその他の情報の正確性を前提とすること、またはそれが確保できない場合には、スクラッチ開発に変更すべきことをユーザーに伝えることも必要である。

6.4　パッケージ選定に関する検討が十分でない

表 6-4 リスク事象ドライバー(4)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
パッケージ選定に関する検討が十分でない		プロジェクト主プロセス
リスク事象ドライバーによって顕在化するリスク事象		
機能の要求を満たせないことや性能が出ないことにより、大きな手戻りが発生し稼働不能になる。		
リスク事象ドライバー有無の把握		
把握の観点	メトリクス(定性的／定量的)	
パッケージ適用の目的・スコープ・採用理由が曖昧	・パッケージ適用の目的・スコープ・採用理由が明らかになっている／いない	
パッケージに関する条件確認が不足	・パッケージの適用条件(動作条件、カスタマイズ条件、サポート体制、ライセンス形態等)に関する確認の有無	
非機能要件についての検証不足	・性能等の非機能要件を満たすか検証している／いない	
パッケージの業務への適合性についての検討不足	・業務への適合性を検討している／いない	
パッケージの運用条件が曖昧	・パッケージを適用した際の業務プロセス、運用フロー等を検討している／いない	
パッケージ適用決定に向けてのレビュー不足	・レビュー実績・記録の有無(レビュー時間、参加者、指摘事項等)	
パッケージ適用決定に向けての確認者・スキル不足	・確認者・確認の記録の有無(確認者・レビュアーの属性:スキル、経験、専門分野等)	
パッケージ適用決定にあたっての承認の不備	・承認記録の有無(必要に応じ、ユーザー、経営者、管理者、関連分野の専門家等)	
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	パッケージ適用の前提条件の明文化(目的、スコープ、使用条件等)	・プロジェクト企画書等へのパッケージ適用目的等の記述の有無
回避	パッケージ適用にあたっての十分な検討	・議事録、検討結果の記録の有無
回避	パッケージ適用決定に向けてのレビューの励行	・議事録、レビューで発見された指摘事項の量と質
回避	パッケージ適用決定の承認とユーザー部門との合意	・承認者の明確化、承認の有無、ユーザーとの合意の有無
軽減	業務において必要となる主な機能を洗い出し、パッケージでの実現性を検証する	・洗い出しを行った業務一覧の有無
軽減	業務プロセス・運用設計との早期の突合せ	・レビューで発見された指摘事項の量と質

軽減	非機能要件、維持保守要件、キャパシティ要件等の整理を行う	・パッケージでの実現可否
軽減	有識者・類似事例経験者・専門部署等の検査、類似事例の検証	・重大な見落としの有無、推移、懸案の量
軽減	プロトタイピングによるパイロット試験の早期実施	・試験で発見された問題点、不備事項

＜リスク事象ドライバー＞

パッケージの適用に至る検討に不備があり、機能・業務への適合、性能等非機能要件等、様々な面の問題要因が払拭できない。

＜リスク事象＞

パッケージにより機能、業務、非機能要件が満たされず、様々な仕様変更や手戻りにより作業計画が破綻し、プロジェクトが頓挫する。

＜リスク事象ドライバー有無の把握＞

パッケージの選定に至る検討の経緯が明確になっていることが、パッケージ適用に向けての検討不足を把握する決め手となる。明確にする要素は、機能、業務との適合性、非機能要件と多岐にわたり、適用決定に向けて、レビュー、検証、承認のプロセスで経験者、有識者を配することも重要になる。これらのいずれかが欠けても、リスク事象ドライバーが発生する。

＜リスク事象ドライバーの予防策＞

選定経緯にかかる曖昧性を排除することを第一に考え、十分な調査を基に前提条件を明文化すること、検討経緯等を記録すること、レビューを確実に実施することなどが基本的な策となる。さらに適用決定の承認については、利用者や経営に近い上位者が承認すること、およびユーザー部門の合意がとれていることが重要になる。軽減する策としては、各種要件の整理や運用等実現イメージとの突合せ、経験者、有識者による確認、試験や検証行為の早期化などを挙げた。

＜解説＞

パッケージ選定の不備は、適用される業務やシステムへの影響が極めて大きく、カスタマイズの制約により機能を満たせない、性能が期待通りに出ない、問題の修復が遅くサービス開始に間に合わない、などの様々なリスクが内在している。

選定にあたっては、十分な検討や要件との突合せが重要であり、適合の判断が容易な要件だけでなく、非機能要件や動作環境、諸々の動作条件等についても検証することが重要になる。その意味でも、まずはパッケージを導入する目的を明確にし、目的に照らした要件の実現性を十分に検討し、慎重にレビューと検証を行い、その上でユーザー部門との合意を得ることが必要になる。

また、リスクの面では標準仕様や動作条件に合わせてパッケージを適用するのが優位であり、適用が安易に選定されて大規模なカスタマイズが必要になるような方向性は避けることが望ましい。

こうした様々な作業と判断を的確に行うため、検討体制には、パッケージの機能に関連する分野の有識者やパッケージ適用の経験者を参画させ、さらにはパッケージの提供元の協力など様々な手段を講じて可能な限り精度の高い検証ができるよう備えることが必要になる。

6.5　性能の検討が十分でない

表 6-5 リスク事象ドライバー(5)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
性能の検討が十分でない		プロジェクト主プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
利用に供する性能が出ないため、稼働不能や大規模改修に繋がる		
リスク事象ドライバー有無の把握		
把握の観点		メトリクス(定性的／定量的)
性能目標の明文化		・要件定義成果での性能にかかる方針の有無(性能目標、見積り資料等)
データ量の把握		・性能見積りの前提となるデータ量の資料または閲覧記録の有無(量、正確性、妥当性等)
性能見積りモデル、見積り式の妥当性		・データ量に基づいた算出の証跡の有無(計算の正誤、見積り式の妥当性、見積りモデルとの整合性)
性能見積りのレビュー		・性能に関する有識者によるレビュー記録の有無
性能目標の承認		・承認記録の有無(必要に応じ、ユーザー、経営者、管理者、有識者等)
性能検証計画		・実装に向けて性能を検証し、精度を上げて行く計画の有無 ・妥当性(誰が何をいつまで、性能保証責任の所在等)
性能検証にかかるリソース割当		・性能検証の体制の有無 ・性能検証作業に向けた要員・環境等アサインの有無
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	性能に関する計画・見積り・実装確認・テスト計画等に有識者を参画させる	・性能関連を継続的に把握・検証するチーム(常勤、非常勤問わず)の有無
回避	要件定義工程完了における性能目標、性能見積り資料の有識者レビュー	・レビューで発見された不備事項 ・所見
回避	性能目標、性能見積り資料のユーザーを交えたレビュー	・レビューで発見された不備事項 ・所見
軽減	性能目標、性能見積り資料の形式性レビュー(データ量、性能モデル等)	・レビューで発見された不備事項 ・所見
軽減	有識者による不定期なレビュー	・性能に関する有識者のレビューの実施有無
軽減	システムの実装に向けた性能検討・検証の計画の明確化	・実装に向けて性能を検証し、精度を上げて行く計画の有無 ・妥当性(誰が何をいつまで、性能保証責任の所在等)

＜リスク事象ドライバー＞

実装に向けて性能関連の検討を整理するための情報が確保できていない。また、性能目標等の検討が不足している。

＜リスク事象＞

性能等の非機能要件は、業務要件に比べるとユーザーが明示的に意識し難い面があるが、システムを安定的に稼働させるためには必須要素であることを認識して、ユーザー、ベンダー間で合意を図るべき事項である。性能が利用に供するレベルでなければ、システムを作ったところで使えないものになってしまい、サービス提供不能や大規模改修といった手痛い状況を引き起こす可能性が高い。

＜リスク事象ドライバー有無の把握＞

性能を検討する前提として、根拠とするデータ量を把握できていないことや見積りモデルが不明確なこと、実装に繋げて行くための計画がないことなどがリスク事象ドライバーとなる。

＜リスク事象ドライバーの予防策＞

性能は、非機能要件の中でも代表的な要素であり、確実な把握と検証を行うには、専門性の高い有識者を如何にしてアサインするかがポイントになる。要員と費用面の手配がつけば、体制内に非常勤でも性能チームを置くことがシンプルな予防策になる。また、最低でも要件定義工程の完了時などの重要な節目では、プロジェクトの性能に関する命運を預けられる有識者をアサインして、性能目標や性能の見積りに関するレビューを実施することが必要と思われる。軽減のためには、関連資料の形式性についてのレビューや不定期な有識者レビュー等が考えられる。

＜解説＞

性能の重要性と有識者のアサインに困難を伴うことを勘案すると、日常的な管理の中では、性能検討の形式面をチェックして不備を発見できるようにし、工程の節目となる時期までに有識者を確実にアサインして本格的な性能対策の見直しを実施することも想定する必要がある。形式面における不備の代表的な例として、性能見積りの資料でデータ量についての扱いが不適切な場合等がある。（例、見積りに使うデータ量が不正確、根拠情報が古い、データ量の変化や按分の論理が不合理等）

また、システムの実装に間に合うように性能設計の精度があがることを目標として、責任を持って性能の確保を成し遂げる計画（体制と活動時期等）を明確にしておくことは重要であり、これらを基に、早め早めに有識者を招請した対応の強化を実施することを認識しておく必要がある。

6.6　可用性の検討が十分でない

表 6-6 リスク事象ドライバー(6)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
可用性の検討が十分でない		プロジェクト主プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
安定稼働の保証がないため、稼働不能や大規模改修に繋がる		
リスク事象ドライバー有無の把握		
把握の観点	メトリクス(定性的／定量的)	
可用性確保の方針が不明確	・要件定義成果での可用性の方針の有無(目標とする稼働率、可用性確保の方式等)	
稼働率の考え方、定義が曖昧	・稼働率見積りの考え方、定義の記述有無(保守時間を含むか等)	
見積り式、見積りの妥当性	・稼働率の定義の有無。見積りプロセスのレビュー有無	
可用性対策・稼働率見積りのレビュー	・有識者(システムリスク、運用方式等)によるレビュー記録の有無	
可用性対策・目標稼働率の承認	・承認記録の有無(必要に応じ、ユーザー、経営者、管理者、有識者等)	
可用性検証の計画	・実装に向けて可用性を検証し、精度を上げて行く計画の有無 ・妥当性(誰が何をいつまで、設計責任の所在等)	
可用性検証にかかるリソース割当	・可用性検証作業(復旧テスト等)に向けた要員・環境等アサインの有無	
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	可用性の計画・方式等の実装確認・テスト計画等を工程ごとに確実に検証する	・可用性に関する工程完了目標の具体化有無 ・確認記録の有無
回避	要件定義工程完了時に可用性対策、稼働率見積りの開発・運用でのレビュー	・レビューで発見された不備事項 ・所見
回避	可用性対策、稼働率見積りのユーザーを交えたレビュー	・レビューで発見された不備事項 ・所見
軽減	可用性検討資料の形式性レビュー(目標との不整合、方式・運用等不明確)	・レビューで発見された不備事項 ・所見
軽減	工程の節目における有識者によるレビュー	・システムリスクや運用方式に関する有識者レビューの有無
軽減	可用性の方式実装に向けた検討計画の明確化	・計画、責任者・責任体制の準備と活動

軽減	可用性検証の計画の明確化	・実装に向けて可用性を検証し、精度を上げて行く計画の有無 ・妥当性

＜リスク事象ドライバー＞

実装に向けて可用性関連の検討をするための情報が確保できていない。また、可用性を検討する責任体制が不明確である。

＜リスク事象＞

可用性等の非機能要件は、業務要件に比べるとユーザーが明示的に意識し難い面があるが、システムを安定的に稼働させるためには必須要素であることを認識して、ユーザー、ベンダー間で合意を図るべき事項である。この検討が不足していれば、安定したシステムの稼働を保証することはできず、場合によってはサービス停止や大規模改修といった手痛い状況を引き起こす可能性がある。

＜リスク事象ドライバー有無の把握＞

可用性を検討して行く前提として、方針や目標の見積りができていないこと、実装に繋げて行くための計画がないことなどがリスク事象ドライバーとなる。

＜リスク事象ドライバーの予防策＞

可用性の要件は、システムの機能だけでなく、運用の仕方や設備・施設面の手配等にも関連する重要な非機能要件である。可用性についてシステムリスクや運用方式に関する専門家などを招請して精度を上げることが推奨される。しかし、そうした専門家が希少であることを鑑みれば、少なくともエンドユーザーの要求する事項をとりまとめて明文化し、目標と方針等をユーザー、開発担当、運用担当内で共有しておくことは重要であり、そのためのレビューを確実に行うことがリスク事象ドライバーの予防策にあたる。

＜解説＞

可用性の要件は、システムの機能だけでなく、運用の手順や設備・施設面の手配等にも関連する重要な非機能要件であり、要件定義完了時までに、実現性に関する専門家や適切なレビューが実施されることが望ましい。しかしながら、可用性が重要であるにも関わらず、専門家や妥当なレビュアーのアサインが難しいことを勘案すると、本項目の検討成果において早期に完璧な結論が出ることを目指すのは、かなり困難なことになる。実装までに確実に方式設計や運用設計に繋げることを意識し、責任を持って実装に繋げて行けるような検討計画を立てることが重要である。また、検討の不足や関連する課題が顕在化した時点で臨機応変かつ迅速に追加の策を検討する旨を意識合わせしておくことが望ましい。

6.7　運用要件の検討が十分でない

表 6-7 リスク事象ドライバー(7)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
運用要件の検討が十分でない		プロジェクト主プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
実現できない、矛盾を含んだ要求仕様ができる(運用要件)		
リスク事象ドライバー有無の把握		
把握の観点	メトリクス(定性的／定量的)	
システム運用関連の調整・作成計画	・システム運用の調整・作成計画がある／ない	
システム運用に対する議論	・運用要件の検討回数	
システム運用の手順書	・システム運用手順書が作成されている／いない	
障害発生時の代替・回復方法の手順書	・障害発生時の代替・回復方法の手順書が作成されている／いない	
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
軽減	システム運用関連の手順書の作成計画 (必要であればガイドラインや条例の調査を含む)	・計画がある／ない
軽減	システム運用手順書作成への、システム基盤、システム運用、業務運用のメンバーの参画	・システム基盤、システム運用、業務運用のメンバーが参画している／いない
回避	障害発生時の代替・回復方法の手順書作成への、システム基盤、システム運用、業務運用のメンバーの参画	・システム基盤、システム運用、業務運用のメンバーが参画している／いない

＜リスク事象ドライバー＞

情報システムの構築に主眼がおかれ、リリース後の運用手順について議論がおざなりになる。

＜リスク事象＞

システム運用の複雑化や情報セキュリティ上の問題など、安全・安心して使えるシステムの運用が難しくなる。

＜リスク事象ドライバー有無の把握＞

システム運用の検討計画の有無、実際のシステム運用に関わる成果物があるかないかで、リスク事象ドライバーの有無を把握する。

＜リスク事象ドライバーの予防策＞

妥当性のあるシステム運用について議論するために、システム運用に関連する仕様書の作成計画(何を、いつ、誰が検討する)を明確にし、適切な議論が行える状況を設定する。また、システム運用は、HW 故障時の復旧および情報の取扱いなどに対応するシステム基盤部分、バックアップやアプリケーション障害などに対応するア

プリケーション運用、システム停止時などに対応する業務運用が関連するので、これらに関わるメンバーでの検討やウォークスルーが必要である。

＜解説＞

運用要件の検討を後回しにすると、プロジェクトは、工程が進むにつれ情報システムのリリースに向け佳境になるため、運用体制や役割の明確化、運用のためのシステムへの機能の組込み、実効性のある運用手順などが不十分のまま運用に入ることになる。その結果、トラブル発生時の障害解析に時間がとられ復旧に時間がかかったり、ハードウェア障害でディスク交換したがディスクの内容に機微な情報が含まれており情報セキュリティ上の問題が発覚したりする、などの問題が発生することが懸念される。

業務要件や業務と情報システムの関係を明確にする、要件定義段階と設計段階で、運用要件や運用のための機能を明確にし、運用面での機能も含めた機能設計を行うなど、運用面を考慮した要件・システム設計が必要である。

6.8　運用に向けての制約条件が明確でない

表 6-8 リスク事象ドライバー(8)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
運用に向けての制約条件が明確でない	プロジェクト主プロセス
このリスク事象ドライバーによって顕在化するリスク事象	
期待された業務運用ができない	
リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的／定量的)
運用についての制約・条件が要求項目にあるか	・要求項目に運用時間、ピークタイムなどがある／ない
運用設計の体制(組織、運用体制、運用開始時期、業務に対する有識者・経験者の有無など)や業務プロセスが明確であるか	・業務フローがある／ない ・業務手順書がある／ない
データ量やスループットなどが見積もられているか	・設計時点でデータ量、トランザクション量、想定所要時間などの具体的な数値が見積もられている／いない
システム運用の手順書等が整備されているか	・システム運用手順書が整備されている／いない
バックアップ等の作業を考慮した運用を想定しているか	・バックアップ等の作業を想定した運用スケジュールを計画している／いない
障害発生時の代替・回復方法を検討しているか	・障害発生時の代替・回復方法の検討がされている／いない
要件定義やドキュメントレビューに運用担当者が参加しているか	・レビュー参加者に運用担当者が参加している／いない
業務視点でのウォークスルーがスケジュールされているか	・ウォークスルーの実施計画がある／ない
テスト項目として業務パターンを洗い出しているか	・業務パターンに関するテスト項目数、網羅率
業務シナリオに基づいたテストシナリオがとりまとめられているか	・業務シナリオに基づいたシナリオがある／ない ・(ある場合)テストシナリオのページ数、レビュー回数、レビュー参加人数、レビュー参加構成に運用担当者が参加している／いない
業務運用を考慮したユーザーへのトレーニングの実施計画があるか	・導入後の業務運用を想定したトレーニング計画がある／ない
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	非機能要求の項目を具体化し、ユーザーの業務に合わせた要件定義を文書化する	・非機能要求グレードによるスコアリング
回避	運用担当組織の決定権者、担当者や業務に精通した有識者が参	・マイルストーンの定義 ・ユーザヒアリング回数

	加したヒアリング、レビューの実施	・参加者構成 ・参加人数
回避	要件定義書を基に現場やステークホルダーが参加した業務のウォークスルーを実施することをプロジェクト計画に織り込む	・業務のウォークスルー計画（回数、参加者構成、参加人数）
回避	業務シナリオに基づいたテストの実施、およびテストシナリオレビューを運用組織の担当者が参加して実施するようにプロジェクト計画に織り込む	・業務に基づいたテストシナリオのレビュー実施回数
回避	非機能要求についての専門家、有識者を交えたレビューの実施	・非機能要求に対してのレビューの実施回数
軽減	RFP の作成段階で非機能要求項目を考慮して作成する	・非機能要求グレードの項目数

＜リスク事象ドライバー＞

要件定義工程で運用要件の検討・設計、考慮不足

＜リスク事象＞

運用要件の検討不足によって期待された業務運用ができない。

＜リスク事象ドライバー有無の把握＞

＜リスク事象ドライバーの予防策＞

このリスク事象を回避するには、要件定義段階でユーザー側とベンダー側の双方が運用の制約条件を検討すべき課題として理解、共有することが重要となる。このために、要件の詳細化を進める過程で、非機能要求グレードを利用して対象業務の運用条件を具体的な項目に落とし込むことが効果的な施策となる。

さらに、要件定義後の工程では①現場ユーザー側が参加したウォークスルーの実施、②業務シナリオに基づいたテストの実施、③運用と実現しようとしているシステムとのかい離がないかを検証するための評価レビューの実施をマイルストーンとしてプロジェクト計画に織り込むことが後続工程でのトラブルの発生を予防するポイントとなる。

＜解説＞

要件定義段階以降の工程で運用の制約条件の漏れを要因とした問題が発覚した場合、ユーザー側とベンダー側双方にとって大きなトラブルとなる可能性が大きい。このため、要件定義段階では適切に運用の制約条件を検討して文書化することが求められる。例えば、既存システムの保守型開発においては、既存の運用要件を

踏襲することを暗黙の前提としていることが多いため、運用要件の考慮不足や非機能要求の漏れが要件定義工程以降で露呈する確率が高い。

そのような状況を回避するために、要件定義段階から意識的に非機能要求についてユーザー側とベンダー側の双方で課題として共有する施策を計画、実施する必要がある。

要件定義段階のベンダー側の姿勢として、非機能要求グレードをガイドラインとして利用して、例えば、定時運用時間、稼働率、レスポンスタイム、業務量増大度、ピークタイムなどの項目をユーザーの業務特性や条件を加味して課題として提示し、専門的な有識者の知見を活かしながら、プロジェクトとしての優先度や達成目標、条件を具体的に文書化して合意形成することが非機能要求の要件漏れを防止するための施策として有効である。また、ユーザー側にとっても、この検討や成果を通じて、カタログ性能を鵜呑みにせずに、実際の運用条件を踏まえて現場ユーザーやステークホルダーへのヒアリングやレビューを行って業務手順書などをとりまとめることで経営層や業務組織にコンセンサスをとった業務条件を決定することができる。

6.9　要件を獲得すべきステークホルダーが網羅されていない

表 6-9 リスク事象ドライバー(9)記述内容

<table>
<tr><th colspan="3">リスク事象ドライバー</th></tr>
<tr><th colspan="2">リスク事象ドライバーの内容</th><th>区分</th></tr>
<tr><td colspan="2">要件を獲得すべきステークホルダーが網羅されていない</td><td>プロジェクト主プロセス</td></tr>
<tr><th colspan="3">このリスク事象ドライバーによって顕在化するリスク事象</th></tr>
<tr><td colspan="3">要件の漏れが発生する(後になってステークホルダーから指摘される)。</td></tr>
<tr><th colspan="3">リスク事象ドライバー有無の把握</th></tr>
<tr><th>把握の観点</th><th colspan="2">メトリクス(定性的／定量的)</th></tr>
<tr><td>ステークホルダー分析を実施しているか</td><td colspan="2">・ステークホルダーのリストがある／ない
・リストのレビューをしている／いない(レビュー記録確認)</td></tr>
<tr><td>ステークホルダーごとに合意形成方法が明確になっているか</td><td colspan="2">・明確になっている／いない</td></tr>
<tr><td>洗い出したステークホルダー全員にヒアリングを実施しているか</td><td colspan="2">・ヒアリングの実施率</td></tr>
<tr><td>システム部門の社内統括力</td><td colspan="2">・エンドユーザーをとりまとめている／いない
・業務部門の状況やキーマンを把握している／いない</td></tr>
<tr><th colspan="3">リスク事象ドライバーの予防策</th></tr>
<tr><th>対応区分</th><th>予防策の内容</th><th>予防策の有効性を確認するメトリクス</th></tr>
<tr><td>回避</td><td>ステークホルダーの分析を着実に実施する(リストの作成とレビュー)</td><td>・リストのレビュー記録</td></tr>
<tr><td>回避</td><td>ステークホルダー全員から着実にヒアリングできるよう戦略を立てる</td><td>・ヒアリングの実施率</td></tr>
<tr><td>回避</td><td>選定されたステークホルダーが、対象業務の要件を代表することを合意する</td><td>・ステークホルダー参画に対する組織的コミットメントの有無</td></tr>
</table>

＜リスク事象ドライバー＞

システム化すべき要件を握るステークホルダーの選定に漏れがあり、情報収集する対象が網羅されていない。あるいは選定されたステークホルダーに対するヒアリングが十分できていない。

＜リスク事象＞

一部のステークホルダーがヒアリング対象から漏れていても、それなりに要件が収集できていれば気付かないまま要件定義が完了してしまい、後工程になって致命的な要件の漏れや前提条件の見落としが発覚する懸念がある。

また、ステークホルダーが多忙でレビューを欠席したり、時間切れでヒアリングできなかったり、合意形成が不十分だったりした場合にも、同様の問題が起こり得る。

＜リスク事象ドライバー有無の把握＞

表６－９は、必要十分なステークホルダーからヒアリングするための、ごく基本的な項目を示しており、少なくともこのいずれかに心当たりがあれば、ユーザーとベンダーが協力して早い段階で対応すべきである。
プロジェクトによっては、これ以外の観点も必要になるだろう。

一般にヒアリングする対象が増えるほど要件も膨らみ、要件定義にも時間がかかるため、対象を絞ることは大切だが、それで重要なステークホルダーを漏らしては本末転倒となるので注意が必要である。

＜リスク事象ドライバーの予防策＞

システムの要件を持つステークホルダーを探し出し、確実に要望を収集することは、要件定義の漏れを無くす上で重要なポイントであるにも関わらず、配慮不足に陥りやすい。

システム構築において、全てのステークホルダーの要望を 100%実現させることは困難なので、できる限り多くのステークホルダーの納得が得られるよう戦略を立てる必要がある。

そのためにはステークホルダーの中でもキーマンを探し出し、対象業務に関する要件決定に関わることに合意してもらうことが重要である。時には情報システム部門やエンドユーザー側の主担当者も気付かない人物がキーマンだという可能性もある。

ステークホルダー（キーマン）を洗い出す、という観点においては、単に声の大きい人物をリストアップするのではなく、業務部門での職位や知識レベル、業務部門内で信頼を得ているか、他のキーマンとの力関係なども考慮する必要がある。

例えば、その人物は、声は大きいが業務部門で浮いており、全体を代表するような意見は出せなかったり、役割が限定的で知識が偏っていたり、レビューでの発言力が弱かったりする場合は適任な選定だとは言えない。
そのようなことがないように考慮すべき観点を事例として下記に示す。

・当該システムまたは業務の投資可否や投資効果に責任を持つ人／組織
・当該システムの操作を実際に担当する人／組織
・当該システムの出力物を利用する人／組織
・当該システムまたは業務の内部統制や監査を担当する人／組織
・当該システムまたは業務のリスク管理を担当する人／組織
・当該システムまたは業務の法務／コンプライアンスを担当する人／組織
・当該システムの保守／運用を担当する人／組織
・当該システムの設計／開発を担当する人／組織
・当該プロジェクトのリソース調達／配置を担当する人／組織
・当該プロジェクトによって業務上影響を受ける人／組織
・当該プロジェクトによって取引が発生／消滅する人／組織

ステークホルダーから着実にヒアリングする、という観点においては、忙しいキーマンを如何に参画させるかがポイントとなる。

そのためには、多忙であってもセッションやレビューに参画しようというモチベーションを高める必要がある。いくら都合の良い時間を調整しても、モチベーションが低ければ「突発事項」を理由に欠席されてしまう懸念は払拭できない。

予防策としては当然のことながら、参加者の負担を極力軽くするためにセッションなどを効率的に実施すること、参加者に関わるテーマになっていること（関係が薄いテーマの聞き役のみにならない）、参加者が一定の責任を担っていること、などが挙げられる。

場合によっては TV 会議システムや電話による参画、メールや SNS を利用した参画、空き時間に個別ヒアリングで対応するなど、参加者が置かれている状況に応じた工夫も必要である。

＜解説＞

ステークホルダーの選定漏れやヒアリング漏れがあった場合、システム化に必要な要望や前提条件が十分に収集できないまま仕様が決定されてしまうことになりかねない。要望が仕様に反映されなかった場合、後工程になって重要な考慮漏れが指摘されるケースもある。最終段階で「使用に耐えない」、「業務が回らない」という事実が発覚すれば、プロジェクトが暗礁に乗り上げる可能性もある。

厄介なのは、収集された要件のレビューはできでも、収集されていない要件はレビューしようがなく、漏れている事実に気付くこと自体が難しいということだ。

このようなことを防ぐためにも、ステークホルダーの洗い出し、ステークホルダーのプロジェクト参画への配慮は慎重に行うべきである。

6.10 システム部門による要件とりまとめが十分でない

表 6-10 リスク事象ドライバー(10)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
システム部門による要件とりまとめが十分でない		プロジェクト主プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
要件がいつまでも確定しない、あるいは要件が想定以上に膨らむ。		
リスク事象ドライバー有無の把握		
把握の観点	メトリクス(定性的/定量的)	
要件の引き出しプロセスが明確になっているか	・明確になっている/いない	
要件のとりまとめ様式が定義されているか	・定義されている/いない	
システム部門の立場は、ユーザー部門に対してどうか	・要件の最終決定権がシステム部門にある/ユーザー部門にある	
システム部門の業務知識は十分か	・要件をまとめるのに十分/不十分	
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
軽減	ユーザー部門との間で要件の引き出しプロセスを明確にし、合意する	・要件の引き出しプロセスが明確になっている/いない ・要件の引き出しプロセスがユーザー部門と合意されている/いない
軽減	要件のとりまとめ様式を定義し、その様式に沿って要件を整理してゆく	・要件のとりまとめ様式が定義されている/いない
軽減	要件に関してシステム部門に決定権を与える	・要件の決定権がシステム部門にある/ない
軽減	業務知識を補うためのサポート体制を作る	・サポート体制がある/ない
軽減	要件定義の進捗があがらない場合、エスカレーションして体制を補強する	・体制におけるスキルの総和 (業務知識、類似経験要員数)

<リスク事象ドライバー>

要件を引き出すことができない、出てきた要件を上手くまとめられない、まとめるための様式が決まっていない等

<リスク事象>

要件が上手く引き出せなければ、いつまでも要件が確定しなかったり、想定で決めた要件が後になって齟齬となり、手戻りになる可能性がある。一方、要件が上手く整理できていなければ、出された要望が全て要件として受け入れられ、要件が想定以上に膨らんでしまう可能性がある。

＜リスク事象ドライバー有無の把握＞

このドライバーには、システム部門の力量や業務知識、ユーザー部門との力関係などの要因が大きく影響している。それらが十分かどうかという判断は非常に難しいが、プロセスや様式が整備されているか否かは間接的ながら明確な判断指標になりうる。

＜リスク事象ドライバーの予防策＞

複数の部門が利用するシステムにおいては、システム部門が全体のとりまとめをしなければ仕様がなかなか固まらない。そのことを経営に理解してもらい、システム部門に仕様をまとめるための道具や権限、業務知識を補うためのサポートをとりつけるなど、システム部門を中心にできる限りの対応をすることが重要である。

＜解説＞

ユーザー企業内に要求を出すユーザー部門と、要件としてまとめるシステム部門がある場合、システム部門が要件定義に果たす役割が重要となる。システム部門が最終的にシステムの機能や費用に責任を持つことから、要件定義はシステム部門が主導することが多い。

しかし、システム部門の業務知識不足やユーザー部門に対する立場の弱さなどから、要件を上手くまとめられないケースは少なくない。まとめる手順や役割分担が曖昧だったり、様式等が整備されていなかったりすることも要因になりうる。

要件がまとまらなければ要件定義が収束できなかったり、曖昧なまま次工程に進まざるを得なかったりすることになり、後から大きな手戻りに繋がる可能性がある。また、システム部門を飛び越えて、ベンダーが直接要件をとりまとめるような状況にもなりかねず、この場合、要件確定の責任所在が曖昧になり、後になってユーザーとベンダーとの責任問題や費用負担問題に波及する懸念もある。

ただし、システム部門が権限を持つことで、本来の要件から離れる可能性もあるので、決定した後にユーザー部門と協同での確認行為や試験等の計画を立てることも重要である。

6.11 ドキュメントの更新が管理されていない

表 6-11 リスク事象ドライバー(11)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
ドキュメントの更新が管理されていない		支援・管理プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
追加・改修開発時に、現行システムから引き継ぐべき要件に漏れが発生する		
リスク事象ドライバー有無の把握		
把握の観点	メトリクス(定性的 / 定量的)	
変更管理を実施しているか	・実施している／いない	
要件のトレーサビリティーを確保しているか	・確保している／いない	
要件の変更が発生した場合、ドキュメントに反映しているか	・反映している／いない	
テスト時にソースを修正した場合、ドキュメントも修正しているか	・修正している／いない	
ドキュメントを修正した場合、レビューを実施しているか	・実施している／いない	
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	変更管理工数を、見積り段階で考慮しておく	・変更管理の見積り実施有無
回避	要件のトレーサビリティーを確保する	・トレーサビリティーを示すドキュメントの有無
回避	要件やソースの変更時には、関係するドキュメントを修正する	・ドキュメントへの変更反映率(反映数／変更数)
軽減	修正ドキュメントのレビューを実施する	・修正ドキュメントのレビュー時間 ・指摘数／レビュー時間
軽減	追加・改修開発の要件定義時に、ドキュメントの最新性を確認する	・最新性の確認タスクがある／ない

＜リスク事象ドライバー＞

要件の追加や変更があった場合、また、テスト工程で検出した欠陥対応のためソースコードを修正した場合に、関係するドキュメントが修正されないまま放置され、現行システムと一致していないことがある。

＜リスク事象＞

現行システムの追加・改修開発時に、現行システムとドキュメントの間に不一致や漏れがあっても気付かないまま作業を進めてしまう可能性がある。

＜リスク事象ドライバー有無の把握＞

要件や設計に変更が生じた際、ドキュメントを修正するべきかについては修正およびレビューを実施しているかによって把握する。

＜リスク事象ドライバーの予防策＞

ソフトウェア開発においては、一旦ドキュメントが完成した後でも、要件や設計の変更が発生した場合には、変更内容を関係するドキュメントに反映することが重要である。

反映漏れを無くすためには、何をどのように変更したかを管理しておくことや、変更の影響範囲を把握できるようにトレーサビリティーを確保しておくことが必要である。

また、ドキュメントが正しく修正されたことを確認するためのレビューも必要である。

さらに、追加・改修開発の要件定義フェーズにおいて「現行との差」や「現行どおり」という要求があった場合に、現行システムの確認をする必要があるが、参照する現行システムのドキュメントが最新かどうかをあらかじめ把握し、最新でない場合は、差分の穴埋めのための作業やユーザー確認のタスクを計画する必要がある。

＜解説＞

国内では、ソフトウェア開発がウォータフォール型で進められることが多い。このため、要件定義が終わった後は、その内容に基づいて設計やコーディングを進めたいが、実際には、要件定義終了後に要件の追加や変更が発生することがある。 要件の追加や変更が発生した場合には、最終成果物と整合性を確保するために、関係するドキュメントを修正する必要がある。また、テスト工程になってから、要件の漏れや設計のミスが発覚した場合にも、関係するドキュメントを特定し、修正しなければならない。

ドキュメントの修正は、いつでも簡単にできるわけではなく、修正が必要になるタイミングに依存する。一般的に、ドキュメントを作成した工程と、そのドキュメントの修正が必要になったタイミングが離れれば離れるほど影響範囲も大きくなり、反映するための作業量は大きくなる。

このため、ソースコード作成以降は、お客様に直接影響が及ぶソースコードの修正が優先され、関係するドキュメントの修正は、優先順位が低くなる場合がある。例えば、テスト工程になってから要件の漏れを検出した場合には、工数や期間の制約もあるため、ソースコードの修正は行われても、要件定義書の修正は優先順位が下げられてしまい、結局修正しないままになってしまう。

システムとドキュメントの内容が一致していない状態になっても、すぐに困るとは限らないが、基盤更改のタイミングで、再構築や大規模改修を行う場合など現行システムの要件や設計を確認したい場合には、頼りにするものがなく、困ることになる。

過去の資産を活かす基盤更改や保守開発は少なくない。開発当初から、変更管理やトレーサビリティーの確保に努める必要がある。

また、追加・改修開発の要件定義時に、既存のドキュメントを闇雲に「正」としてしまうと実システムとドキュメントの記述に差があることに気付かない可能性があるため、差異の有無を確認し、差異がある場合にどうすべきかをユーザー側と合意しておく必要がある。

6.12 仕様の変更管理ができていない

表 6-12 リスク事象ドライバー(12)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
仕様の変更管理ができていない		支援・管理プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
規模の肥大化、費用に関する PJ の目標未達		
リスク事象ドライバー有無の把握		
把握の観点	メトリクス(定性的 / 定量的)	
仕様変更の一覧化	・公式に維持されている仕様変更一覧の有無	
仕様変更一覧における仕様変更の承認の不備	・押印等承認記録の有無 ・承認者名が形骸化(直接関連しない上位者に統一されている等)	
仕様変更内容の確認を行う会議体	・会議体を定めたルールの有無 ・会議体の開催条件が不明確(主催者、参加者、記録等)	
仕様変更内容の概要や明細記録	・仕様変更内容に関する明細記録の有無(資料の乱丁、欠落等含む)	
仕様変更管理の確認・審議・承認の記録	・仕様変更の確認・審議・承認の方法の文書化の有無 ・同様内容の記録(議事録等)の有無	
確認・審議にあたっての有識者の参画	・仕様変更の内容に応じて、技術面などの有識者を参加させるル―ルの有無 ・記録の有無	
仕様変更の影響範囲の把握	・仕様変更の影響範囲、影響の大きさなど、費用・稼働を推定できる情報の有無	
仕様変更管理と投資等管理の関係	・経営層による仕様変更一覧の確認の有無(承認、閲覧の印、記録等)	
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	① 公式な(定期的に最新化される)仕様変更の一覧を制定する	・仕様変更一覧の存在 ・最新化されているか(更新日付)
回避	② 仕様変更内容の承認行為を確実に実施する	・仕様変更の確認・承認記録の有無
回避	③ 仕様変更の内容について必ず明細票を作成する	・仕様変更に関する明細票(事象、検討内容等)の有無
回避	④ 仕様変更の内容確認を会議体や有識者を交えて行う	・会議の議事録、有識者との検討の記録の有無
軽減	⑤ 仕様変更管理の有識者が関連の記録一式をレビュー(不定期)	・レビュー記録の有無

軽減	⑥　定期的な仕様変更管理の影響範囲・規模等の記述のチェック	・記述不備数、同推移（継続的に発生しているか、増加しているか等）
軽減	⑦　仕様変更一覧に関する経営のレビュー（不定期）	・レビュー記録の有無

＜リスク事象ドライバー＞

変更管理における仕様変更の可視化ができていない。物理的な意味での会議や管理帳票の不在、内容確認・承認の不備など。

＜リスク事象＞

仕様変更が可視化されていないことにより構築の規模等の基本的な管理に不備が生じ、追加開発の規模が膨らむなどの事象からプロジェクトのQCD悪化や、目標未達につながる。

副次的には、優先順位の明確でない作業の肥大化により、要員のモチベーションの低下やステークホルダー間のトラブル等に発展する可能性もある。

＜リスク事象ドライバー有無の把握＞

仕様変更の案件の一覧化やトレーサビリティー確保等の記録、およびそれを維持する仕組みに問題がある場合リスク事象ドライバーとなる。

＜リスク事象ドライバーの予防策＞

仕様変更のコントロールは、システム構築プロジェクトにおける最重要な管理事項の一つであり、管理すべき事項も細かく挙げたらきりがない。まずは、予防策として、仕様変更に関する情報を多くの関連者が共有、具体的には一覧表と明細票を作成し、それらを最新化するルールの整備を行うことが必須事項となる。さらに、プロジェクトの体力に応じて一覧表、明細票、及びそれらの確認・審議・承認等の記録について、精度を確認するための仕組みを用意する。ここでは有識者（過去の管理経験者等）によるレビュー等が効果的と思われる。また、仕様変更管理の真の目的は、ユーザー要求とその実装の対応を可視化し、構築作業の健全化に資することであるので、一覧表にあるような大局の情報については、ユーザー、ベンダー双方で経営層を含めて共有することが重要である。

＜解説＞

ユーザー要求とその実装の対応を経営者や管理者が正しく認識できるような精度で可視化するには、仕様変更の内容を幾つかの要素（対象、作業難度、影響範囲等、ステークホルダーに係る制約等）で具体化した上で、仕様変更の実現性や有効性などを検討の土俵に乗せ、それを根拠に改修や機能追加等、後段の作業を定義することが必要になる。これら全てを精緻に机上で詰め、詳細に記録することは、大きな稼働とスキルを要するので、プロジェクトの体力によっては現実的でない場合がある。

組織的に有効な対応をとるためには、予防策の項で述べたように一覧表と明細票を確実に最新状態に保つことが基本になるが、その上で会議体や有識者（仕様変更や総合的プロジェクト管理の経験者等）によって、記録上の不備を記述者や関係者が正す活動が重要になる。

また、仕様変更を検討していく過程で発生する技術的な課題や、ステークホルダーに関連する懸案については、各分野の専門家や当事者に確実に問い合わせ、レビューを行うこと（その記録をとること）などが結果的にリスクを顕在化させない対策にあたる。

6.13 ユーザーによる仕様の確認が充分でない

表 6-13 リスク事象ドライバー(13)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
ユーザーによる仕様の確認が充分でない		支援・管理プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
要件の漏れや認識の齟齬が生じる		
リスク事象ドライバー有無の把握		
把握の観点		メトリクス(定性的 / 定量的)
要件定義確認、完了時までの要件確認やドキュメントレビューが充分になされているか		・ヒアリング回数、ヒアリング参加人数、ヒアリングに現場ユーザーの参加している/いない、 ・ドキュメントレビュー回数、指摘数、改版数、ドキュメントのエラー分類が評価されている/いない
要件検討やレビュー結果がユーザー側と共有されているか		・レビュー回数、指摘数、QA 数 ・レビューチェックリストの項目数
非機能要求がドキュメント化されているか		・非機能要求グレードのスコアリング
業務視点でのウォークスルーがスケジュールされているか		・ウォークスルーの実施計画あり/なし
業務パターンを洗い出し、業務シナリオに基づいたテストシナリオが取りまとめられているか		・シナリオがある/ない ・(ある場合)テストシナリオのページ数、レビュー回数、レビュー参加人数、レビュー参加構成に運用担当者が参加している/いない
業務パターンを洗い出し、テスト項目としているか		・テストシナリオのテスト項目数 ・テストシナリオのテスト網羅率
要件定義完了時点でユーザー側組織に対して報告責任が適切になされたか		・ステアリングコミッティの開催あり/なし
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	要件定義書を基に現場ユーザーやステークホルダーが参加した業務のウォークスルーを実施することをプロジェクト計画に織り込む	・ウォークスルー計画(回数、参加者構成、参加人数)
回避	業務シナリオに基づいたテストシナリオのレビューを現場ユーザーや運用組織の担当者が検証する	・テストシナリオのレビュー担当者の選定、実施回数
軽減	テスト項目での業務パターンの網羅性の検証	・テスト項目としての業務パターンの網羅率
軽減	UI、画面デザイン、帳票レイアウトはプロトタイプを作成し、具体的に見えるもの、触れるものとして仕様の確認を行う	・プロトタイプや画面イメージ等を利用して仕様確認のタスクがある/ない

軽減	非機能要求を具体的に文書化する	・非機能要求グレードのスコアリング
軽減	システムと業務との関わりが分かるフローなどのドキュメントを整備する	・業務フローが作成されている/いない
受容	要件定義工程完了以降で要件確認をフォローするための具体的な現場へのレビュータスクや要件変更の受容期日をスケジュール化して合意する	・要件の再確認タスクは計画されている/いない ・承認のマイルストーンがプロジェクトで計画されている/いない
受容	テスト段階で発生した要件変更を管理し優先度を付けた上で別スケジュールとして対応する	

＜リスク事象ドライバー＞

要件定義工程以降の試験工程において、テスト段階で初めてシステムに触れる。

要件の精度を確認する計画が不明。

＜リスク事象＞

要件定義に漏れがある状態で要件定義工程が完了し、製造工程が進む（最終的にテスト段階で発覚する）こと。

＜リスク事象ドライバー有無の把握＞

＜リスク事象ドライバーの予防策＞

本来ならば、この事象が起きないように、あらかじめプロジェクトがキックオフした段階から現場ユーザーや業務に精通しているメンバーに参画してもらう体制を作ることが望ましいが、要件定義で作成された文書を基に、現場ユーザーやステークホルダーが参加したウォークスルーを実施し、要件の漏れや認識の齟齬を解消する機会をあらかじめプロジェクト計画に織り込むことが重要である。また、特にベンダー側では、ユーザー側との要件の漏れや認識の齟齬が生じないようにするために、プロトタイプを作成するなどの具体的に見えるもので仕様を確認して稼働後のイメージを共有することに配慮したり、業務シナリオに基づいたテストシナリオのレビューを現場ユーザーに依頼するなどの施策を取り入れる工夫が重要となる。

さらに、要件定義完了前にやむを得ず確認できなかった場合も含め、要件の再確認や承認計画をタスク化しておく必要がある。

＜解説＞

テスト段階になって初めてシステムに触れた時点で、ユーザー側とベンダー側との認識のずれや要件定義に漏れが発覚すれば、プロジェクトは追加稼働や手戻り等で多大な影響を受け、ユーザー、ベンダーは相互に大きな被害を受けることになる。このような事態を避けるために、適切な時期に、適切なカタチでのユーザーの参加、及び協力を求める策を計画に織り込む必要がある。

仮にウォークスルーやプロトタイプ画面や具体的なイメージを提示してしたとしても、現場ユーザーやステークホルダーから指摘される要件漏れや現行業務の不整合点が全くないことが担保されるわけではなく、場合によっては現場の意向に沿った形での新たな要求が提示されることにもなる。

したがって、それらの予防施策を計画すると同時に、ユーザー指摘事項に関して①ユーザー側と合意の上で取捨選択、優先順位を付けて管理する、②課題の解決策は、後続作業での対応漏れが生じないようにドキュメントを最新化し、ユーザー側でもトレースができるようにする、等の施策をあらかじめ合意しておく必要がある。

6.14 要求の優先度が曖昧になっている

表 6-14 リスク事象ドライバー(14)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
要求の優先度が曖昧になっている		支援・管理プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
開発規模が膨張する		
リスク事象ドライバー有無の把握		
把握の観点	メトリクス(定性的 / 定量的)	
要求に優先度付けを実施しているか	・実施している/いない	
要求の先後関係を明らかにしているか	・実施している/いない	
要求の優先度をお客様に確認しているか	・実施している/いない	
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	システムのスコープを明確にする	・スコープ内外が明確になっている要求数/全要求数
回避	システムの目的と要求のトレーサビリティーを確保する	・トレーサビリティーが確保されている要求数/スコープ内の全要求数
回避	要求に優先度を付ける	・優先度付き要求数/スコープ内の全要求数
回避	要求の優先度についてお客様の承認を得る	・承認された優先度付き要求数/全優先度付き要求数

<リスク事象ドライバー>

要件定義またはシステム要件定義・ソフトウェア要件定義に於いて、洗い出された要求に優先度を付けていない、あるいは優先度が曖昧になっている。これに関連して、要求に先後関係(あることを行うためには、その前提として行わなければならないことがある)がある場合には、それが曖昧である場合も含まれる。

<リスク事象>

優先度の低い要求が混入することで、当初の見積もり以上に規模が増加する可能性がある。

ユーザーにとってはさほど優先度が高くないにも関わらず、システム化する上では難度が高い要求があった場合、その実現の検討に多くの時間を費やしてしまう等の無駄が生じやすい。

<リスク事象ドライバー有無の把握>

要求に優先度を付ける作業を実施しているか、要求間の先後関係を明らかにしているか、また、その結果についてエンドユーザーを含むステークホルダーに確認しているかによって把握する。

＜リスク事象ドライバーの予防策＞

そもそも、優先度がつけられないのは、システムの目的やスコープが明確になっていないことが原因で起こることが多いため、まずその点を明確にする。

その上で合意した納期や予算に合わせてシステムの目的を満たすため、要求に優先度を付けたり切り捨てたりする必要がある。

まとめると、次のようになる。

①要求を実現するためにどこまでをシステムで実現するべきか、スコープを明確にする。

②どの要求を実現すればシステムの目的を満たせるのか明確にするため、目的と要求のトレーサビリティーを確保する。

③スコープ内の要求に優先度を付ける。要求の変更や追加があれば優先度を見直す。

④優先度を付けた要求について、顧客側責任者およびステークホルダーに承認を得る。

＜解説＞

システム開発を進める上で、要求に優先度を付けることは重要である。優先度が明確になっていないと、要求の追加や変更を求められた場合に、何をどこまで反映するべきかの判断が難しくなってしまう。そこで、要求の優先度はどのように決めるべきかを考えたとき、システムの目的およびスコープを基に判断するのが良いと言える。

つまり、目的に対して貢献度の高い要求の優先度を高くするべきである。そうしなければ、投入するコストや工期と、システムから得られる効果が見合わなくなってしまう可能性がある。

開発途中で要求の追加や変更が発生した場合の対応も、その要求の優先度を考慮する必要がある。もし、追加・変更される要求の優先度が高い場合には、当初の計画を維持するために優先度の低い要求を除外するのか、それとも計画を見直して実現時期をずらすのか等、検討することになる。

顧客側責任者およびステークホルダーと、システムの目的やスコープ、要求の優先度を合意さえできていれば、要求の追加・変更で開発規模が膨張する場合でも、計画の変更について調整しやすくなると言える。

6.15 業務要件の網羅性が検証できない

表 6-15 リスク事象ドライバー(15)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
業務要件の網羅性が検証できない	支援・管理プロセス
このリスク事象ドライバーによって顕在化するリスク事象	
画面、帳票等の形式的、定型的な情報はあるが、業務の全体にわたる要件や要望、今後の展望などがあいまいである	
リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的 / 定量的)
提示された現行の画面や帳票の内容は、ドキュメントに記載された内容から理解可能か、あるいはドキュメントは確認する上で適切な状態であるか	・現物とドキュメントの比較において相違がない/ある ・現物に即して最新化が図られている/いない
現行の画面や帳票の稼働周期や制限などが明示されているか	・いる/いない
現行の画面や帳票で利用されていないものがあるか	・ある/ない
提示された現行の画面や帳票以外から業務要件を把握するための人的な体制が準備されているか	・業務とシステムの関連を把握している有識者が存在する/しない ・QA 回答が適切に回答される/されない
要件定義確認、完了時までの要件確認やドキュメントレビューが充分になされているか	・ヒアリング回数、ヒアリング参加人数、ヒアリングに現場ユーザーの参加している/いない、 ・ドキュメントレビュー回数、指摘数、改版数、ドキュメントのエラー分類が評価されている/いない
データ量やスループットなどが見積もられているか	・設計時点でデータ量、トランザクション量、想定所要時間などの具体的な数値が見積もられている/いない
非機能要求が具体的に明示されているか	・非機能要求項目を考慮している/いない ・非機能要求グレードのスコアリングを行っている/いない
業務視点でのウォークスルーがスケジュールされているか	・ウォークスルーの実施計画がある/ない
業務パターンを洗い出し、業務シナリオに基づいたテストシナリオが取りまとめられているか	・シナリオがある/ない ・(ある場合)テストシナリオのページ数、レビュー回数、レビュー参加人数、レビュー参加構成に運用担当者が参加している/いない
業務の適合性やパッケージからの変更点が認識されている	・Fit/Gap が実施されている／いない ・Gap の数が許容範囲である／ない

要件定義完了時点で組織に対して報告責任が適切になされたか	・プロジェクトマネジメントレビューやステアリングコミッティの開催した/しない
(パッケージ導入の場合)デモンストレーションなどにより実際の画面や帳票と具体的に比較検討しているか	・デモンストレーションが実施されている/いない ・実施されている場合 回数、参加人数、参加者の人選、現行の機能との比較検討がされている/いない
(パッケージ導入の場合)プロジェクト目的がユーザー側で充分に浸透、共有されているか	・プロジェクトマネジメントレビューやステアリングコミッティの開催あり/なし ・開催回数
(パッケージ導入の場合)現場ユーザーが作成したスプレッドシート等が現行の画面や帳票よりも重要な役割を担っているものがないか	・あり/なし
(パッケージ導入の場合)データ移行等の業務を継続するための計画が考慮されているか	・移行計画のマイルストーンが明確となっている/いない ・ユーザーの役割の明文化がされている/いない

リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	デモンストレーションを実施する	・実施回数
回避	FIT＆GAP 分析などの現状対比が可能な分析手法を導入する	・GAP の数
回避	非機能要求を具体的に文書化する	・非機能要求グレードのスコアリング
回避	要件定義書を基に現場ユーザーやステークホルダーが参加する業務ウォークスルーの実施タスクをプロジェクト計画に織り込む	・ウォークスルー回数

＜リスク事象ドライバー＞

システム部門が対象業務に関する要件を的確に提示できない

＜リスク事象＞

業務の全体や、細部にわたる要件や要望などが実は曖昧であるため、要件・要望の漏れや齟齬がテスト段階等で発覚する。

＜リスク事象ドライバー有無の把握＞

＜リスク事象ドライバーの予防策＞

このリスク事象を回避するには、要件定義段階で提示された現行画面や帳票などの形式的、定型的な情報を業務との関連で整理し、要件を詳細化する過程で機能と非機能の両面、および業務運用を踏まえた要求や要望を検討することが重要となる。

ベンダー側では、①レビュー時に開発コンセプトに沿った画面や帳票を具体的に検討する、②現状の FIT＆GAP 分析や導入パッケージのデモンストレーションを行い新システム導入による業務プロセスへの影響の有無

を確認する、③非機能要求に漏れが起きないように具体的な項目に落とし込んで共有するなど、ユーザー側から的確に情報を引き出し精査することがポイントとなる。また、ユーザー側では、業務に精通している有識者や経験者を加えて業務の網羅性が担保されていることを検証するための体制づくりや、現場ユーザー側が参加したレビューやウォークスルーの実施をプロジェクト計画に織り込むことがトラブルの発生を予防するポイントとなる。

＜解説＞

業務要件の網羅性という観点では、画面や帳票等の機能面と、実業務を考慮した運用面の両面から、要件の漏れ・抜けがないかをユーザー側とベンダー側双方で検証し合意を掲載することが必要である。

ユーザー側では、現状の画面や帳票等で具体的な形式的側面の業務仕様があることにより、実業務との関連での深掘りが行われないまま工程が進み、現場ユーザーやステークホルダーが関与し始めた時点で要件漏れや不備が発覚した場合には、大規模な仕様変更による手戻りが起こり、納期遅延などが発生する確率が高くなる。このため要件定義段階において適切な合意形成が求められる。

機能面については、パッケージ導入に限らず FIT＆GAP 等の分析を通じて機能と業務を洗い出し、現状業務との適合性や相違点をレビューし、ユーザー側とベンダー側で課題を共有することが欠かせない。また、非機能面については、非機能要求グレードを利用して画面や帳票を運用するユーザー側にとって理解しやすい具体的な項目や数値に落とし込み、プロジェクトとしての優先度や達成目標、条件を文書化することが有効である。

また、ユーザー側の現場担当者が要件定義に関与する度合が低い場合や、システム部門の担当者が当該業務に精通していない場合には、要件の詳細化の過程で充分に業務とシステムとの網羅性の検討が実施できていない可能性があるため、業務に精通している担当者や経験者の協力や評価を得る機会をプロジェクト計画に織り込み、網羅性を担保する施策も重要である。

6.16 設計と実業務の整合性が検証できていない

表 6-16 リスク事象ドライバー(16)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
設計と実業務の整合性が検証できていない	支援・管理プロセス
このリスク事象ドライバーによって顕在化するリスク事象	
業務に合わない情報システムになる(要件定義に漏れが発生する)	
リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的 / 定量的)
業務の実効性	・エンドユーザーを含めた業務フローのウォークスルーができている／いない
エンドユーザーによるユーザーインターフェイスの確認	・エンドユーザーが確認している／いない
外部システムインタフェイス	・外部システムとの全ての入出力仕様が明確である／でない部分がある
サブシステム間インタフェイス	・サブシステム間のインタフェイスが明確である／でない部分がある
キャパシティとパフォーマンス	・キャパシティとパフォーマンスの評価ができている／いない
ユーザーレビューの実効性の評価	・業務の企画者、実務担当者の両方の観点でレビュアーをアサインできたか ・事前資料回付、レビュー時間の確保など、レビューが機能する条件を整えられたか ・実効性のある指摘が得られたか ・誤字脱字以外の指摘数/ドキュメント頁数等 ・レビュー指摘事項記録票(にあたるもの)を作成しているか ・レビュー結果をドキュメントへの反映までトレースする仕組みがあるか ・レビュー結果の横展開をチェックする仕組みがあるか ・業務用語集はあるか ・誤解のない構文や用語を用いるためのチェックリストやルールはあるか
インタフェイスの確認	・出し側、受け側双方の関係者が、レビュアーとして参加しているか ・検証タイミング、検証方法が明らかになっているか

キャパシティとパフォーマンス	・利用者数、データ件数などの前提データは、平常時・ピーク時に分けて明示されているか ・利用者数、データ件数などの前提データは、5 年程度の将来予測を踏まえているか ・利用者数、データ件数などの前提データを算定するにあたって、ユーザーが参画しているか ・運行監視、障害時の運用、切替、縮退、リカバリの際の手順が文書化され、ユーザーが理解しているか

リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
軽減	ユーザーインターフェイス(画面、帳票)	・仕様が確定し文書化されている／いない
軽減	外部システムインタフェイス	・仕様が確定し文書化されている／いない
軽減	機能仕様の分割	・仕様が確定し文書化されている／いない
軽減	サブシステム間インタフェイス	・出し側、受け側双方の関係者が、レビュアーとして参加している／いない ・仕様が確定し文書化されている／いない
軽減	データベース、ファイル	・出し側、受け側双方の関係者が、レビュアーとして参加している／いない ・仕様が確定し文書化されている／いない
軽減	デッドロック対策	・仕様が確定し文書化されている／いない
軽減	キャパシティ／パフォーマンス	・仕様が確定し文書化されている／いない
軽減	障害対策／運用方針	・仕様が確定し文書化されている／いない
軽減	ユーザーレビューの実効性	・適切なレビュアーが参加している／いない ・レビュー時間が確保されている／いない

＜リスク事象ドライバー＞

実際の業務の理解、及び機能設計が不十分のままシステム構築を進める。

＜リスク事象＞

機能不足やインタフェイスが発生し、実業務が回らない情報システムができ上がる。

＜リスク事象ドライバー有無の把握＞

業務面と情報システム面の双方から点検（ウォークスルー）し、業務に対する機能の解釈が誤っていないかを確認する。

＜リスク事象ドライバーの予防策＞

機能項目を網羅するだけでなく、個別のスペックまでも明確にし、ユーザーとベンダーでの理解の齟齬がないようにする。

＜解説＞

同じ言葉でもユーザーとベンダーで解釈が異なることがある。例えば「入力フィールドは 100 まで」と決めた場合、100 は入るのか入らないのか、整数以外も認めるか認めないか、など曖昧さが残る。このような状態を放置したまま機能設計が完了していった場合、機能項目としては網羅されるが、機能不足や機能の誤り、インタフェイスでの齟齬が発生し手戻りが発生する。例示した内容は簡単なものだが、機能設計を曖昧にしてしまったことで要件との齟齬が甚だしくなり、設計を全面的にやり直した事例もある。

また、情報システムの機能に焦点が絞られ、情報システムの運用面での機能が考慮されず、トラブル時の障害解析に時間が取られ復旧に時間がかかったケースや、ハードウェア障害でディスク交換したが、ディスクの内容に機微な情報が含まれており情報セキュリティ上の問題が発覚したなどの事例もある。

業務を想定した機能設計や、情報システムの運用も含めた機能設計など、ユーザーとベンダーの相互認識が一致するような機能設計を行なう必要がある。

6.17 経営層によるプロジェクト運営の関与が十分でない

表 6-17 リスク事象ドライバー(17)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
経営層によるプロジェクト運営の関与が十分でない	組織的プロセス

このリスク事象ドライバーによって顕在化するリスク事象
要件定義を始めとする工程の節目や重要な意思決定場面で、優先順位やリソース配分に対する方向性が決まらないため、プロジェクトの意思決定が誤ったものとなる。

リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的 / 定量的)
プロジェクト憲章(プロジェクトを公式に認可する文書で経営やビジネスのニーズ・期待、PM 権限等を記載)に対する経営層の承認	・プロジェクト憲章を作成している/いない ・プロジェクト憲章が経営層に承認されている/いない
プロジェクト成功時の経営的意義の提示	・成功時に経営に対するインパクトが示されている/いない
経営層からプロジェクトに対する経営上の位置づけの提示	・経営上の位置づけが示している/いない ※経営上の位置づけ:経営戦略との関連や制約のこと 例1:(ユ)新規顧客開拓戦略の中核であり顧客の利便性実現を第一にする 例2:(ユ)コスト削減の一環として費用対効果を最大化する。 例3:(ベ)最重要案件とし人材を広く社内から集める。
経営的視点での優先順位と理由の提示	・経営的視点での QCD の優先順位が示されている/いない
経営層から関連組織への参画指示	・関連組織に向けたメッセージがある/ない
重要なタイミングにおける経営層の参加	・参加している/いない 重要なタイミング例:キックオフ時、ステコミ時
経営層の継続的関与	・プロジェクトに対し月次報告など報告要求がある/ない
プロジェクト計画の前提条件(プロジェクトで責任を負えないリスク)に関する経営層の承認、前提条件が変わった場合の意思決定ルールの承認	・前提条件および条件変更時の意思決定ルールが文書化され承認されている/いずれかがない

リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
軽減	プロジェクト憲章を作成し、経営層の承認を得る	・プロジェクト憲章の承認記録
回避	重要なタイミングにおける経営者の参加の重要性を説明し了承を得る	・出席承認記録
回避	プロジェクト成功時の経営的意義を文書化する	・経営者からのメッセージ文書の有無
回避	経営層の会社としての関わり方(位置づけ)を文書化する	・プロジェクトの位置づけを示した文書の有無

回避	関連組織への参画指示を文書化する	・経営者からのメッセージ文書の有無
回避	優先順位に関する経営層の考え方を文書化する	・QCD に関する優先順位を示した文書の有無
軽減	経営層への定期的あるいはフェーズごとの報告の必要性の了解を得る	・報告計画の文書の有無
回避	プロジェクト計画の前提条件を文書化し、経営層の承認を得る	・前提条件の文書の有無、承認記録

＜リスク事象ドライバー＞

経営層がプロジェクト運営に消極的な場合、様々なリスクを引き起こすが、これはプロジェクトに対する経営層のコミットメント不足である。このドライバーは、ユーザー側の経営層、ベンダー側の経営層どちらも当てはまる。

＜リスク事象＞

プロジェクト内で解決できない、相反する要求やリソース配分に対し、経営上の優先判断がされない、あるいは判断を IT 部門・プロジェクトに委ねることにより、プロジェクト内に思惑のずれが生じて迷走や軋轢となり、モチベーション低下・非効率・品質低下につながる。

＜リスク事象ドライバー有無の把握＞

プロジェクトの開始時、定常報告時に確認する。

＜リスク事象ドライバーの予防策＞

経営層からのプロジェクトおよびプロジェクトメンバーへの直接的メッセージは、口頭・文書の形式を問わず参加メンバーの意思統一・モチベーション形成に向けて非常に有効である。したがって、経営層からは、適切なタイミングでプロジェクトの経営上の位置づけや関与の方法を明示してもらうようにする。メッセージを有形の文書として残しておくことで、途中から参加するメンバーに対して、また、メンバー間の考えの相違が生じた際にも基本的スタンスとして拠り所にできるためリスク軽減に有効である。関連部署は必ずしもそのプロジェクトに積極的にならないこともあるため、関連部署に向けた経営層からの支援指示(要請)は実質的支援を得るために有効である。

＜解説＞

プロジェクトが直面するリスクには、プロジェクトチーム内部で解決できるものと解決できないものがある。経営層からのコミットメントは、プロジェクトが解決できない問題に対する経営的サポートを得られることの裏付けとなる。

例えば、顧客に開始を宣言した新サービスに向けてのシステム開発のプロジェクトの場合、サービス開始日には稼働していることが必須であり、会社の信用的観点からコスト制約や要求追加・変更への制約は相対的に低くなる。したがって、経営とあらかじめ優先順位に関する承認を得ておくことで、プロジェクト外の要因で遅延リスクが顕在化した場合、経営に対し追加的予算処置や関係部署のより一層の支援を得やすくなる。

あるいは別の例では、高信頼性を求められるシステム開発では、品質が最優先され、コスト・稼働時期の優先順位が下がる。この場合、稼働時期を早めることを求める要求に対して経営的判断として品質優先であることを主張できる。

プロジェクトは、その目的により QCD に優先順位が生じるのが通常である。要件定義以降は、変更要求や外部要因による計画変更のリスクが常に存在する。優先順位に関する合意を得ておくことで計画変更を抑止し、結果的に QCD 悪化のリスクを抑え込むことができる。

また、優先順位だけではなく、プロジェクトを計画した際に前提とした条件を明示し、経営と合意しておくことも重要である。例えば、当該プロジェクトにおけるキーマンが、必要な時期に確実に参加できることを前提に計画が組まれている場合、キーマンが参画できなくなるリスクはプロジェクトでコントロールできないため、経営にエスカレーションされるべきリスクとして経営と合意しておく必要がある。他の例では、複数の関連プロジェクトが同時並行に進んでいて、他プロジェクトの成果物（例えば、利用予定の技術開発)を前提としてもう一つのプロジェクトが計画されている場合、その前提を経営と合意しておくことで、他プロジェクトの情報が得やすくなり、課題発生時には経営的サポートを得られる可能性が高くなる。

6.18 経営層によるスコープ決定への関与が十分でない

表 6-18 リスク事象ドライバー(18)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
経営層によるスコープ決定への関与が十分でない		組織的プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
調整の不調による費用の膨張や要件定義の誤り		
リスク事象ドライバー有無の把握		
把握の観点		メトリクス(定性的 / 定量的)
プロジェクトの各目標達成(費用面、各種効果・KPI、品質)に関して、それぞれ責任者が定まっているか		・定まっている/いない
ビジネス要求、ステークホルダー要求、ソリューション要求の階層間の関連が整備されているか		・ビジネス要求、ステークホルダー要求、ソリューション要求の関連網羅率
ソリューション要件の変更がどのレベルの要求にまで影響を及ぼすか、把握されているか		・ビジネス要求、ステークホルダー要求、ソリューション要求の階層間の関連一覧が作成されている/いない
要件の変更時に、及ぼす影響に応じた責任者の承認を得るルールが存在するか		・存在する/しない
システム化の目的、優先順位は明らかとなっているか		・なっている/いない
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
軽減	ビジネス要求の達成責任者による要件承認	・責任者によるレビューおよび承認記録
軽減	要件の一覧に要求の種別(階層)を記載する	・要求種別の記載率
回避	要求の一覧に上位、下位要求との関連を明記	・ビジネス要求、ステークホルダー要求とソリューション要求との関連網羅率
軽減	要求の各階層間での整合性を確認する	・整合性確認率(チェック済みの割合)
軽減	要件の変更時に、及ぼす影響に応じた責任者の承認を得るルールを整備	・要件変更時のエスカレーションルールの定義の有無
軽減	ビジネス要求の充足率、達成率を設定することで、定例会議、報告における経営層の興味を喚起する	・ソリューション要求の実施率から導かれる、ビジネス要求充足率、達成率

＜リスク事象ドライバー＞

ユーザー側の経営がプロジェクトスコープに関して、初期の設定・承認のみに関与し、プロジェクトの遂行途上で発生する各種の変更や調整には関与しないケース。

＜リスク事象＞

直接的には、調整の不調による費用の膨張や要件定義の誤りであるが、最終的には、プロジェクトにより獲得したいビジネス要求が達成されないということになる。

※当リスク事象ドライバーおよび防止策については、BABOK（A Guide to the Business Analysis Body of Knowledge：ビジネスアナリシス知識体系ガイド）における要求区分の考え方をベースとする。

＜リスク事象ドライバーの有無の把握＞

経営が求めるビジネス要求を現場で把握できるか否か、ソリューション要求の変更がビジネス要求にどのような影響を及ぼすかを把握できていなければ、要件変更の経営へのインパクトが明確にならず、経営がプロジェクトスコープの変更について判断を下すことができない。したがって要求階層間の関連が明らかになっており、変更時のエスカレーションルールが整備されているか否かがリスク事象ドライバーの有無と関連する。

また、そもそもプロジェクトの目標達成の責任者が個別に設定されている場合は、目標達成のための要求間の調整や確認が行われ、リスク事象に発展しない可能性が高い。

＜リスク事象ドライバーの予防策＞

リスク事象ドライバーの予防策としては、経営のコミットを高めて、ビジネス要求が開発現場で置き去りにされることの防止が主となる。

したがって、ソリューション要求の変更がビジネス要求の変更に及ぼす影響を明らかにすること、また、それを如何に経営に認識してもらうかの工夫が、現場における当リスク事象ドライバーの予防策となる。

＜解説＞

BABOK では要求を、「ビジネス要求」「ステークホルダー要求」「ソリューション要求」「移行要求」の大きく四つに整理し、階層化して定義している。当リスク事象ドライバーは「経営によるプロジェクトへのコミット不在・不十分」がもたらす要求の階層間での不整合を想定している。

システム開発プロジェクトはユーザー側企業の 5 カ年計画などの戦略を実現させるための手段として、各事業部門にて構想されるケースが一般的である。また、採算よりも企業改革・業務改革を意識した大プロジェクトがトップダウンで行われ、各事業部門がそれぞれ計画を立案するケースもある。

いずれも、基本計画の立案（投入予算と、達成目標のコミット、リスク評価）までは経営は十分コミットするが、要件定義以降についてはほぼノータッチとなることが珍しくない。

例えば、大型のプロジェクトであればトッププライオリティとして財務的効果があるとしても、その他に顧客サービスやリスク回避、インフラ強化などの目的を含んでいる。これら個別のビジネス要求は、経営が意識する全体方針と関連するものとして各事業部門での企画フェーズまでは意識されている。

しかし、ビジネス要求は、現場での担当レベルでステークホルダー要求に、さらに IT プロジェクトとして PM 視点によってソリューション要求へとブレークダウンされていく段階で、列挙されたソリューション要求とビジネス要求の関連が見えにくくなる。このため、個別の要求の取り下げや変更が事業計画全体にどのような影響を与えるのか、開発の現場では判然としなくなることが多い。

その結果、経営がノータッチであると、現場の PM が要件を絞り込む必要がある際に、ビジネス要求への影響度の大小を判断できず、何が本当にプロジェクトにとって重要なのか、適正なジャッジができないリスクへとつながるのである。

こういった事態を防ぐためには、会社として何を選ぶのか、どの程度リスクを受容するのかを経営が適切に判断する必要があり、要件定義以降もスコープのマネジメントに参画してもらう必要がある。そして、経営が適正にプロジェクトにコミットできるよう、要件定義段階での PM レベルでの要求理解（ソリューション要求）と経営の要求理解（ビジネス要求）の間の関連付けを整備し、両者が共通言語で認識合わせできるようにする必要がある。こういった工夫なしに会議体を設定するだけでは、経営が参加しても効果的なマネジメントは行われないであろう。

また、ユーザー側のプロジェクト責任者は個々のビジネス要求の達成責任者を適正に定めてもらうよう経営に要求することも、適正なコミットを得るためには不可欠である。

6.19 経営層がパッケージ導入の意図・目的を明示していない

表 6-19 リスク事象ドライバー(19)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
経営層がパッケージ導入の意図・目的を明示していない		組織的プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
パッケージ導入でカスタマイズ仕様が膨張する		
リスク事象ドライバー有無の把握		
把握の観点		メトリクス(定性的 / 定量的)
パッケージの採用方針(導入意図や、現行業務維持方針など)について、プロジェクト企画書などの公式文書に記載しているか		・記載している/いない
カスタマイズに関する制約条件(コストや工数の上限など)について、プロジェクト企画書などの公式文書に記載しているか		・記載している/いない
カスタマイズの有無や影響について、経営層がレビュー・承認する仕組みがあるか		・仕組みがある/ない
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	パッケージの採用方針の、公式文書への記載	・経営層による、文書化された採用方針の承認
回避	カスタマイズに関する制約条件の、公式文書への記載	・経営層による、文書化された採用方針の承認
回避	経営層による、カスタマイズの有無や影響に対するレビュー	・経営層による、カスタマイズの有無や影響に対する承認

＜リスク事象ドライバー＞

ユーザー側の経営層がパッケージ導入を判断したが、その意図(「スクラッチ開発より低コスト」や「短期間での納入」など)や目的(パッケージに合わせた業務改善など)をプロジェクトに対して明確に示しておらず、プロジェクトを十分に制約していない。

＜リスク事象＞

パッケージ導入の意図や目的に対して、優先度の低い要求が混入する事で、当初の想定以上にカスタマイズ仕様が増加する可能性がある。

業務をパッケージの機能に合わせるのか、業務に合わせてカスタマイズするのかの判断は、経営層の判断に基づいている必要がある。

＜リスク事象ドライバー有無の把握＞

パッケージの採用方針やカスタマイズに関する制約が明文化されていること、また、経営層が「どこまで何をカスタマイズ」するのかをレビュー・承認する仕組みがあるかで把握する。

＜リスク事象ドライバーの予防策＞

まず、ユーザー側の経営層がパッケージ採用方針を明確に示し、その内容を文書化してプロジェクトのメンバーで共有することである。そして、採用方針を踏まえた上で、どこまで現行業務を維持するのか、どこまでカスタマイズを行うのかを判断し、経営層の承認を得る必要がある。

＜解説＞

ベンダーにパッケージの導入を依頼する場合、ユーザー側が得られるメリットの１つとして、スクラッチ開発より低コストで済むことや、短期間で導入できることが考えられる。

ユーザー側の経営層は、このようなメリットを考慮に入れて、パッケージの導入を決定しているはずである。メリットを享受するためには、なるべくカスタマイズ仕様を少なくして、パッケージの機能を活用することが求められる。

一方で、パッケージは汎用的な機能を提供するものであり、ユーザーの要求と比較すると、機能が不足していたり、振る舞いが異なったりしていることもある。場合によっては、パッケージの機能を活かすために、業務内容を変更しないといけなくなる場合があるかも知れない。

パッケージは通常、パラメタによるカスタマイズの機能を持っている。ユーザーごとに要求が異なることが想定される機能については、追加の開発をしなくても、パラメタの設定によって機能や振る舞いを調整することができるようになっている。しかし、パラメタ設定だけでは、要求を完全には満たせない場合、そのことが追加開発の要求につながる場合もある。

パッケージ導入で追加開発を行う場合、その量や内容によっては、計画されたコストや工期に大きく影響することがあるため、パッケージの機能や特性を十分に理解した上で、低コストや短納期というメリットとのトレードオフを考慮する必要がある。

ユーザー部門では、現行業務手続きを変えることに抵抗があり、現行業務を前提にカスタマイズをしがちである。

これを防ぐには、経営が明確な方針を示す必要がある。

6.20 ステークホルダー間の力関係のアンバランスである

表 6-20 リスク事象ドライバー(20)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
ステークホルダー間の力関係のアンバランスである		組織的プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
全体最適が実現しないことによる QCD の乱れ		
リスク事象ドライバー有無の把握		
把握の観点	メトリクス(定性的 / 定量的)	
交渉の余地のない契約・取り決め事項がある	・存在する/存在しない	
変更管理時に工数・コスト・負担先をジャッジする仕組みがない	・ある/ない	
納期折衝・スコープ調整の余地がない	・ある/ない	
要件定義を含めた一括契約である	・該当する/しない	
該当プロジェクト以外の年間取引高が大きい	・顧客売上高/受託者年商	
ステークホルダー間で資本関係や縁故関係がある	・関係数/ステークホルダー数	
プロジェクトと連動する営業上の目的や利益がある	・ある/ない	
異なる役割を持つステークホルダー間で指導・統括・評価関係がある	・関係数/ステークホルダー数	
ステークホルダー間で過去にクレーム・トラブル・係争がある	・係争等の数	
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	①契約上要件定義を分離する。	・要件定義分離有無
回避	②キックオフまでに、コストとリスクのジャッジを含む変更管理を合意する。	・コスト・リスク負担を明確にした変更管理票有無
回避	③フェーズ毎に制約事項・スコープ・納期調整のイベントを設置する。	・制約事項・スコープ・調整イベント数/プロジェクト全期間
回避	④異動発令などで、ライン等のしがらみを切っておく。	・関係数
回避	⑤交渉・対決を含む役割を定義・合意しておく。	・交渉・調整・対決にあたる文言を明記した役割数/全役割数
回避	⑥利益相反的な無理のある役割定義をしない。	・マッチポンプ数/組織図総箱数 (ex.無理な役割定義の例:オーナー兼特定部門長、営業部門担役兼リスク管理部門担役、課長兼 PM)

回避	⑦発注先を分散するなど、集中を回避する。	・最大個社発注額/発注総額
軽減	①プロジェクト外の目的やその優先順位を可視化する。	・目的・優先順位記載数
軽減	②意思決定プロセスや調整プロセスを可視化する。	・意思決定プロセス記載文字数
軽減	③経営レベルの高次監視・決定・調整機関を設ける。	・監視レベル(①取締役会、②社長、③専務、④役員、⑤役員未満)

＜リスク事象ドライバー＞

部門(ステークホルダー)間の力関係がアンバランスで、合理的な決定や役割分担の最適化を阻害する。

＜リスク事象＞

全体最適でない決定や、適切でない役割分担により、手戻りや品質未達、遅延、コスト超過が発生する。

＜リスク事象ドライバーの有無の把握＞

＜リスク事象ドライバーの予防策＞

プロジェクトの目標達成以外の思惑や利益、面子、対人関係などの影響を極力排除し、不合理な債権・債務関係の発生を抑制して、健全なプロジェクト運営の基盤を整えることである。

＜解説＞

わが国ではとかく、「作業の詳細が明確になる前に総額での見積りを求める」「一度決めた予算額は、いかなる理由があろうと変更することができない」といった不条理な慣行があり、「委託側が強い」という力関係のアンバランスから、プロジェクトが迷走しがちである。このことは、必ずしも委託者側の利益にはならない。品質にしわ寄せが来れば、最もひどいダメージを受けるのはICTシステムの利用者である。そもそも受託者も営利事業者である以上、見積額に多めのリスクを組み込んだり、外注先に転嫁したり、継続契約・随意契約で利益を取り戻しにかかったりと、不明朗な(ユーザーにとっても見えづらい)回避策を講じることになる。

また、ビジネスを牽引する営業部門、予算配分を統括する経理部門、経営直轄の監査部門や品質管理部門など、いわゆる「声の大きい」部門や、叩き上げのベテラン、上位職経験者などの権威者が存在し、プロジェクトの決定に大きな影響を及ぼすこともある。事業におけるプロジェクトの重要性や課題の優先順位と、「声の大きさ」が一致していればまだ良いが、そうでない場合、プロジェクトにとって本質的に重要な意見や知見を、それらの「力の強い」ステークホルダーが圧殺する構図も発生し得る。

この他、社内プロジェクト等で多いのは、(元・現)上司・部下などのステークホルダー間の関係が、プロジェクト内の役割分担とは無関係に結論に影響を及ぼしたり、職務上の立場とプロジェクト内の立場の混同によって意思決定プロセスが機能不全を起こしたりしているケースである。

プロジェクトの全体最適を目的とする健全な判断・決定のためにはゼロベースな見方が必要で、そのためには独立性を担保できるような環境や、ステークホルダー間の関係が必要である。

要求や個別利益のコンフリクトが恒常的に発生するプロジェクト運営において、ステークホルダー間で目的を共有して全体最適の結論を導き、相互にプロセスを可視化して破綻を回避するためには、ステークホルダー間の権力勾配を健全に維持する必要がある。

6.21 高次の調整・決定機関が機能していない

表 6-21 リスク事象ドライバー(21)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
高次の調整・決定機関が機能していない	組織的プロセス

このリスク事象ドライバーによって顕在化するリスク事象
重要事項の決定が遅れたり、調整できなかったりする

リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的 / 定量的)
ステコミにあたる機構(以下、ステコミ)が存在しない	−
ステコミの開催要件(定期開催サイクル、開催条件など)が定義されていない	−
ステコミの開催頻度が低い	・プロジェクト期間中開催回数、月間開催回数
ステコミが単なる報告会になっている	・メンバーの質問回数/回、決定事項数/回
ステコミメンバーが決定権限を持っていない	−
ステコミの議事録が作成されていない	−
ステコミの資料が定義されていない	−
ステコミでの結論がひっくり返ることが容認されている	・変更決定数
ステコミの議題が不足しているなど、不適切である	・審議網羅率(スコープ決定、リスク評価、工程完了、納期変更、コスト変動…)

リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	①ステコミ設置について、提案(稟議)段階で合意しておく。	・ステコミ提案有無
回避	②ステコミの開催頻度やメンバーを提案(稟議)の前提としておく。	・ステコミ開催頻度(回/プロジェクト期間)
回避	③キックオフ時に、会議体・議題・資料などを定義しておく。	・会議体定義充足率(頻度、議題、資料、出席者、議事録承認手続きのうち、合意形成されているものの数)
軽減	④ステコミ不在時のリスクを提案(稟議)前に説明・共有しておく。	・リスク管理台帳の有無/記載の有無
軽減	⑤プロジェクトチャーターなどのプロジェクト文書をオーナー名で発行する。	・オーナー発行のプロジェクト憲章有無
軽減	⑥利益相反する役割の兼務などの無理のある役割定義をしない。	・無理な役割数/組織図総箱数(ex.利益相反の例:オーナー兼特定部門長、営業部門担役兼リスク管理部門担役、課長兼 PM)

＜リスク事象ドライバー＞

舵取り委員会が存在しない、または機能しない。

＜リスク事象＞

スコープ確定や計画変更などの重要事項の決定ができない、遅れる、部門間調整ができずに迷走する、手戻りが起こるなど、プロジェクトの進行阻害。

＜リスク事象ドライバーの有無の把握＞

＜リスク事象ドライバーの予防策＞

ステコミの設置を提案・合意すること。名称にこだわらず、利害や要求を調整してコンフリクトを解消するための何らかの機構・仕組みを実現することが第一である。

機構として実現しなければ、プロジェクト期間を通じて、その役割を担うべきステークホルダーを明確化し、そのステークホルダーとのコンタクトチャネルを開設しておく。また、現況・課題を可視化して、ステークホルダーと共有しやすくしておく。「現場の声の大きい人」の判断ではなく、プロジェクトの公式判断を固定し、覆りにくくするための方策を講じる。

＜解説＞

ソフトウェア開発には、様々な利害関係者が絡む。それぞれのステークホルダーの関心事や知識・経験の範囲は偏っているのが普通である。例えば、対象業務に詳しい業務担当者は実装技術やソフトウェア開発、プロジェクト管理の経験や知識に乏しいことが多い。PM や開発担当者は業務知識や意思決定プロセス、業務部門間の力関係にうとい場合がある。また、経営層にとってコスト削減やプロセス合理化・改革が重要であっても、現業部門は慣れた現行業務の維持に固執する可能性もあるため、経理部門は開発コストに目を光らせ、その削減を求める。コンプラ部門は事務効率よりも統制を重視し、事務部門は正確性や利便性、効率性を重んじる。また、リスク管理部門はトレーサビリティーとセキュリティを重視し、保守担当者は開発効率よりも保守効率を優先する。その結果、プロジェクトは期間・コスト・技術・リソースによる制約を受ける。

したがって、プロジェクトの定常状態として、要求のコンフリクト制御や、優先順位の設定、方針の維持と転換、スコープ調整や納期調整が必要になる。換言すれば、プロジェクトは時々刻々、何かを「あきらめる」ためのデシジョンを必要とする。実現するためのデシジョンは比較的たやすいが、「あきらめる」ためのデシジョンは困難であり、会社間、部門間、また、個人間の利害を調整する仕組みが必要である。

往々にして、高次意思決定者は方針決定にのみ参画し、「あとは司司で」の丸投げに陥りやすい。しかし、痛みを伴うデシジョンこそ、高次意思決定者が担うべきものである。したがって、しかるべきオーナーや意思決定者が、共有された情報に基づいて英断を下す機構が必須になってくる。

プロジェクトの最高意思決定機関であるステアリング・コミッティ(舵取り委員会)は、常に設置が実現するものではないにしても、少なくともプロジェクト関係者はその意義を理解した上で設置に向けた努力をし、何らかの理由で設置できないならば、不在によって発現する可能性に対して備えなければならない。

存在するドライバーの数やメトリクスによって、このリスク事象の発現確率はある程度判断できる。対象業務や規模・期間、ステークホルダーの数など、他のリスク要素と照らし合わせて、容認し得る確率にとどめる対策が必要である。

6.22 十分なコミュニケーションが取れていない

表 6-22 リスク事象ドライバー(22)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
十分なコミュニケーションが取れていない	組織的プロセス
このリスク事象ドライバーによって顕在化するリスク事象	
作業効率の悪化により工程の進捗が遅れる	
リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的 / 定量的)
プロジェクト計画書にコミュニケーション計画が定義されているか	・プロジェクト計画書のコミュニケーション計画の有無
コミュニケーション計画として以下のようなものが明確か ・課題管理とエスカレーションルール ・会議体定義(定例会議、臨時会議等) ・情報共有手段(メール・掲示板・ファイル共有) ・複数拠点における会議支援システム ・オフラインコミュニケーション(懇親会や階層を越えた意見交換会など) ・フェイスツーフェイス(遠隔拠点では定期的な顔合わせの場を設定)	・コミュニケーション計画が具体的である/でない

リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
軽減	プロジェクト計画書にコミュニケーションに関する項目を文書化する	・プロジェクト計画書の該当項目の有無
軽減	課題管理表の更新を確認する	・課題管理表で長期間更新の無い課題の有無
軽減	プロジェクトメンバーとの直接対話の機会を作る	・管理層からの報告との差異の有無
軽減	コミュニケーション支援ツールを活用する	・コミュニケーション支援ツールの利用状況の確認
軽減	チーム横断の活動組織(例:開発環境向上委員会、レク委)を作る	・チーム横断組織活動の活動公開の有無
軽減	他者・他チーム課題解決への貢献を評価する仕組みを作る(貢献された方が、した方にThank You カードを発行するなど)	・仕組みの実施状況の確認

＜リスク事象ドライバー＞

プロジェクト内のコミュニケーション不足、あるいは一方通行である。

＜リスク事象＞

プロジェクト内のメンバー間あるいはチーム間でコミュニケーションが不足することで、お互いの理解に齟齬が生じていることに気付かなかったり、問題・課題が認識されず放置され、エスカレーションされなかったりすることで、遅延・非効率・モチベーション低下につながる。

＜リスク事象ドライバー有無の把握＞

＜リスク事象ドライバーの予防策＞

コミュニケーションについては、「成り行き」に任せるのではなく、効率的・効果的なコミュニケーションが発生するようなツールやルールを構築し、実際にそれらが機能していることをモニタリングするなどの仕組みが必要である。

＜解説＞

プロジェクトは、本来ゴールに向かって方向・価値観が共有され、プロジェクトに影響の出る課題は協力・協調して解決することが望ましい。しかし、ユーザーとベンダーの間を典型として、ベンダー間・ベンダー内でもチーム間や関係組織の関係で協力・協調関係が阻害されることがある。阻害要因としては、利害の対立や無関心、一方通行が起きた場合のコミュニケーション不全が考えられる。コミュニケーションを機能的・効率的に行うための公式なルートを先ずは整備し適正に運用することで、組織間・メンバー間のコミュニケーションの敷居を低くするとともに、阻害要因が生じていないかの非公式な情報収集が不可欠である。

6.23 業務用語が共有されていない

表 6-23 リスク事象ドライバー(23)記述内容

リスク事象ドライバー		
リスク事象ドライバーの内容		区分
業務用語が共有されていない		組織的プロセス
このリスク事象ドライバーによって顕在化するリスク事象		
要件定義・設計上の誤謬による手戻りや品質問題が発生する。		
リスク事象ドライバー有無の把握		
把握の観点		メトリクス(定性的 / 定量的)
業務用語辞書の不存在		・存在する/しない
業務用語辞書網羅率		・項目数/開発工数
DB 項目説明書の不存在		・存在する/しない
DB 項目説明書網羅率		・定義項目数/全項目数
パッケージ用語のみによるドキュメント		・注釈(顧客用語による説明など)がある/ない
外国起源パッケージ、外国起源業務などの独自用語要素		・要素数、カスタマイズ不可独自用語数
項目定義・整理タスクの不存在		・存在する/しない
同名異議、異名同義語の存在		・同名異議語数異名同義後(サンプルチェック)
対象業務の特異性		・開発者同一業務経験回数、業務継続年数、従事者数/全産業従事者数
業界の歴史的経緯による難解・特殊用語の存在		・業界継続年数
規程等業務定義書の不存在		・存在する/しない
規程等業務定義書の網羅率		・ページ数/開発工数
ドキュメント独りよがり度合い		・略語・カタカナ語使用率/ドキュメント
リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
回避	①網羅性の充分な業務用語辞書を作成する。	・搭載用語数 ・ユーザー工数/辞書作成全工数
回避	②網羅性の充分な DB 項目定義書を作成する。	・搭載項目数 ・ユーザー工数/辞書作成全工数
回避	③ドキュメントをテーラリングする。	・パッケージドキュメント流用率 ・ドキュメント加筆率
回避	④思想や用語の合わないパッケージを採用しない。	・パッケージ選定基準有無 ・選定基準中の用語評価有無
回避	⑤項目定義・整理のタスクを計画する。	・項目定義・整理工数/要件定義工数
軽減	①業務説明会の実施。	・業務説明会時間/開発工数

軽減	②パッケージ/ソリューション説明会の実施。	・(パッケージ・ソリューション説明会時間)/開発工数
軽減	③職場見学・実習等の実施。	・見学等述べ時間/開発工数
軽減	④有識者、経験者のアサイン	・業務経験者比率

＜リスク事象ドライバー＞

業務用語・システム用語などの共有が不足する。

＜リスク事象＞

要件定義・設計上の誤謬発覚が遅れ、手戻りや品質未達、遅延、コスト超過が発生する。

＜リスク事象ドライバーの有無の把握＞

＜リスク事象ドライバーの予防策＞

業務用語辞書、DB 項目定義書などの共有用ドキュメントを準備して、認識齟齬を回避する。その際、業務を知悉したユーザーが多く関与することが望ましい。ユーザーに通じにくいシステム用語や、ソリューションに特化した用語を定義・説明なしに使用することは避け、相互に不明点・曖昧点を問いただすようにする。用語の定義・整理のためのタスクを設け、「支払額」「取引額」「価格」など、差異が分かりにくい項目や、「マルユウ」「ATM」などの略語、「ロス(給付金支払い)」「ガル(上開き荷台)」といった外来語、「ケイソン(鮭鱒)」「イゴン(遺言)」といった特殊読みなど、業界・業務特有の用語を整理する。

＜解説＞

業務知識の共有の必要性は、開発時によく話題になる。

開発担当者が、業務の流れや関連法規などについて、要件定義時によくヒアリングして把握する必要があることは論を俟たないが、その際に「言葉の定義」を曖昧にすると、せっかくの詳細なヒアリングが的外れなものになってしまいかねない。

用語の共有化の必要性は相互に言えることであり、開発担当者は開発担当者で、パッケージやソリューション、機器に特有な用語などを、よく説明したり、顧客に理解できる言葉に置き換えたりして、齟齬が出ないようにする必要がある。

業界・業務によっては、日常語と同じ言葉を、異なる意味で使用したり、独自の略語・読み・転用などを行っている場合があるので注意が必要である。用語辞書や解説書などは、当該用語に詳しい担当者(業務用語なら業務担当者、システム用語なら開発担当者)が主導して整理するのが基本であるが、用語に慣れすぎていて、その用法の特殊性や誤解の可能性を認識できなくなっている場合もある。したがって、「わかったつもり」での、聞き流し・読み流しをせず、相互に確認しあって辞書や解説書を充実させていくことが望ましい。辞書、解説書については、システム部門内にとどまらず、エンドユーザー部門などユーザー企業内で広く共有し、ブラッシュアップすることも重要である。

6.24 業務知識が不足している

表 6-24 リスク事象ドライバー(24)記述内容

リスク事象ドライバー	
リスク事象ドライバーの内容	区分
業務知識が不足している	組織的プロセス
このリスク事象ドライバーによって顕在化するリスク事象	
仕様変更、仕様追加の多発により、サービス開始の遅れや品質の極端な悪化を招く	
リスク事象ドライバー有無の把握	
把握の観点	メトリクス(定性的 / 定量的)
＜ユーザー側＞	
ユーザー側キーマンの確認	・企画キーマン、業務キーマン、現場リーダが明確である／ない
ユーザー側の体制	・要件定義時に企画キーマン、業務キーマン、現場リーダが参加している／いない
ユーザー側のレビュー時間の確保	・レビュー工数の比率(目標%)
新業務の目標イメージが明確である	・基本計画(要件定義着手前の業務側企画書)が明文化されている／いない
新業務の利用イメージ・サービスレベルが明確である	・利用者区分・利用者数・稼働時間帯・業務ピークの有無・処理量(トランザクション数、頻度)
関連業務が明確である	・関連業務とのデータのI／O定義、頻度、項目の意味が明確である／ない
関連法規・外的制約が明確である	・関連法規や外部制約に関する記述がある／記述がない
新業務サイクルが明確である	・日次、週次、月次、四半期、年次等の業務サイクルが明確である／ない
現行の業務フローが明確である	・ノーマル系以外の取扱いが明確である／ない
＜ベンダー側＞	
ベンダー側の体制	・同一業種・業態の開発経験者がいる／いない
ベンダー側の体制	・現行システムの維持管理者あるいは開発経験者がいる／いない
現行業務調査が十分である	・現行業務を現場レベルで調査している／いない
新業務のスコープ把握	・データ一覧・機能一覧・画面一覧・帳票一覧・バッチ一覧が作成され、承認されている／いない
新業務処理の把握精度	・ノーマルケース以外の例外処理が記述され、承認されている／いない
機能の把握精度	・データや機能定義に推定や想定の項目がない／ある
＜共通＞	
移行の計画に係る条件が明確である	・移行方法と実現可能性が明確である／ない
教育のイメージが明確である	・教育の方法と実現方法が明確である／ない

システム停止時の業務取扱の把握	・システム停止時の業務取扱（業務コンティンジェンシープラン）が検討されている／いない

リスク事象ドライバーの予防策		
対応区分	予防策の内容	予防策の有効性を確認するメトリクス
軽減	ユーザ・ベンダーの体制表を確認する	・体制と期待される役割・経験の記述の有無
回避	基本計画を確認する	・基本計画書の有無
回避	現行業務分析を行う	・現行業務分析書の有無
回避	要件定義書を確認する	・要件定義書の有無と必要事項の検討結果の有無
回避	サービスレベル定義書を確認する	・サービスレベル定義書の有無
回避	関連業務定義書を確認する	・関連業務定義書の有無
回避	移行、教育、コンティンジェンシープランを確認する	・関連定義の有無
回避	要件定義の十分性をユーザ・ベンダー相互に確認する	・要件定義完了時のレビューの有無 ・レビュー観点表の有無

＜リスク事象ドライバー＞

業務知識の不足により、業務ニーズを必要・十分に把握できず、要件定義が不十分に終わる。このリスクはユーザー側・ベンダー側の双方の関係において発生する。

＜リスク事象＞

要件定義以降の工程で仕様の変更・追加が多発した場合、コスト増大、納期遅延、品質低下、もしくは”使えない”システムとなる。

＜リスク事象ドライバー有無の把握＞

＜リスク事象ドライバーの予防策＞

ベンダーは、開発のみを請け負う場合でも、プロジェクトの前段となる基本計画書を確認し、プロジェクトの背景やユーザーの真に希望することを理解する努力が必要である。また、当該業務システムだけでなく、そのシステムとデータ連携する前後のシステム、業務側で連携される前後の業務についてもインタフェイスや連携のタイミング、例外的扱いなどの確認は、ベンダーだけでなくユーザーにとっても重要である。さらに、システム運用に関するニーズ・要求について、サービスレベル定義書等を用いて要件定義の早い段階でインプットすることで、「実用性」を重視した要件定義を行うことができる。

また、現行業務はユーザーが提供すべきという考えもあるが、ユーザーが現行業務を熟知しているとは限らないため、どのように提供すべきか不慣れなことも多い。したがって、ベンダーは現行業務分析を積極的に行うことがユーザーが言うところの「業務知識」獲得に有効である。

＜解説＞

業務知識は、ベンダー側だけに求められるものではない。システム化対象の業務イメージをユーザー側・ベンダー側が等しく持たなければ、開発着手後に要件の追加・変更が発生し、手戻り・工数の増加・開発期間への無理な皺寄せ・品質の低下・コスト増加を招き、最悪の場合、使えないシステムとなる可能性がある。

業務知識は、一般に、量・質的にもユーザー側に多くベンダー側に少ないため、ユーザー側は積極的に情報提供し、ベンダー側も委細漏らさず吸収すべきである。しかし、多くの場合、ユーザー側は“ベンダー側が何を知っていて、何を知らないか”、あるいは“何を伝えるべきか”を把握していないため、ベンダーがユーザーからの情報提供を待つ状況になると、ベンダー側の情報不足を生み出すことになる。

したがって、ベンダー側は、過去の経験に基づき、新業務のイメージづくりの過程で上手に業務知識・業務要件を引き出す工夫をする必要がある。特に、現場レベルでの例外処理や現行のシステムに乗っていない処理は、ユーザー側でも気付いていないことが多く、必ず現場レベルでの確認が必要である。

また、業務改革を伴う開発の場合、ユーザーはベンダーに対し同業他社での開発経験や類似開発の経験を期待している。しかし、多くの場合、ベンダーの経験に従っていると、開発が進むにつれそのままでは適用できないことが明らかになり、ユーザーからするとベンダーに対し「（当社の）業務知識不足」を指摘することになる。このことを回避するために、ベンダーは経験上の業務を適用する際に、業務側がなすべきこと、あるいはユーザー個社の特性・特徴を極力早い段階で認識しておくことが必要である。

付録

A） 標準リスクモデルとは

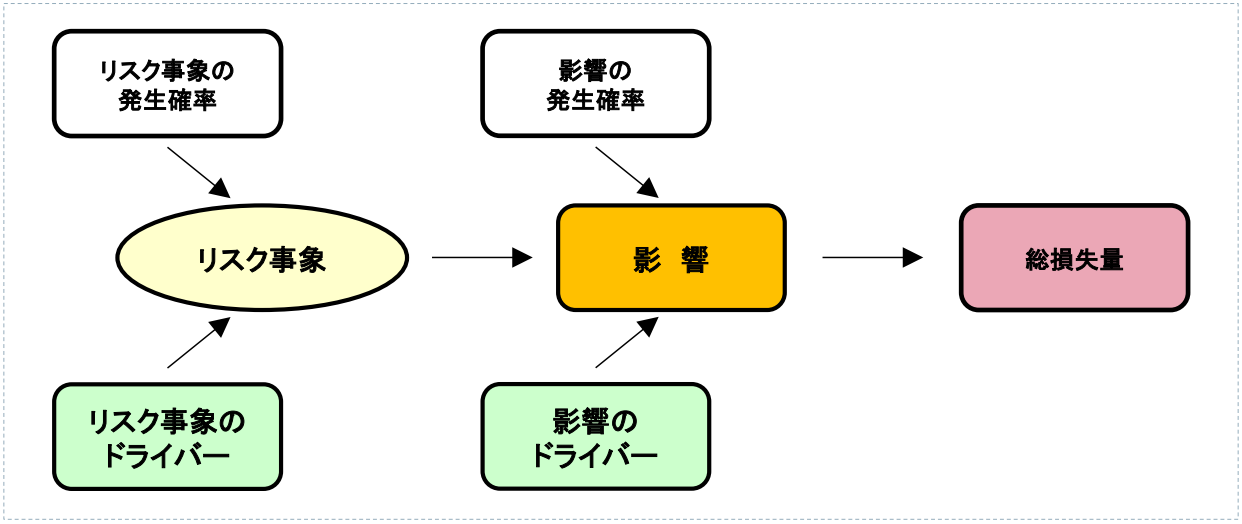


図 A−1　標準リスクモデル

出典：プレストン・G・スミス、ガイ・M・メリット（澤田美樹子訳）『実践・リスクマネジメント』生産性出版（2003）

「実践・リスクマネジメント」の標準リスクモデルは、以下の7つの構成要素から成り立つ。

- **リスク事象**：損失を引き起こす出来事
- **影響**：リスク事象が発生した結果として、生じる可能性のある潜在的な損失
- **リスク事象ドライバー**：プロジェクト環境の中に存在し、リスク事象の発生を導くもの
- **影響のドライバー**：影響の発生につながる、プロジェクト環境中に存在しているもの
- **リスク事象の発生確率**：リスク事象が発生する見込み
- **影響の発生確率**：リスク事象の条件下における、影響の起こる見込み
- **総損出量**：リスク事象が発生した場合に生じる損失の大きさで、日数または金額で表現

7つの要素に分けることによって下記の 2 点を実現している

・相互の関連が理解しやすい

リスク事象に関連する（リスク事象ドライバー）、（リスク事象の発生確率）については、リスク事象ドライバーを変更するとリスク事象の発生確率を下げることができる。

同様に、影響に関連する（リスク事象）、（影響のドライバー）、（影響の発生確率）については、リスク事象の発生を避けられない場合でも、影響のドライバーを変更すれば総損失量を下げることができるなど相互の関連が理解しやすくなっている。

・相互の関連のうち、特に原因と結果の関係が明確化されている

原因（リスク事象）と結果（影響、総損失量）の関係を明確化している。リスク事象は影響と総損失量の原因（ドライバー）となる要素である。これは、リスク予防に関連して効果的なリスクマネジメントは先手で行われるという概念を強調している。原因（リスク事象）を取り除けば結果（影響）は起こらなくなる。

また、原因（リスク事象）と結果（影響）を分け、因果関係を明確にし、それぞれを引き起こす要因（ドライバー）を識別することで、「リスク事象」に結びつかない課題などが混入しにくくなるという利点を持つ。

B）リスク事象の連続構造

B-1 リスク事象の連鎖構造

リスク事象には、あるリスク事象がより上位の異なるリスク事象に対するリスク事象ドライバーに相当するという連鎖的な構造がある。

リスク事象は、2.2.1 項で述べたとおり、「発生する可能性のある事象・出来事」である。一方、リスク事象ドライバーは、2.2.2 項で述べたとおり、「プロジェクトにおいて存在する事象・要因」である。即ち、リスク事象はまだ発生していない事象であり、リスク事象ドライバーはすでに顕在化した事象である。

しかし、ひとたびリスク事象が顕在化すると、より上位の異なるリスク事象に対するリスク事象ドライバーになる場合がある。この関係を図 B-1 に示す。図 B-1 では、実線で示したリスク事象が、破線で示したより上位のリスク事象に対するリスク事象ドライバーになっている様子を表している。

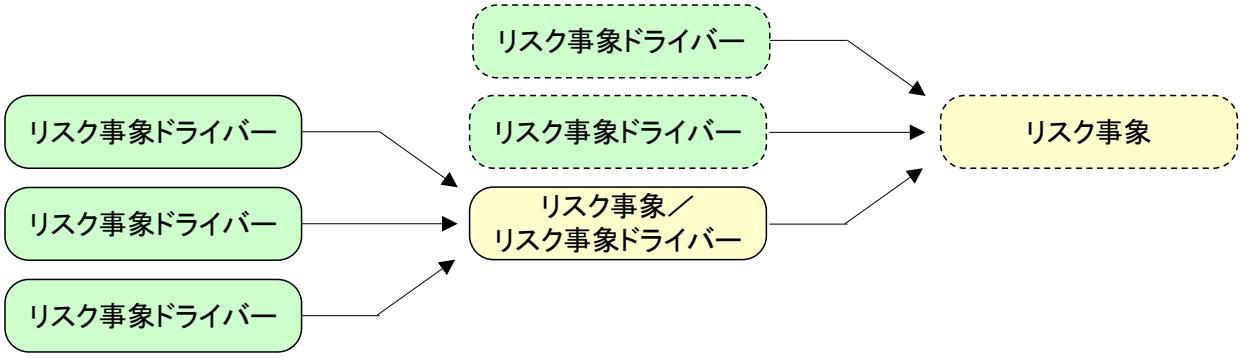


図 B-1 リスク事象とリスク事象ドライバーの関係

B-2 リスク事象の把握の観点

リスク事象の連鎖構造において、どの層のリスク事象に着目するかによって、対象とするリスク事象ドライバーの層は異なる。例えば、図 B-1 において、図の右端にあるリスク事象に着目する場合と、図の中央にあるリスク事象に着目する場合では、リスク事象の層が異なるため、それぞれのリスク事象の原因となるリスク事象ドライバーの層も異なる。異なる層にあるリスク事象ドライバーを対象として扱うと、リスク事象ドライバーの層にばらつきが生じ、誰が、どのような予防策をとるのかが分かりにくくなる。

そのようなことを避けるために、リスク事象とリスク事象ドライバーを把握するにあたって、組織内における階層の視点を明確にすると良い。例えば、リスク事象はプロジェクト管理者の視点で把握し、リスク事象ドライバー（とその予防策）はプロジェクトのメンバーの視点で把握する。特に、メンバーの視点でリスク事象ドライバーとその予防策を把握すると、プロジェクトの直接的な業務に関わるメンバーが、それぞれの役割の範囲でリスク事象ドライバーの有無を把握したり、回避策や予防策の実効性を確認したりできるようになる。対象となるリスク事象について、メンバー全員が予防的措置を講じられることになり、リスク事象の発現を高い確率で回避できると期待できる。

C) リスク連鎖における予防策の考え方

(1)リスクとリスク事象ドライバー

リスクは、その要因となる幾つかのドライバーが引き金となって顕在化確率が高まる。例えば下図で、飲酒が必ずしも寝坊を引き起こすとは限らないが、寝坊するリスクを高める要因の1つだとは言える。

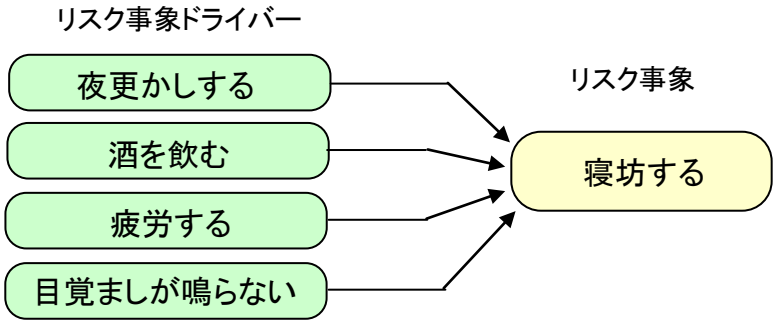


図 C-1 リスク事象(寝坊する)のリスク事象ドライバー

ここで、リスク事象ドライバーが多いほどリスクが顕在化する確率が高まると仮定するならば、逆にリスク事象ドライバーの存在を減らせばリスクの顕在化確率を低くできるとも言うことができる。

実際には個々のリスク事象ドライバーの性質や影響度などがあると思われるが、ここでは話を簡単にするため、それらの要素は同等であると仮定する。

(2)リスクの連鎖

前節ではリスク事象ドライバーという言葉を使ったが、このリスク事象ドライバーそのものもリスク事象の1つに他ならない。例えば、「目覚ましが鳴らない」というリスク事象ドライバーをリスク事象だと考えると、その顕在化確率を高めるリスク事象ドライバーが幾つか考えられる。即ち、リスクの顕在化に影響するリスク事象ドライバーを次々と遡ることができるのである。

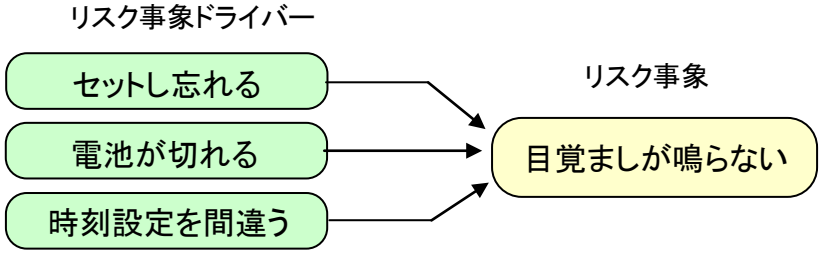


図 C-2 リスク事象(目覚ましが鳴らない)のリスク事象ドライバー

さらに言えば、下図のように、「寝坊する」というリスク事象も別のリスク事象に対するリスク事象ドライバーになりうる。

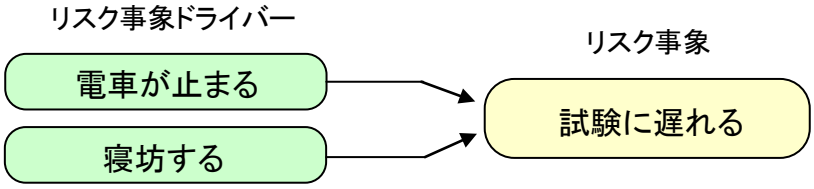


図 C-3 リスク事象ドライバー(寝坊する)のリスク事象

このように、リスク事象は、それが別のリスク事象のドライバーとなって連鎖していると考えることができる。

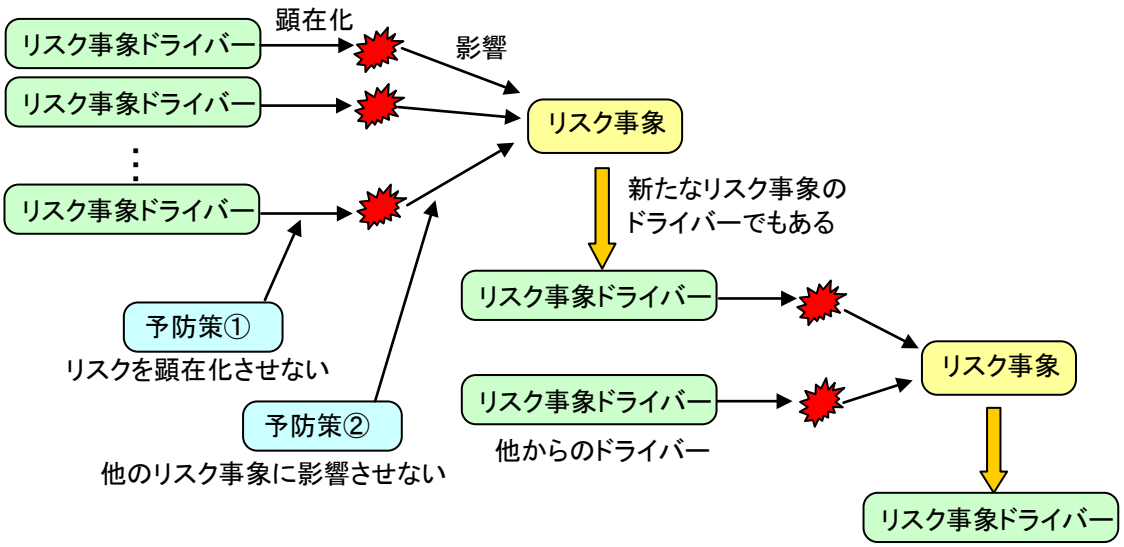


図 C-4 リスク事象の連鎖

通常、リスクに対する予防策は、その顕在化を回避あるいは軽減するものだが、リスクの連鎖における予防策には、これに加え、顕在化の影響が他のリスク事象に及ぶことを回避、あるいは軽減するための予防策があることを付け加えておく。

(3)対応すべきリスクレベル

一般的に、連鎖の初期段階に遡るほどリスクは軽く予防しやすいが、本来予防したいリスク事象からは離れ、対処すべき数も多くなる。

先の例では、「寝坊する」というリスク事象よりも、むしろ、その先の「試験に遅れる」ということが本来予防したいリスク事象だと考えられる。それに対し、「目覚まし時計の電池が切れないように買っておく」という防止策を考えたとする。

これは、遡った先にあるドライバーの1つの防止策ではあるが、「試験に遅れる」というリスク事象に対する予防策としては離れ過ぎており、このレベルで対策を1つ1つ考えることが現実的だとも思われない。

(4)システム開発におけるリスクの連鎖

ここで、システム開発プロジェクトにおけるリスクの連鎖を考えてみよう。

この場合、究極のリスク事象は「プロジェクトが失敗する」というものだが、そこに至るリスクの連鎖は相当なものがある。

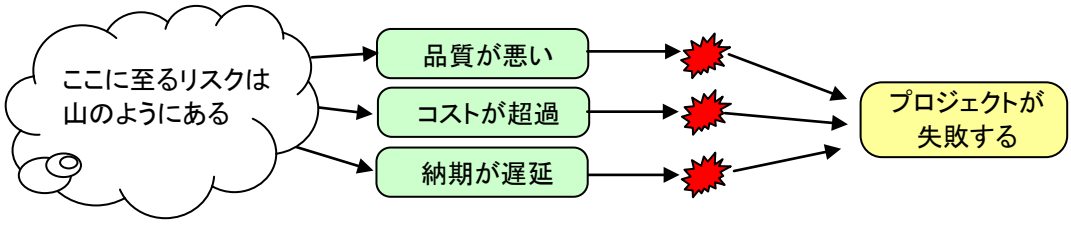


図 C-5 システム開発におけるリスク事象

そもそも、この段階になってリスクの予防策を考えても全く手遅れであり、むしろ考えるべきはリカバリープランということになってしまう。

即ち、「プロジェクトが失敗する」というリスク事象からどこまで遡り、どの段階のリスク事象ドライバーに対して予防策を講じるのが最も有効で手が打ちやすいのかを考えることが重要となる。

(5)リスクの連鎖における予防ポイントの決め方

前節ではシステム開発におけるリスク連鎖について述べたが、実際には「プロジェクトが失敗する」、あるいは「QCD のトラブルに陥る」というレベルから遡っていくことにはいささか無理がある。

考え方としては、「仕様変更が多発して手戻りになる」といった、よくありがちなトラブルに焦点を合わせ、そのリスク要因から遡るのが現実的であろう。

プロジェクトの性質や環境にもよるが、例えば仕様変更が多発する要因となるリスク事象ドライバーを次のように列挙することができる。

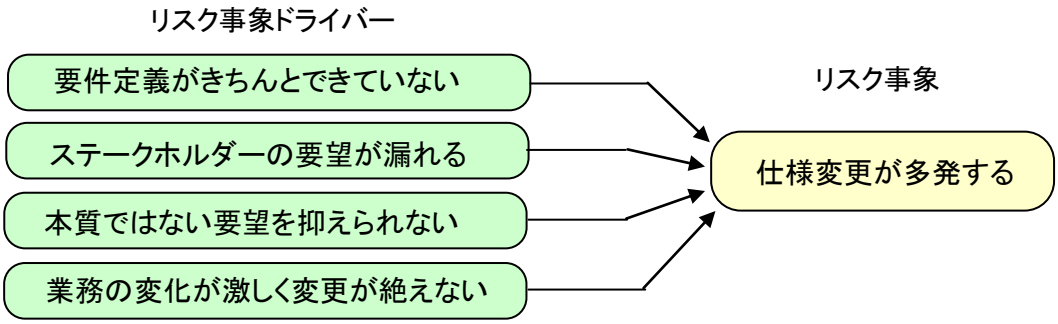


図 C-6 リスク事象(仕様変更が多発する)のリスク事象ドライバー

この中で、「要件定義がきちんとできていない」というリスク事象ドライバーに着目すると、予防策を考えるにはまだ荒く、もう少し遡って考える必要があるように思える。

一方で、「ステークホルダーの要望が漏れる」というリスク事象ドライバーでは、これをリスク事象としたリスク事象ドライバーを洗い出せば、その段階で有効な予防策が打てそうに思える。

例えば、ステークホルダーの洗い出しが不十分なのであれば、その洗い出しを徹底することが予防策になるであろうし、ステークホルダーのレビュー参加率が低いのであれば、如何にモチベーションを上げて参画させるかを考えることが予防策となる。

即ち、これから実施するプロジェクトに「要件のヒアリング対象となるステークホルダーの決め方が曖昧」、あるいは「ステークホルダーのプロジェクト参画意識が低い」といったリスク事象ドライバーが存在すると思われるなら、「仕様変更が多発する」というリスク事象を避けるために、リスク事象ドライバーを回避、あるいは軽減するための予防策を打っておく。

また、「本質ではない要望を抑えられない」というリスク事象ドライバーでは、「システム化する目的が不明確」、「要望を受け入れるルールの不備」、あるいは「要件コントロールするリーダの力量問題」など、幅広いリスク事象ドライバーに遡るため、優先度の高いものからもう一段階程度遡るのが適当だと思われる。

最後の「業務の変化が激しく変更が絶えない」というリスク事象ドライバーの場合は、仕様変更の発生を抑えることが難しい状況だと思われる。このため、仕様変更を起こさないための予防策というより、如何に仕様変更に対応するかという予防策の方が有効だと思われる。

⑹メトリクスの設定

前節で説明したように、リスク事象を遡ってゆくと、そのリスク事象ドライバーがプロジェクトに存在するかどうか判断しやすい粒度に到達する。ただ、それはリスク事象やプロジェクト環境によって異なる。

リスク事象ドライバーによっては、その存在が分かりにくいものもあるので、何らかのメトリクスがあった方がプロジェクト関係者間で合意しやすい。

メトリクスといっても、多くはチェックリストの項目のようなもので、前節の例で言えば「ステークホルダーの一覧が作られているか」、あるいは「ステークホルダー分析を行っているか」といったもので十分である。

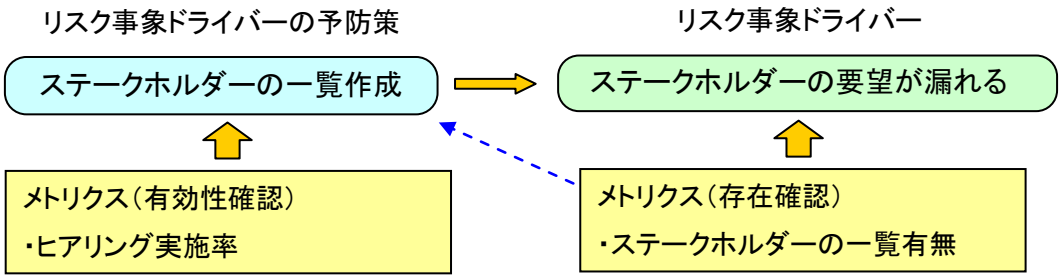


図 C-7　メトリクスの例

リスク事象ドライバーの存在を確認し、その予防策を打ったとして、次に必要なのは、その予防策が有効に機能しているかということだ。このモデルでは、そのためのメトリクスの設定も提案している。

先の例では、ステークホルダーの一覧を作成しても、それが活用されなければ意味がないため、例えば一覧にあるステークホルダーのうち何人にヒアリングを実施したかという実施率がメトリクスになりうる。

(7)まとめ

このように、リスク事象が連鎖する中で、自分達のプロジェクトにそのリスク事象ドライバーが存在するかどうかを適切な粒度で見極めること、その存在の確認はメトリクスを使って関係者と合意すること、そしてその予防策を立案・実施し、できればその有効性をメトリクスで確認することが、このモデルの骨子である。

参考文献:

プレストン・G・スミス、ガイ・M・メリット（澤田美樹子訳）『実践・リスクマネジメント』生産性出版（2003）
『ISO/IEC 12207：2008　ソフトウェアライフサイクルプロセス（JISX0160:2012）』
独立行政法人情報処理推進機構ソフトウェア・エンジニアリング・センター『共通フレーム2013』
独立行政法人情報処理推進機構ソフトウェア・エンジニアリング・センター『非機能要求グレード』
独立行政法人情報処理推進機構ソフトウェア・エンジニアリング・センター『IT プロジェクトの「見える化」』
International Institute of Business Analysis（IIBA）『BABOK　A Guide to the Business Analysis Body of Knowledge：ビジネスアナリシス知識体系ガイド）』(2009)
IBM（富永章・栗江哲誠　他）『PROVISION Winter 2005 No.44』(2005)
PMI『プロジェクトマネジメント知識体系ガイド（PMBOK®ガイド）』第4版、PMI（2009）

- **執筆者**

  定量的管理基盤 WG

野中　誠	学校法人東洋大学
沖汐　大志	日本ユニシス株式会社
小倉　理礼	ソニー生命保険株式会社
小田　雅一	IT ホールディングス株式会社
小浜　耕己	スミセイ情報システム株式会社
菅野　和浩	株式会社日本経営データ・センター
早乙女　真	株式会社 NTT データ経営研究所
服部　克己	日本ユニシス株式会社
藤原　良一	三菱電機インフォメーションシステムズ株式会社
古川　正伸	株式会社東京証券取引所
山口　一郎	東京ガス株式会社（株式会社ティージー情報ネットワーク）
三毛　功子	独立行政法人情報処理推進機構

- **運営支援**

独立行政法人情報処理推進機構	志水　裕香

- **オブザーバ**

独立行政法人情報処理推進機構	山下　博之
独立行政法人情報処理推進機構	大和田　裕
独立行政法人情報処理推進機構	鈴木　三紀夫