資料1-2

情報処理システム

高信頼化 教訓集

（ITサービス編）

2017年度版

（2017年度新着教訓追加版）

ガバナンス／マネジメント教訓　19件
技術領域教訓　29件

ガバナンス／マネジメント領域の教訓(1)

教訓 ID	教訓概要
G1	システム開発を情シス部門だけの仕事にせず、各事業部門が自分のこととして捉える「態勢」をつくることが大切
G2	発注者は要件定義に責任を持ってシステム構築にかかわるべし
G3	運用部門は上流工程（企画・要件定義）から開発部門と連携して進めるべし
G4	少しでも気になった事象は放置せず発信し、とことん追求すべし
G5	サービスの拡大期には業務の処理量について特に入念な予測を実施すべし
G6	作業ミスとルール逸脱は、個人の問題でなく、組織の問題！
G7	クラウド事業者と利用者が連携した統制がとれたトラブル対応体制を整備すべし
G8	共同利用システムでは、非常時対応を含めて利用者間の情報共有を図ること
G9	システム利用不可時の手作業による代替業務マニュアルを作成し定期的な訓練を行うべし
G10	関係者からの疑義問合せは自社システムに問題が発生していることを前提に対処すべし！

ガバナンス／マネジメント領域の教訓(2)

教訓 ID	教訓概要
G11	システムの重要度に応じて運用・保守の体制・作業に濃淡をつけるべし
G12	キャパシティ管理は、業務部門とIT部門のパートナーシップを強化するとともに、管理項目と閾値を設定してPDCAサイクルをまわすべし！
G13	キャパシティ管理は関連システムとの整合性の確保が大切
G14	設計時に定めたキャパシティ管理項目は、環境の変化にあわせて見直すべし
G15	保守作業は「予期せぬ事態の発生」を想定し、 サービス継続を最優先として保守作業前への戻しを常に考慮すること
G16	本番環境へのリリースは、保守担当が無断でできないような仕組みを作るべし！
G17	サービスの重要度を識別し、それに応じた連絡体制や障害検知の仕組みを作れ
G18	障害対策とは許容時間内の回復や停止中の業務継続まで具体化すること
G19	みんなで唱和！障害減らす教訓共有
G20	「システム運用環境変更時の品質向上」は正攻法の成功事例に学べ!
G21	サーバ証明書等の有効期限の確認方法を工夫せよ

技術領域の教訓(1)

教訓 ID	教訓概要
T1	サービスの継続を優先するシステムにおいては、疑わしき構成要素を積極的にシステムから切り離せ（“フェールソフト”の考え方）
T2	蟻の目だけでなく、システム全体を俯瞰する鳥の目で総合的な対策を行うべし
T3	現場をよく知り、現場の知識を集約し、現場の動きをシミュレートできるようにすべし
T4	システム全体に影響する変化点を明確にし、その管理ルールを策定せよ
T5	サービスの視点で、「変更管理」の仕組み作りと「品質管理責任」の明確化を！
T6	テスト環境と本番環境の差異を体系的に整理し、障害のリスク対策を練る
T7	バックアップ切替えが失敗する場合を考慮すべし
T8	仮想サーバになってもリソース管理、性能監視は運用の要である
T9	検証は万全？それでもシステム障害は起こる。回避策を準備しておくこと
T10	メッシュ構成の範囲は、可用性の確保と、障害の波及リスクのバランスを勘案して決定する

技術領域の教訓(2)

教訓 ID	教訓概要
T11	サイレント障害を検知するには、適切なサービス監視が重要
T12	新製品は、旧製品と同一仕様と言われても、必ず差異を確認！
T13	利用者の観点に立った、業務シナリオに即したレビュー、テストが重要
T14	Webページ更新時には、応答速度の変化等、性能面のチェックも忘れずに
T15	緊急時こそ、データの一貫性を確保するよう注意すべし
T16	システム構成機器の修正パッチ情報の収集は頻繁に行い、緊急性に応じて計画的に対応すべし
T17	長時間連続運転による不安定動作発生の回避には定期的な再起動も有効！
T18	新たなサブシステムと老朽化した既存システムとを連携する場合は両者の仕様整合性を十分確認すべし
T19	リレーショナルデータベース（ＲＤＢＭＳ）のクエリ自動最適化機能の適用は慎重に！
T20	パッケージ製品の機能カスタマイズはリスクを認識し 特に必要十分なチェック体制やチェック手順を整備して進めること

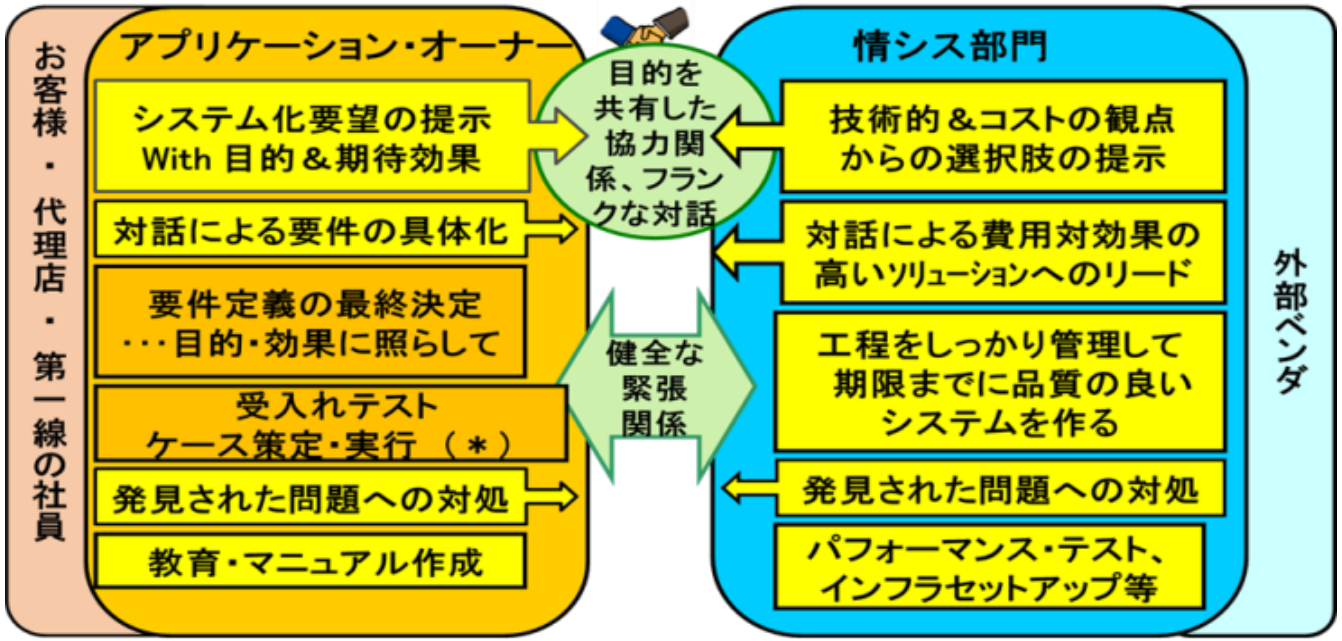
技術領域の教訓(3)

教訓ID	教訓概要
T21	作業ミスを減らすためには、作業指示者と作業者の連携で漏れのない対策を！
T22	隠れたバッファの存在を把握し、目的別の閾値設定と超過アラート監視でオーバフローを未然に防止すること
T23	障害監視は、複数の観点から実装し、障害の見逃しを防げ！
T24	サービス縮退時の対策を考慮せよ
T25	障害原因が不明でも再発予防と発生時対策はできる
T26	既存システムの流用開発はその前提条件を十分把握し、そのまま利用可能な部分と変更する部分を調査して実施する
T27	パッケージはサポートを買え
T28	パッケージを更新する時は、変更内容の詳細確認と回帰テストで二重に安全を確保せよ
T29	単位などの定義が異なる制限値、連携するシステム間で使っていませんか？

G1：システム開発を情シス部門だけの仕事にせず、各事業部門が自分のこととして捉える「態勢」をつくることが大切

システムトラブルの8割は、上流の要件定義局面でのコミュニケーション・ギャップから問題が生じていることが判明
→システム開発におけるビジネスサイドの役割と責任を明確化し、コミュニケーションの質を高める態勢「アプリケーション・オーナー制度」

- システム開発は、情シス部門に任せきりにすべき仕事ではなく、自分の考えた商品や施策を具体化するために行う自分自身の仕事であるという「オーナーシップ」の考え方を持たせる。
- 事業部門に、要件の詳細が固まるまで、情シス部門と対話を繰り返す責任を持たせ、要件定義の最終責任を負わせる。
- 事業部門に、要件定義どおりにシステムが出来たかどうか受入れテストを実施する責任を負わせる。



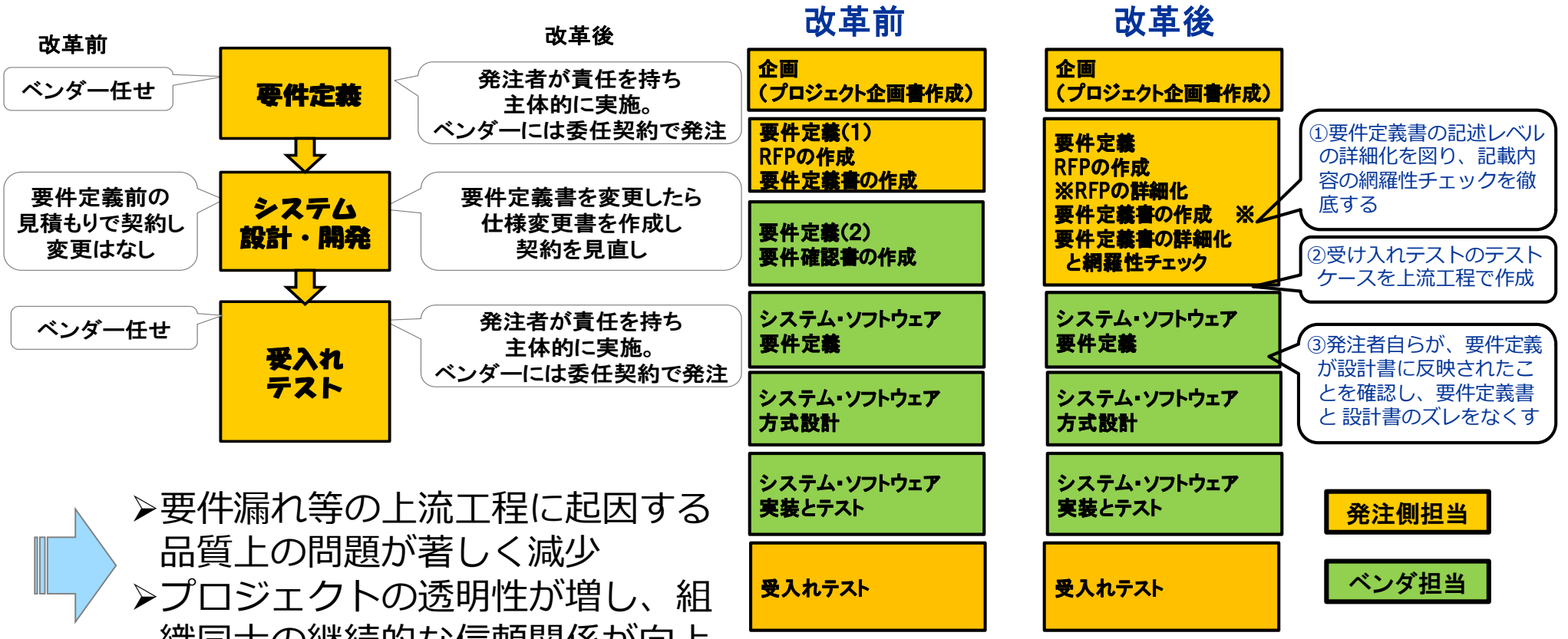
＊　要件定義に責任を持つ以上、要件通りできたかの受入れテストも実施することが重要。このように手を動かす責任にしない限り、表面的なものになる。

アプリケーション・オーナー制度：責任と役割分担（東京海上日動火災株式会社の例）

G2：発注者は要件定義に責任を持ってシステム構築にかかわるべし

システム開発・運用をITベンダに外部委託するケースで、開発案件の増加に伴い任せる業務が徐々に拡大し、要件定義や受入れテスト等、発注者としての役割を果たし切れていない

1) 要件定義書の中身と受入れテストについての責任は発注者とする
2) 開発プロセス標準を見直し、上流の要件定義を押さえる（上流工程完璧主義）



- 要件漏れ等の上流工程に起因する品質上の問題が著しく減少
- プロジェクトの透明性が増し、組織同士の継続的な信頼関係が向上

G3：運用部門は上流工程（企画・要件定義）から開発部門と連携して進めるべし

【問題】システム運用の現場で、オペレータの操作ミスや入力データミスの多発。

【原因】運用要件の検討モレあるいは軽視。→運用者が要件定義作業へ参加していないことや、参加していても運用要件が十分に取り込まれなかったことに起因。

【対策】企画・要件定義作業において、運用者の視点からシステム要件を確認（下表）
これを参考にして、対象となるシステムに合わせて役割分担表などに表現。
プロジェクト開始前に関係者でレビューして合意し、プロジェクトに適用。

参考：「運用者が企画・要件定義工程で確認する項目」の例

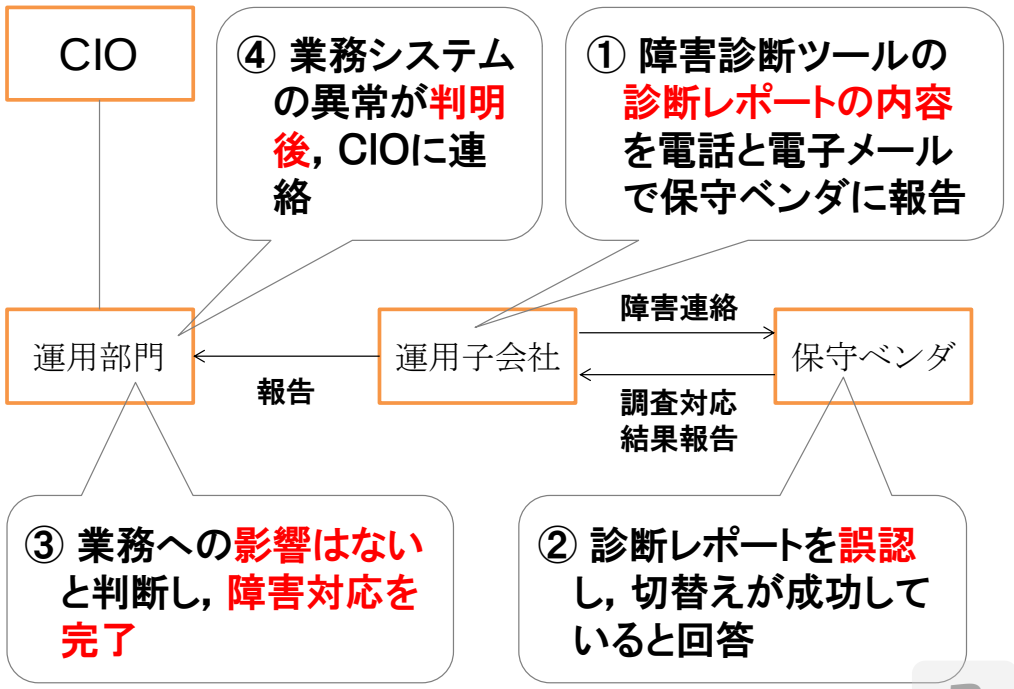
No	工程	分類	項目	全体で確認する項目	運用者が確認する項目
1	企画	起案	■経営戦略を見据えたシステム構築及びシステム構築の目的・目標の明確化	①経営戦略の具現化 ②情報（システム）戦略の具現化 ③システム化の目的・方針 ④納期（スケジュール） ⑤システム利用期間、ライフサイクル概要計画	①要件の把握 ②納期（スケジュール）の確認 ③システムのライフサイクル（更新間隔）確認 ④経営戦略の理解 ⑤情報（システム）戦略の理解
2		現状分析	■システム（業務）の現状分析、問題点・課題の抽出と分析	①運用状況 ②問題点・課題 ③最新のシステム動向、技術動向の分析	①構築ノウハウの提供 ②現状課題の提供 ③最新技術動向・トレンド等の情報分析提供 ④運用状況の報告
2-2		企画立案	■新システムの企画立案	①全社目標の体系化と施策の定義 ②情報システム要件のまとめ ③システム化の企画立案	①運用改善から見た新システム要件の提案
3		投資対効果	■システム構築における投資対効果の明確化	①投資対効果 ②コスト計画の立案	①運用要件者側に必要な費用の見積作成
4		承認	■システム構築の承認	①システム構築の承認を得る	

G4：運用者は、少しでも気になった事象は放置せず共有し、とことん追求すべし

【問題】運用者及び保守者が夜間作業中の異常を誤認してオンラインサービスの開始が遅れ、数時間停止

【原因】運用担当者が察知した異常を、保守担当者が異常なしと誤認．運用部門責任者も十分に確認せず、報告そのままに問題なしと判断し、ＣＩＯへの報告を怠る．
運用担当者は確かな異常状況に気づいていたが、運用マニュアル通りに作業し、気づきを運用部門の責任者等に伝えなかった．

【対策】①「運用担当者が現場での異常を察知したときには、状況判断できる運用部門の社員にその情報を連絡して協議する態勢を作る」ことを運用マニュアルに明記．
②障害対応体制面での改善・強化：
・事象として障害と断定できない場合でも障害の可能性がある場合は早期に上位役職者へ報告するルールの追加作成
・状況判断できる運用部門社員の24時間常駐
③確認手順及び項目の明確な定義
④教育・訓練

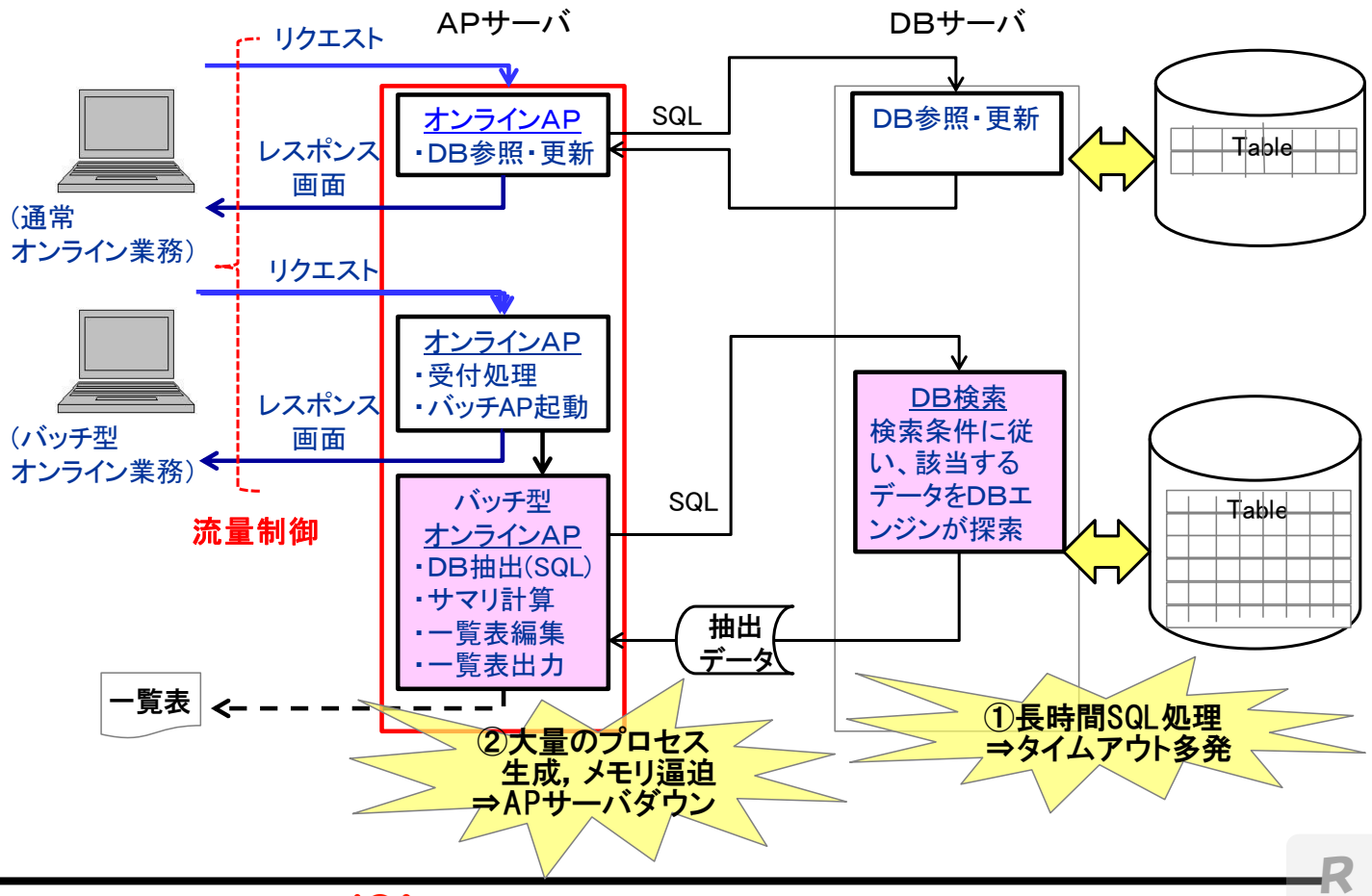


G5：サービスの拡大期には業務の処理量について特に入念な予測を実施すべし

【問題】共同利用システムが、稼働開始１年後の利用増大傾向の中、負荷集中によりダウン

【原因】共同利用各社の処理件数予測が不十分であった（ベンダ任せ）ため、DBサーバ内ソフトのメモリリーク・バグが発現

【対策】
- 利用各社による運営協議会の設置
- キャパシティプラニングを含めた共同利用各社の責任の明確化
- ベンダとの契約に運営協議会で決めるべき項目を明記



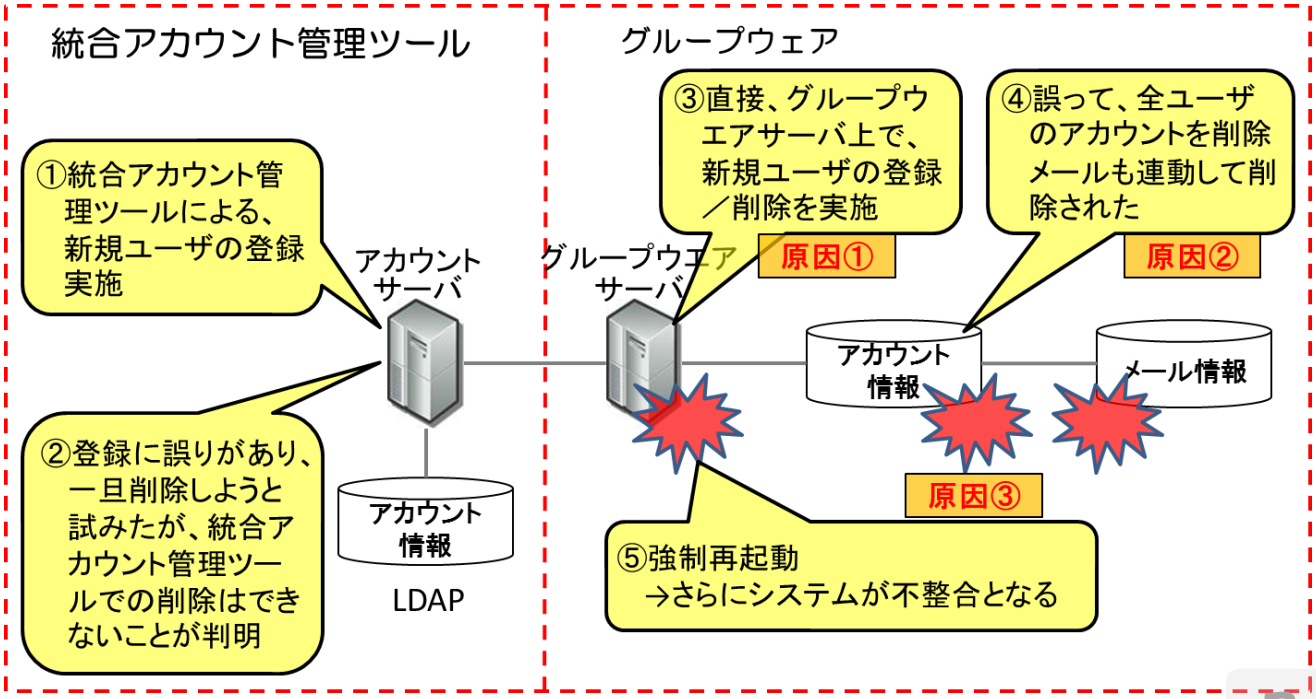
G6：作業ミスとルール逸脱は、個人の問題でなく、組織の問題！

【問題】運用作業者がグループウェアの全ユーザデータを削除

【原因】不慣れな運用作業者（新人）が、独断で、運用規定外の手段（管理ツールを介さないサーバへの直接アクセス）により、誤操作（ルール逸脱）
繁忙な環境下、迅速な処理が求められる状況で、各メンバがお互いの作業に追われて連携できず，不慣れな作業者は、多忙な熟練者にも聞くことができず，自分が業務を遅らせる原因になってはいけないというプレッシャーから，ルール逸脱

運用チーム内のスキルの共有も不十分

【対策】組織的な総合対策：
- 作業を受ける場合のリスクを考慮した受諾の判断基準作成
- 複数名体制での作業実施等、ルールを逸脱しない作業規定の作成
- 普段のチーム内のコミュニケーション

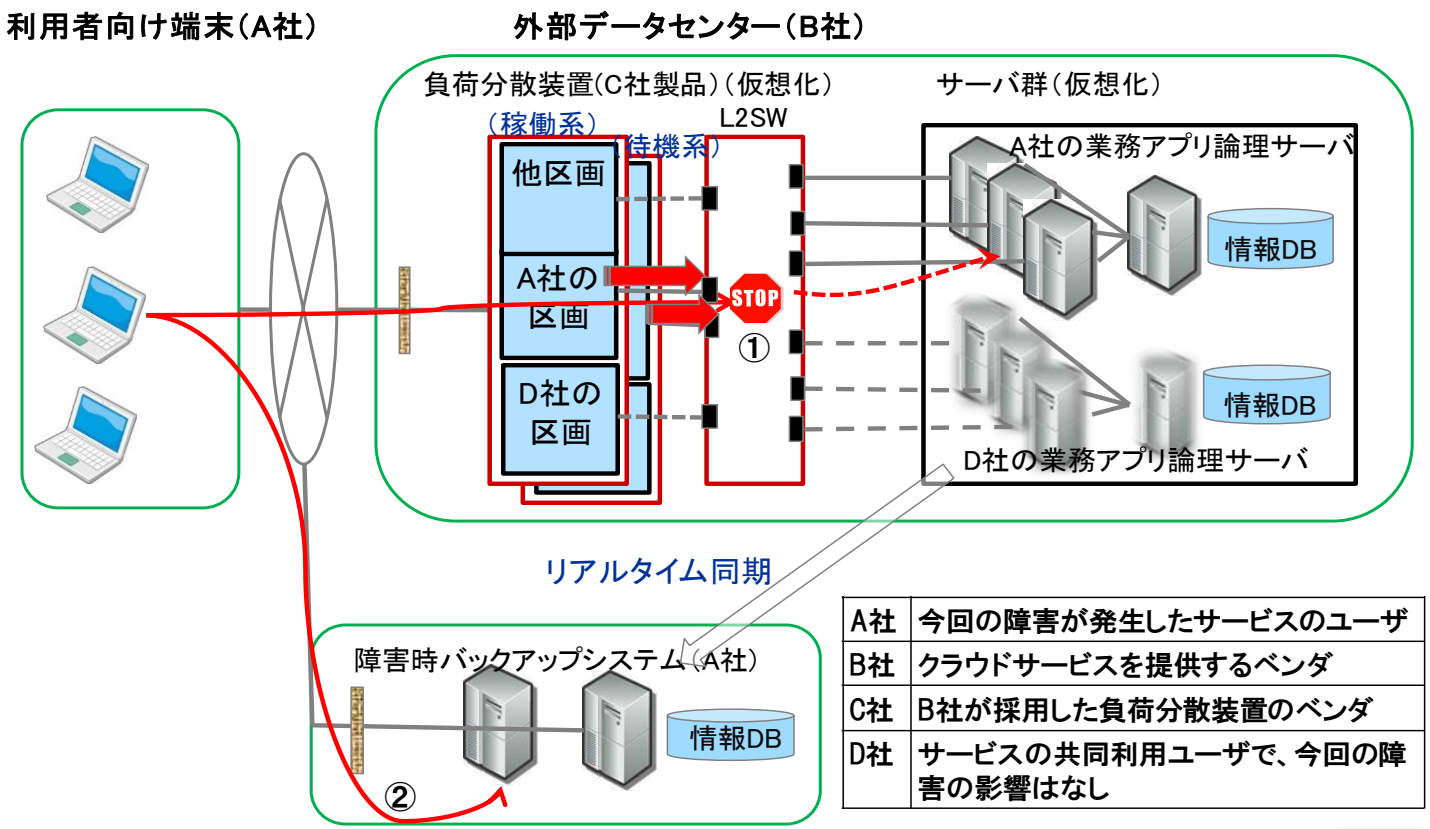


G7：クラウド事業者と利用者が連携した統制がとれたトラブル対応体制を整備すべし

【問題】クラウドサービスに移行したシステムの通信機器に障害が発生し、（単なる負荷分散装置の障害にも関わらず）丸１日間業務が停止

【原因】運用時のトラブル管理体制が決まっていなかった

【対策】体制の整備：
- 障害対応体制（報告，連絡，相談）の明確化
- 契約におけるサービスレベル定義（責任分界点の明確化）
- クラウド事業者と関連サードパーティ業者間の障害対応体制の確立（適確な連携体制）



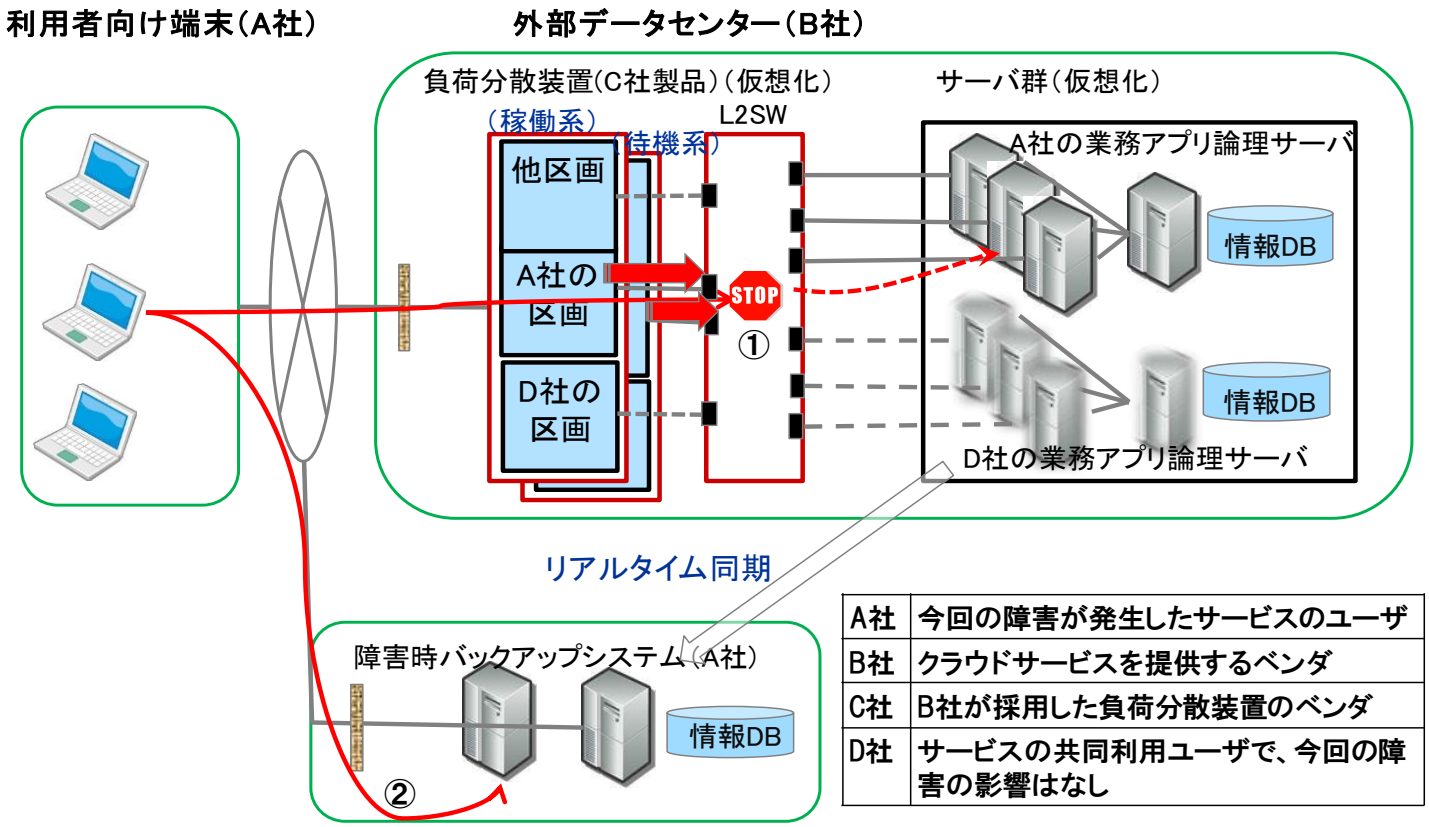
A社	今回の障害が発生したサービスのユーザ
B社	クラウドサービスを提供するベンダ
C社	B社が採用した負荷分散装置のベンダ
D社	サービスの共同利用ユーザで、今回の障害の影響はなし

G8：共同利用システムでは、非常時対応を含めて利用者間の情報共有を図ること

【問題】クラウドサービスに移行したシステムの通信機器に障害が発生し、（単なる負荷分散装置の障害にも関わらず）丸１日間業務が停止

【原因】ベンダーおよび共同利用者間の障害発生時の連絡体制が決まっていなかった。

【対策】
・障害復旧時の停止/再起動単位と利用者影響の明確化
・システム停止/再起動の条件や責任についてのＳＬＡ合意
・利用者間の非常時緊急連絡体制の確立



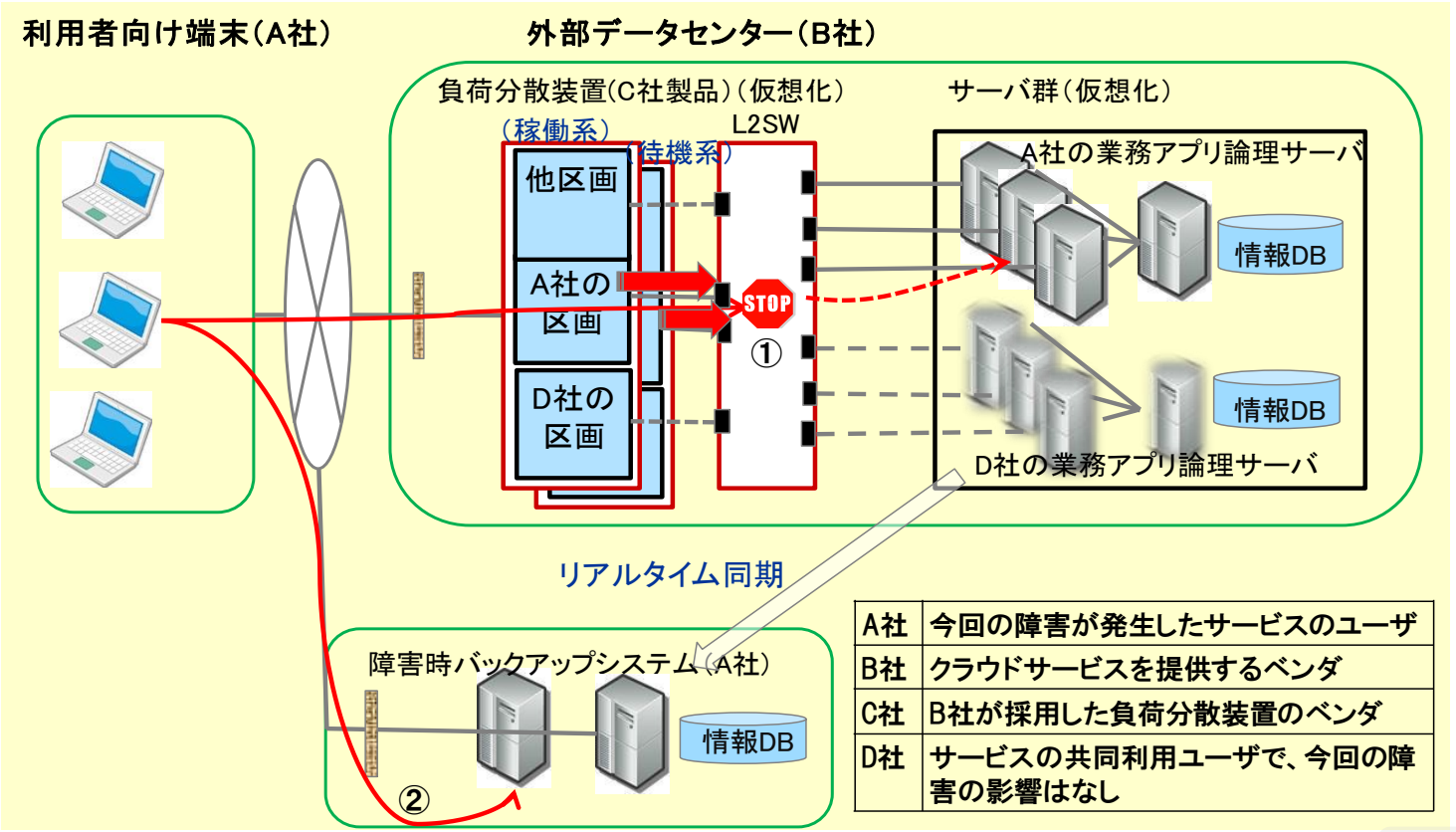
A社	今回の障害が発生したサービスのユーザ
B社	クラウドサービスを提供するベンダ
C社	B社が採用した負荷分散装置のベンダ
D社	サービスの共同利用ユーザで、今回の障害の影響はなし

G9：システム利用不可時の手作業による代替業務マニュアルを作成し定期的な訓練を行うべし

【問題】 システム停止時の顧客対応が十分にできなかった

【原因】過去にシステム障害が発生したことがなく、システムが利用できない前提での業務マニュアルがなかった

【対策】システム利用不可時の代替業務マニュアルの作成



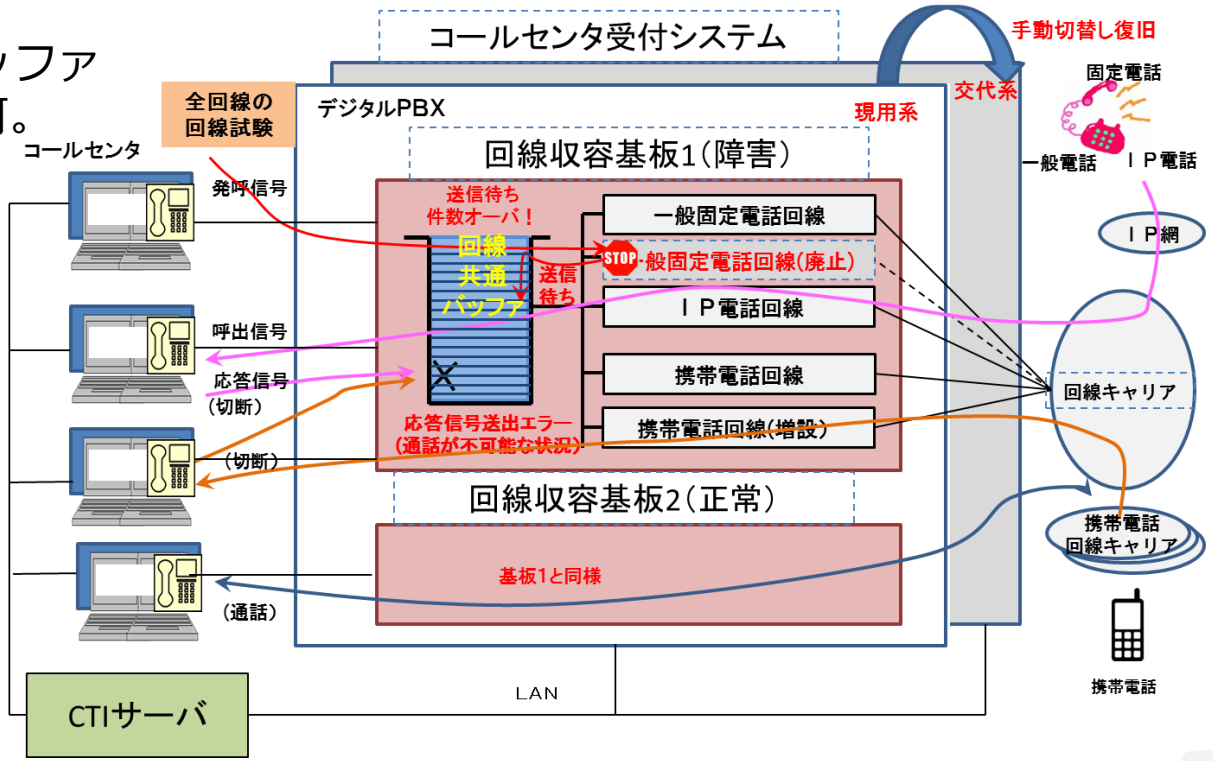
A社	今回の障害が発生したサービスのユーザ
B社	クラウドサービスを提供するベンダ
C社	B社が採用した負荷分散装置のベンダ
D社	サービスの共同利用ユーザで、今回の障害の影響はなし

G10: 関係者からの疑義問合せは自社システムに問題が発生していることを前提に対処すべし！

【問題】コールセンタにおいて電話コールの一部が着信後に即切断されてしまう事象が発生していたがイタズラ電話との認識、また通信回線事業者からコールの接続異常が時々発生しているが問題はないかと問合せはあったが他の事業者は正常との回答であったため問題視しなかった。システム障害と気付くまでに4時間経過していた。

【原因】設定変更のミス。回線収容基板の一部廃止に伴い回線試験用の設定も削除するべきだったが漏れた。
エラー電文の蓄積が共通バッファオーバフローを招き通話不可。

【対策】(復旧措置)
交代系への切替えで復旧
(作業漏れへの対策)
保守運用マニュアルの改善
(障害状態検知への対策)
バッファ監視機能追加
(異常申告への対策)
回線事業者連絡会の設置
(交代系への切替え判断)
原因調査より復旧作業優先



G10: 関係者からの疑義問合せは自社システムに問題が発生していることを前提に対処すべし！

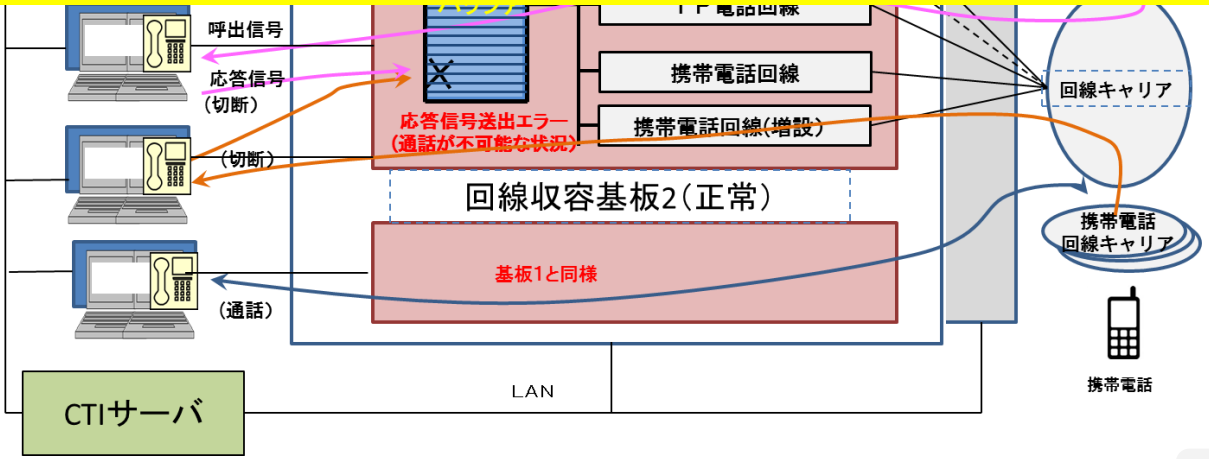
【問題】コールセンタにおいて電話コールの一部が着信後に即切断されてしまう事象が発生していたがイタズラ電話との認識、また通信回線事業者からコールの接続異常が時々発生しているが問題はないかと問合せはあったが他の事業者は正常との回答であったため問題視しなかった。システム障害と気付くまでに4時間経過していた。

【原因】設定変更のミス。回線収容基板の一部廃止に伴い回線試験用の設定も削除するべき

断続的に発生するオンラインシステムのサイレント障害はセンタ側では気付かないことが多い。利用者の声は貴重です。

【対策

(作業漏れへの対策)
保守運用マニュアルの改善
(障害状態検知への対策)
バッファ監視機能追加
(異常申告への対策)
回線事業者連絡会の設置
(交代系への切替え判断)
原因調査より復旧作業優先



G11: システムの重要度に応じて運用・保守の体制・作業に濃淡をつけるべし

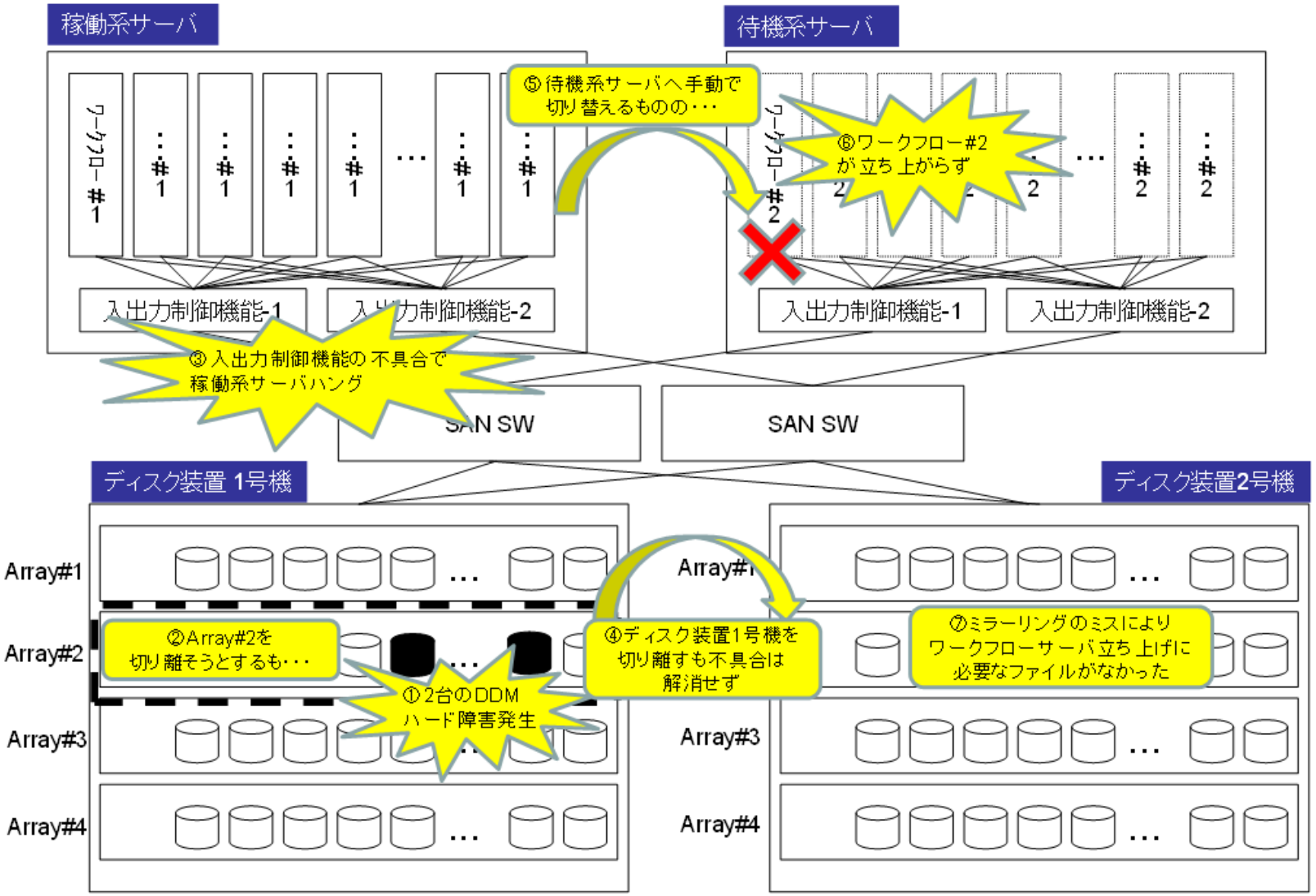
【問題】A社の社内ワークフローシステムに障害が発生し、連携している顧客向けサービスが終日全面停止した。システム関係者が集まったものの、状況を把握するのに時間が掛かり、システム停止時間は長時間に及んだ。

【原因】RAID5構成のストレージにおいて、ディスクの同時故障が発生。当該ARRAYを切り離す仕様となっていたが、製品不具合によりサーバがハングアップした。サーバを待機系に切り替えようとするも、ミラーリングの誤設定により必要なファイルが待機系になく、成功しなかった。

【対策】製品不具合については、ベンダから修正パッチが提供されていたにもかかわらず、他社で大きな影響が出ていなかったため、A社には知らされていなかった。ミラーリングの誤設定については、保守作業におけるチェック体制の見直しを行った。また、復旧に多くの時間が掛かり、顧客への影響が大きくなってしまったことに注目し、以下の対応を行った。

- **障害対応メンバ間の意思疎通がしやすいよう、システム共通の資料を準備**
- **システム障害発生時に、組織内で適切に情報の共有を行うため、システム障害の対策本部を新規に立ち上げた。**
- **基幹システムを中心に、顧客への影響の有無、推定される損害額など、システムの重要度に応じてランク付けを行い、そのランクに応じてシステム保守対応を行うことにした。**

G11：つづき



G11：つづき

- システム障害対策本部の**設置を運用規定化**
- 全員が全体俯瞰図、システム構成図等**同じ資料を持って議論**
- 顧客への影響の有無、推定される損害額など、**システムの重要度に応じたランク付けとシステム保守対応**

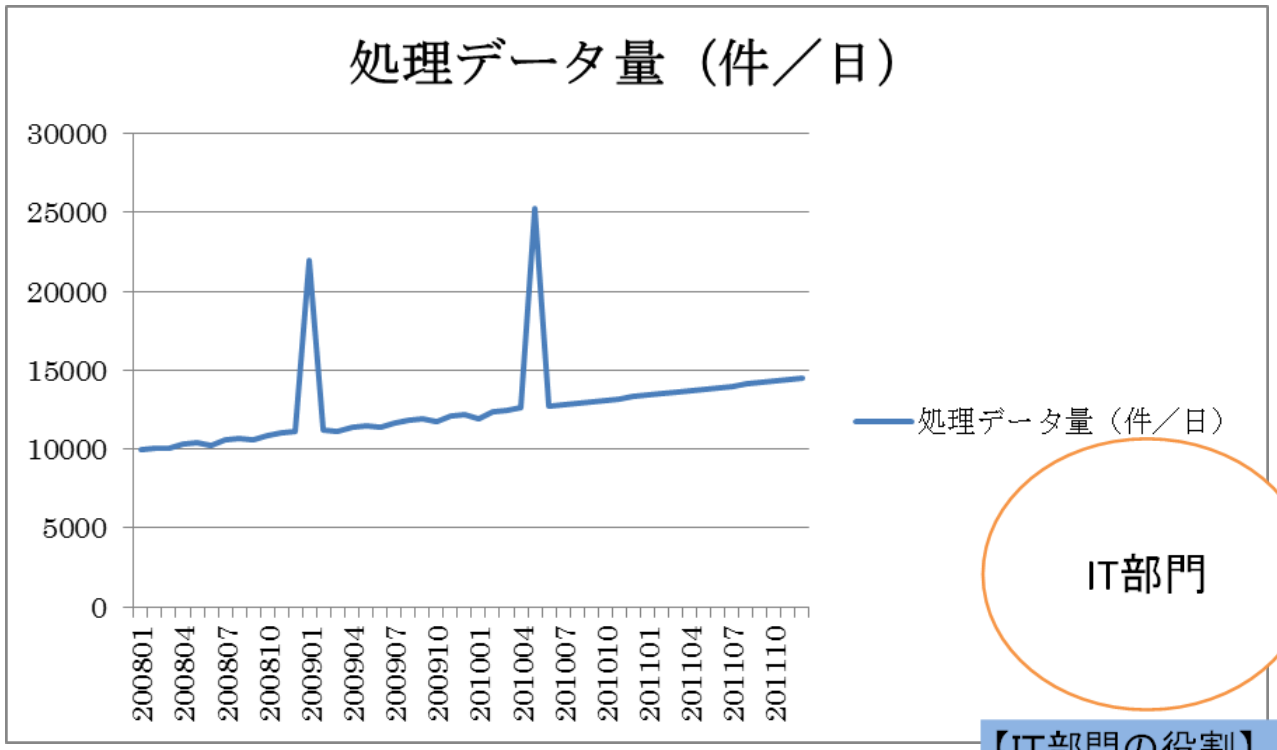
G12：キャパシティ管理は、業務部門とIT部門のパートナーシップを強化するとともに、管理項目と閾値を設定してPDCAサイクルをまわすべし！

【問題】A社のシステムはサービスの継続を優先するデータの非同期送受信（メッセージ交換型）のオンラインシステムである。このシステムの処理件数は、以前から一般的な増加とともに、突発的な事象によるデータ量の急増がみられた。このA社のシステムにある日、処理能力を越えた注文が殺到し、サービスの時間が短縮となった。

【原因】近年はコンピュータでの取引が進み、処理件数の変化が激しい。一般的なデータ量の増加に加えて、想定を超える突発的な事象によるデータ量の急増が起こる。主な原因としては、業務部門とIT部門とのキャパシティの合意形成やキャパシティの管理方法が明確でないことが多い。

【対策】**①キャパシティ管理については、システムごとに責任を持つ業務部門を決め適材適所で役割分担し、コミュニケーションをとる協力体制を作る。**
②過去の実績をもとに算出したルール（例えば、「突発的な増加に対応可能な」過去最高実績の２倍のキャパシティを確保する）に基づいて性能を拡張する。
③システムごとに管理項目と閾値を設定し、キャパシティの拡張方法や拡張限界等を明確化する。
④業務部門が日々の業務量やビジネス環境などから将来予測を行い、IT部門がシステムの拡張を検討する。

G12：つづき



IT部門

連携

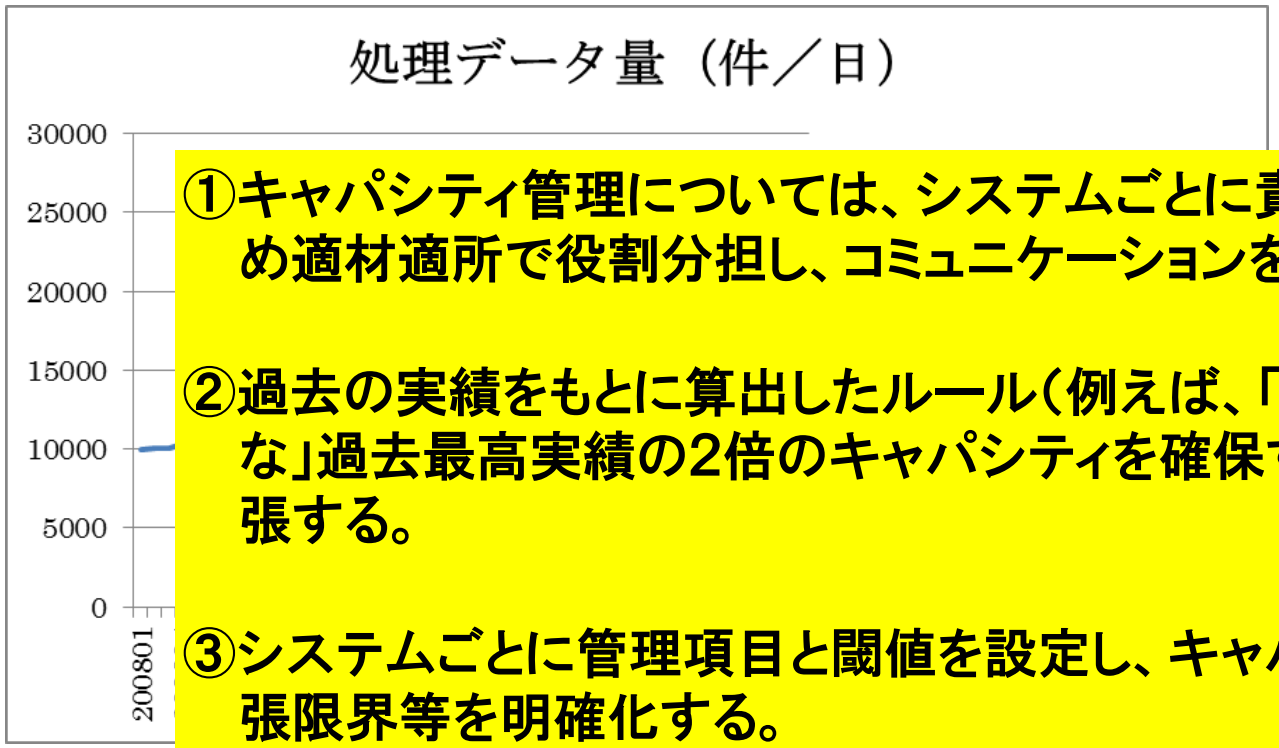
業務部門
（オーナ）

【IT部門の役割】
・キャパシティ計画作成
・システムの拡張検討

【連携】
・キャパシティ計画レビュー
・回帰分析で予測値を研究するなど

【業務部門の役割】
・日々の業務量把握
・将来の業務量予測

G12：つづき



①キャパシティ管理については、システムごとに責任を持つ業務部門を決め適材適所で役割分担し、コミュニケーションをとる協力体制を作る。

②過去の実績をもとに算出したルール（例えば、「突発的な増加に対応可能な」過去最高実績の2倍のキャパシティを確保する）に基づいて性能を拡張する。

③システムごとに管理項目と閾値を設定し、キャパシティの拡張方法や拡張限界等を明確化する。

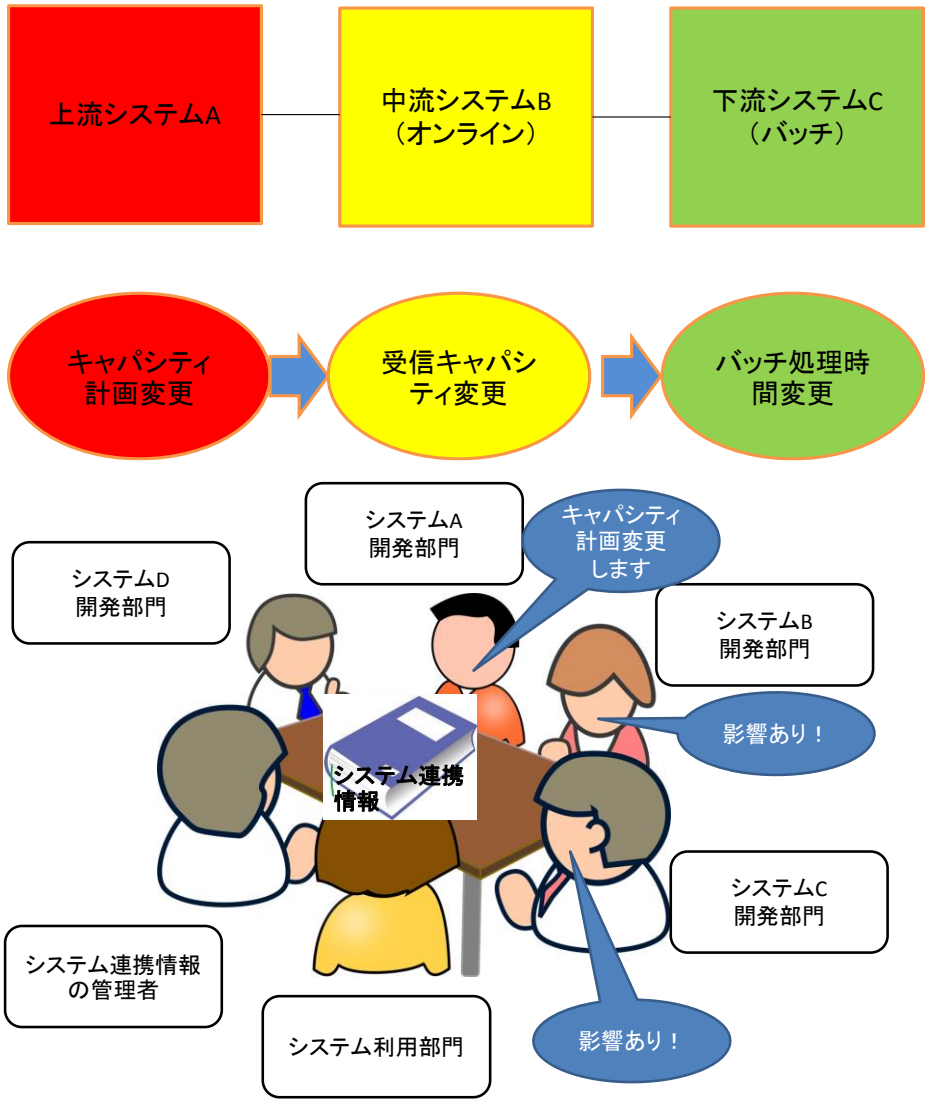
④業務部門が日々の業務量やビジネス環境などから将来予測を行い、IT部門がシステムの拡張を検討する。

G13: キャパシティ管理は関連システムとの整合性の確保が大切！

【問題】A社のシステムは、以前から一般的な増加とともに、突発的な事象によるデータ量の急増がみられた。このA社のシステムにある日、処理能力を越えた注文が殺到し、サービスの時間が短縮となった。

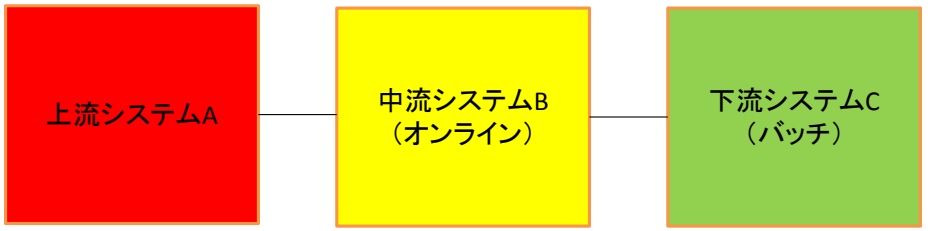
【原因】近年はコンピュータでの取引が進み、処理件数の変化が激しい。一般的なデータ量の増加に加えて、想定を超える突発的な事象によるデータ量の急増が起こる。主な原因としては、関連システムとの整合性の確保ができていないことが多い

【対策】キャパシティ管理に関する課題を解決し、全社的なキャパシティ管理業務を担う会議体を作り、システム連携情報を管理する。



G13: キャパシティ管理は関連システムとの整合性の確保が大切！

【問題】A社のシステムは、以前から一般的な増加とともに、突発的な事象によるデータ量の急増がみられた。このA社のシステムにある日、処理能力を越えた注文が殺到し、サービスの

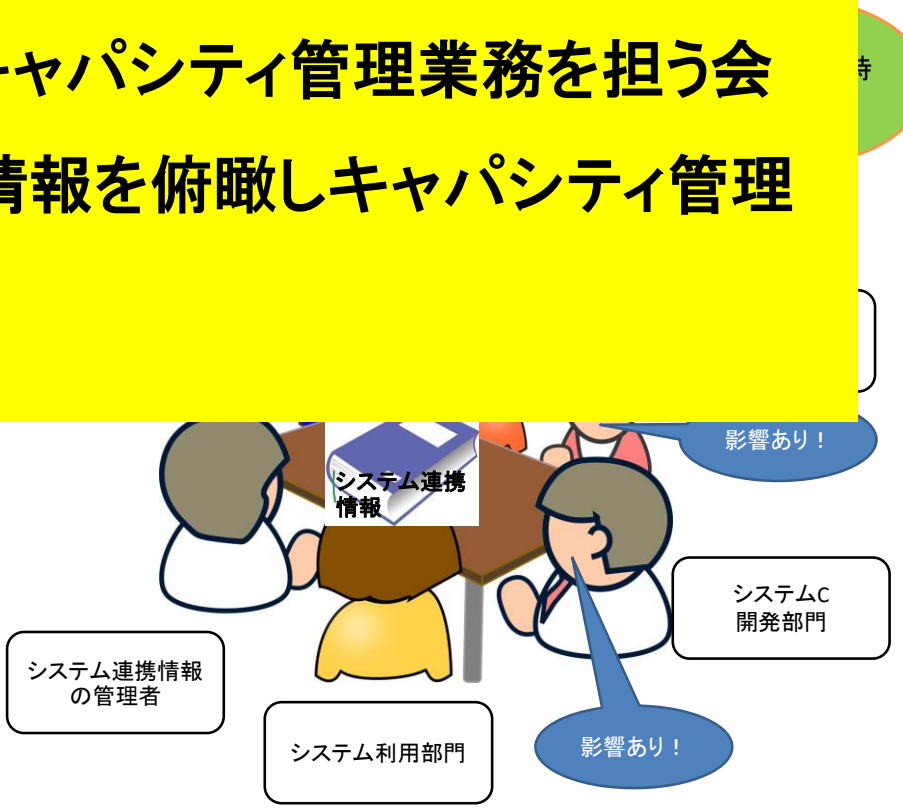


【原因】

各担当が参加する全社的なキャパシティ管理業務を担う会議体を作り、システム間連携情報を俯瞰しキャパシティ管理に関する課題を解決する

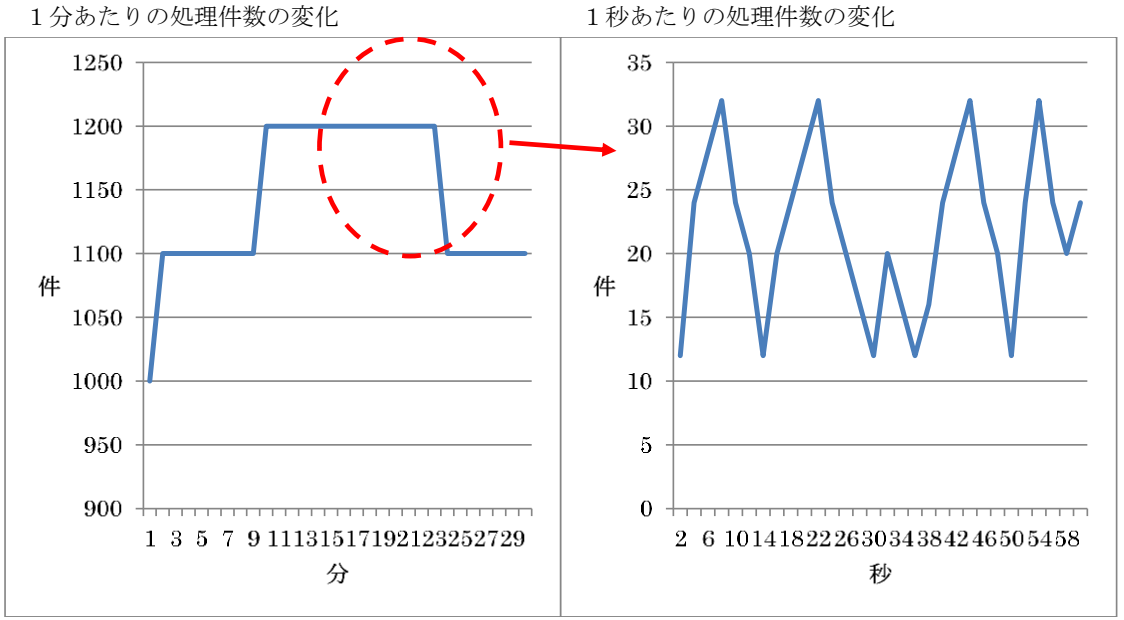
合性の確保ができていないことが多い

【対策】キャパシティ管理に関する課題を解決し、全社的なキャパシティ管理業務を担う会議体を作り、システム連携情報を管理する。



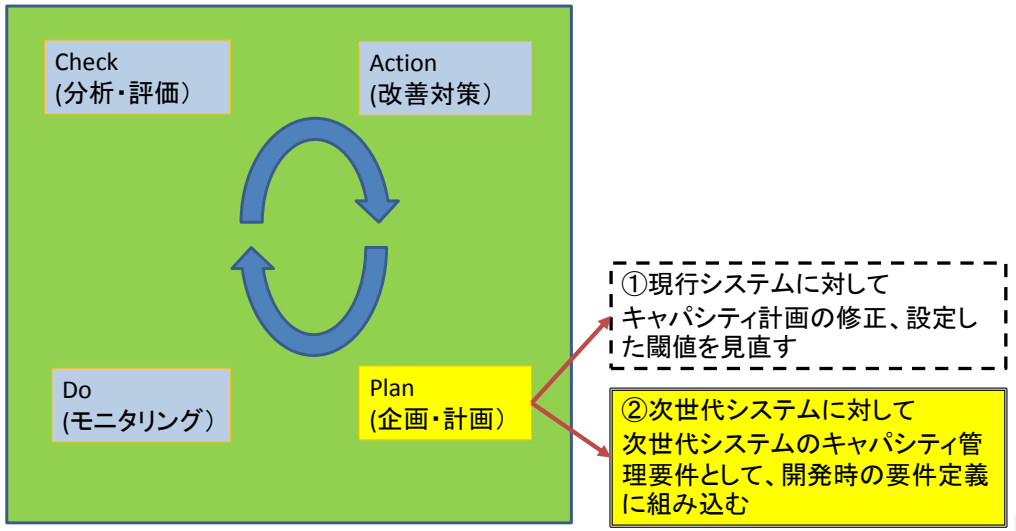
G14: 設計時に定めたキャパシティ管理項目は、環境の変化にあわせて見直すべし！

【問題】A社のシステムは、以前から一般的な増加とともに、突発的な事象によるデータ量の急増がみられた。このA社のシステムにある日、処理能力を越えた注文が殺到し、サービスの時間が短縮となった。



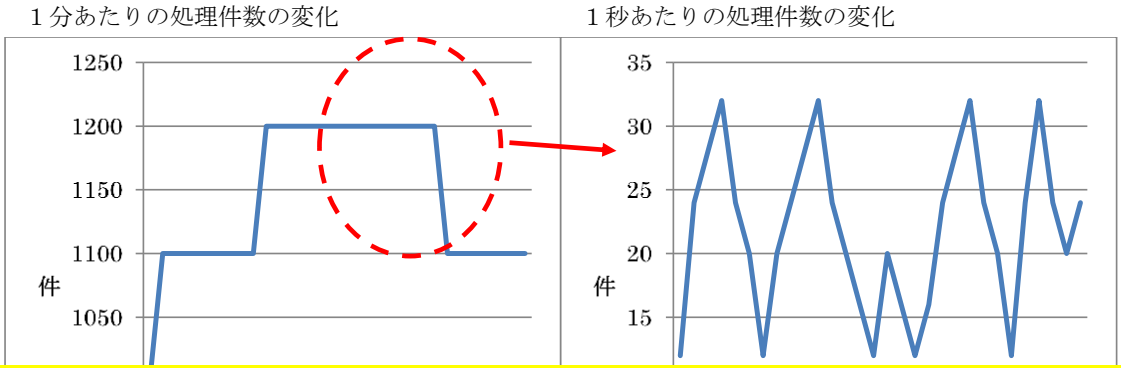
【原因】設計時に定めた監視する時間間隔（キャパシティ管理項目）では十分にシステムを監視できていなかった。

【対策】①現行システムに対しては、キャパシティ計画の修正と設定した閾値の見直しを行う。
②次世代システムのキャパシティ管理要件として、開発時の要件定義に組み込む。



G14: 設計時に定めたキャパシティ管理項目は、環境の変化にあわせて見直すべし！

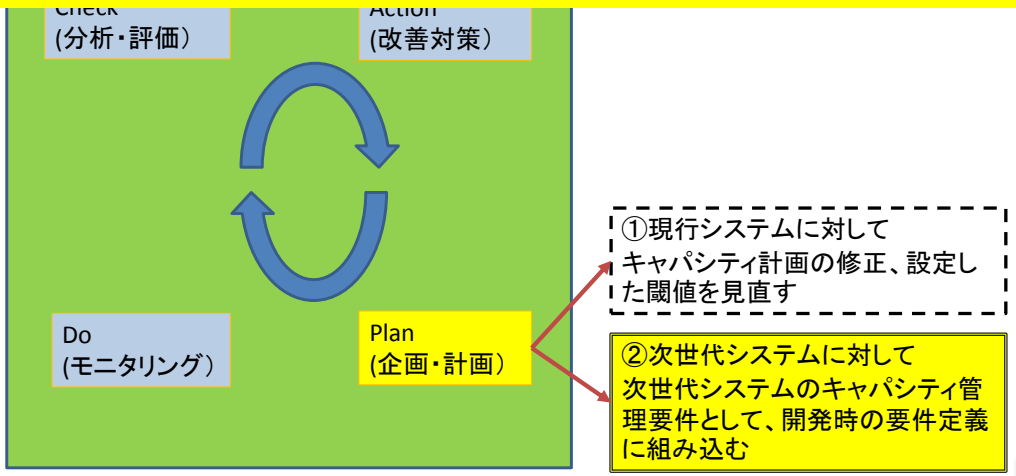
【問題】A社のシステムは、以前から一般的な増加とともに、突発的な事象によるデータ量の急増がみられた。このA社のシステムにある日、処理能力を越えた注文が殺到し、サービスの時間が短縮となった。



- **毎分のトランザクション量監視から毎秒に変更し、きめの細かい監視・予測が可能**
- **日々の運用変化への耐性を持つシステムは突発的な変化にも対策・対応が可能**

できていなかった。

【対策】①現行システムに対しては、キャパシティ計画の修正と設定した閾値の見直しを行う。
②次世代システムのキャパシティ管理要件として、開発時の要件定義に組み込む。



G15: 保守作業は「予期せぬ事態の発生」を想定し、サービス継続を最優先として保守作業前への戻しを常に考慮すること

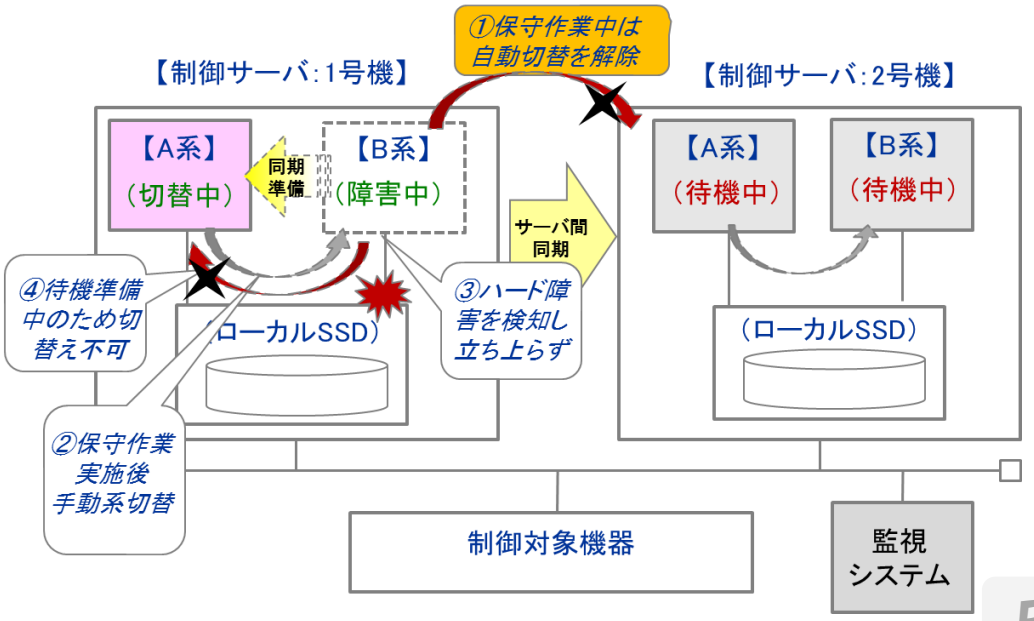
〔問題〕堅牢なホットスタンバイ4重化システム（１号機（稼動系）と交代系（2号機）、各々Ａ系とＢ系の構成）において、保守作業実施後、Ｂ系に切り替えたところハード故障が発生しサービスが10分程度停止した。

〔原因〕1号機から2号機への**自動切替えを解除**していたためスタンバイしている2号機に瞬時に切り替らなかったもの。手動で2号機へ切り替えるまでサービス停止した。なお、1号機内Ｂ系⇒A系の自動切替えは同期準備中でまだスタンバイ状態になっていなかった。

〔対策〕保守作業実施時にも1号機から2号機への自動切替えは解除しないこととし、保守作業マニュアルを見直した。

〔教訓〕システムの保守作業を実施する場合、**サービス継続を最優先とし、予期せぬ事象の発生を考慮した保守作業マネジメントが重要**である。

保守作業は時間との戦いである。



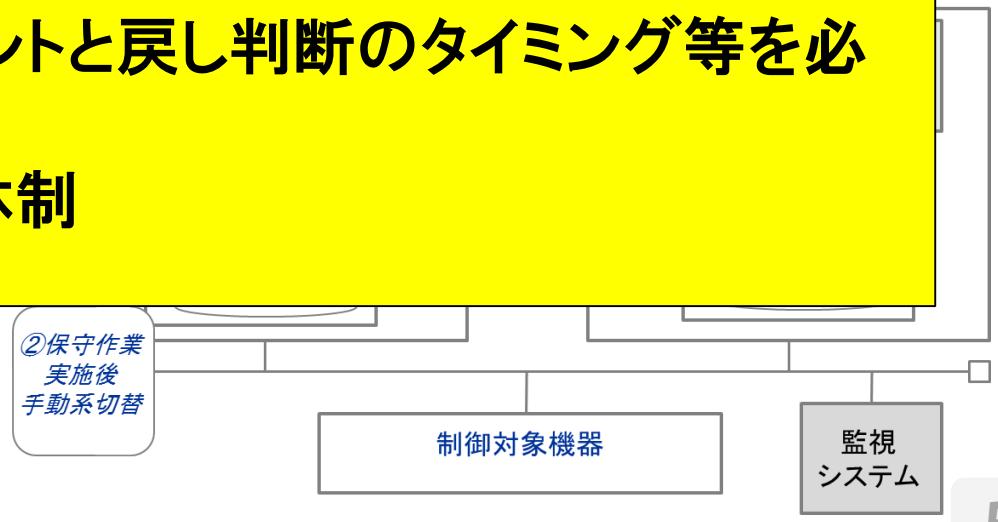
保守作業マネジメントはサービス継続の業務特性レベルに応じて実施する。
●サービス停止が許されない重要なシステムの場合：（ホットスタンバイ構成）
●数分間のサービス停止が許容できるシステムの場合：（コールドスタンバイ構成）
●夜間・休日などサービスを停止可能な時間帯枠が確保可能(通常はこのケースが多い)な場合：

G15: 保守作業は「予期せぬ事態の発生」を想定し、サービス継続を最優先として保守作業前への戻しを常に考慮すること

〔問題〕堅牢なホットスタンバイ4重化システム（１号機（稼動系）と交代系（2号機）、各々Ａ系とＢ系の構成）において、保守作業実施後、Ｂ系に切り替えたところハード故障が発生しサービスが10分程度停止した。

〔原因〕1号機から2号機への**自動切替えを解除**していたためスタンバイしている2号機に瞬時に切り替らなかったもの。手動で2号機へ切り替えるまでサービス停止した。なお、1号機内Ｂ系⇒A系の自動切替えは同期準備中でまだスタンバイ状態になっていなかった。

・オンラインサービス開始を最優先し予期せぬ事態が発生する場合を想定して、
・旧システムへの戻し時間を確保した上で
・保守作業計画書にチェックポイントと戻し判断のタイミング等を必ず盛り込む
・非常時の戻し判断の意思決定体制



●数分間のサービス停止が許容できるシステムの場合：（コールドスタンバイ構成）
●夜間・休日などサービスを停止可能な時間帯枠が確保可能(通常はこのケースが多い)な場合：

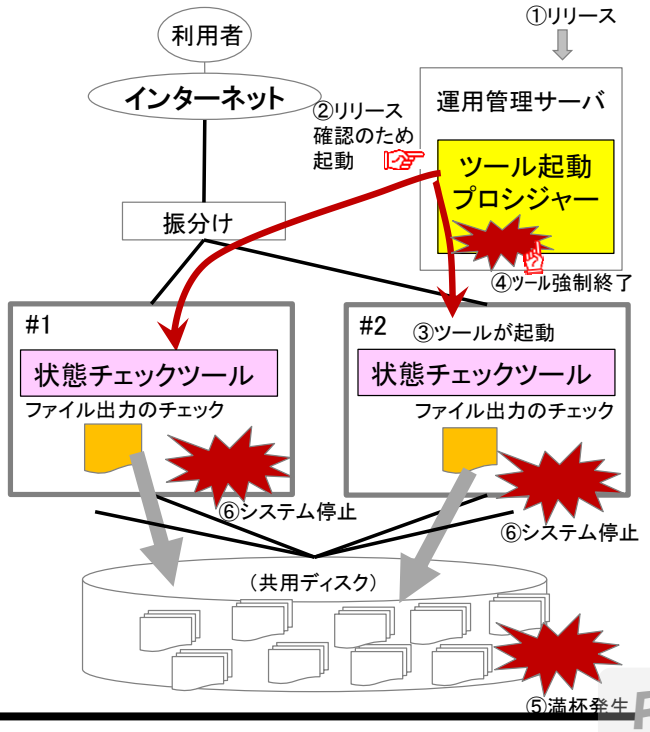
G16: 本番環境へのリリースは、保守担当が無断でできないような仕組みを作るべし！

〔問題〕いつでもどこからでもサービス要求を受付可能な24時間運転のWebシステムが、システム障害が発生したことにより、数十分間、サービス要求を受け付けられない状態となった。

〔原因〕直接の原因は、本番環境で実行した保守作業ツールを強制終了したことにより、想定外の事態となり、共有ディスクが一杯になってしまったため。また、保守担当は、今回の作業を運用改善作業の位置づけと考え、本来諮るべきユーザ企業主体のシステム変更会議に諮っていなかった。保守担当が、ユーザ企業の承認なしに本番環境で作業を行っていたことがそもそもの問題である。

〔対策〕本番環境での作業を、保守担当が無断でできないようにするため、以下の観点でルールを策定した。

- 作業実施にあたっては、ログインIDの払い出しを受ける。
- ログインIDの払い出しを受けるため、ユーザ企業主体のシステム変更会議に諮る。
- 会議には、作業手順書を含む保守作業計画書の提出を必須とし、記載内容について有識者を交えたレビューを行う。
- 承認された作業手順書以外のことは実施しない。



G16: 本番環境へのリリースは、保守担当が無断でできないような仕組みを作るべし！

〔問題〕いつでもどこからでもサービス要求を受付可能な24時間運転のWebシステムが、システム障害が発生したことにより、数十分間、サービス要求を受け付けられない状態となった

〔原
想定
回の
会議
いた

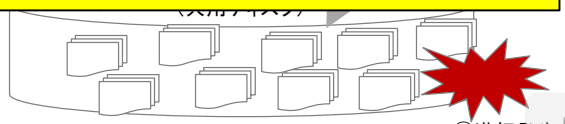
〔対
う(

・
・
・
・承認された作業手順書以外のことは実施しない。

・突然のシステム障害発生！
今日、保守作業の予定は入っていないはずだけれど、
誰か、何か本番環境で作業した？

⇒良かれと思って実施した作業がシステム障害を引き起こす

・改善であれ、調査であれ変更管理会議に申請し許可を受けること！
・本番環境のログインIDの管理は厳密に！



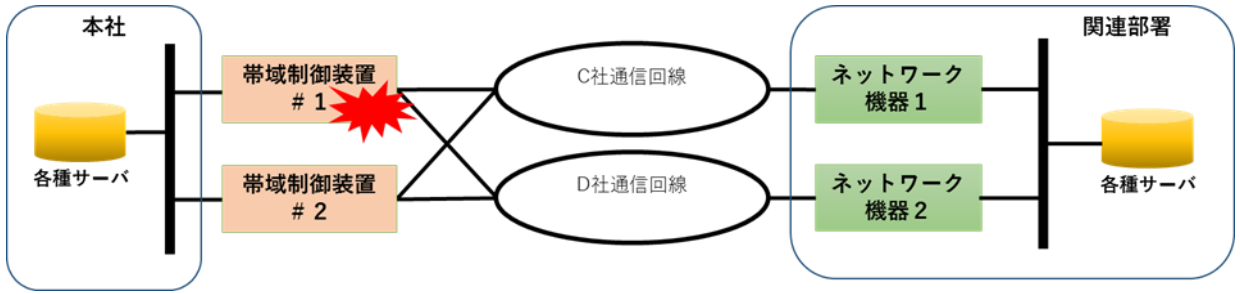
G17：サービスの重要度を識別し、それに応じた連絡体制や障害検知の仕組みを作れ

NEW!

〔問題〕帯域制御装置のメンテナンス作業を実施した際に、本社から関連部署に提供している通信が遮断され、停止が許されない重要度の高いサービスの運用が約30分にわたって停止した。その際、一部の部署への通信が不通になったが、装置自身は正常動作中と自己判断し警報を発しなかったため、異常に気づくのが遅れた。

〔原因〕メンテナンス作業の実施が直接の原因であることは明白であったが、作業の実施手順はこれまでと同様であり、これまでは問題なし。別環境を構築してテストをしても再現せず、どういう条件で発生するのかは不明のまま。したがって、再発の恐れあり。

〔対策〕重要度の高いサービスに対しては特段の障害発見・対応の迅速化が必要と考え、周辺機材の活用など運用監視方法を工夫して、装置自身とは別の観点から障害の発生を検知する仕組みを構築し、障害検知を強化した。さらに別種のエラーに備えて、発生時は利用者から運用管理者に直接連絡が入るよう、通常とは別の連絡系統を追加した。



〔教訓〕サービスの重要度を識別し、それに応じた連絡体制や障害検知の仕組みを作れ

G18：障害対策とは許容時間内の回復や停止中の業務継続まで具体化すること

〔問題〕オンラインシステムを構成するサーバの動作がエラーになり代替機に切り替わったが、切替え処理が正常動作せず、システム全体の動作が不安定になり、サービスが１時間停止。

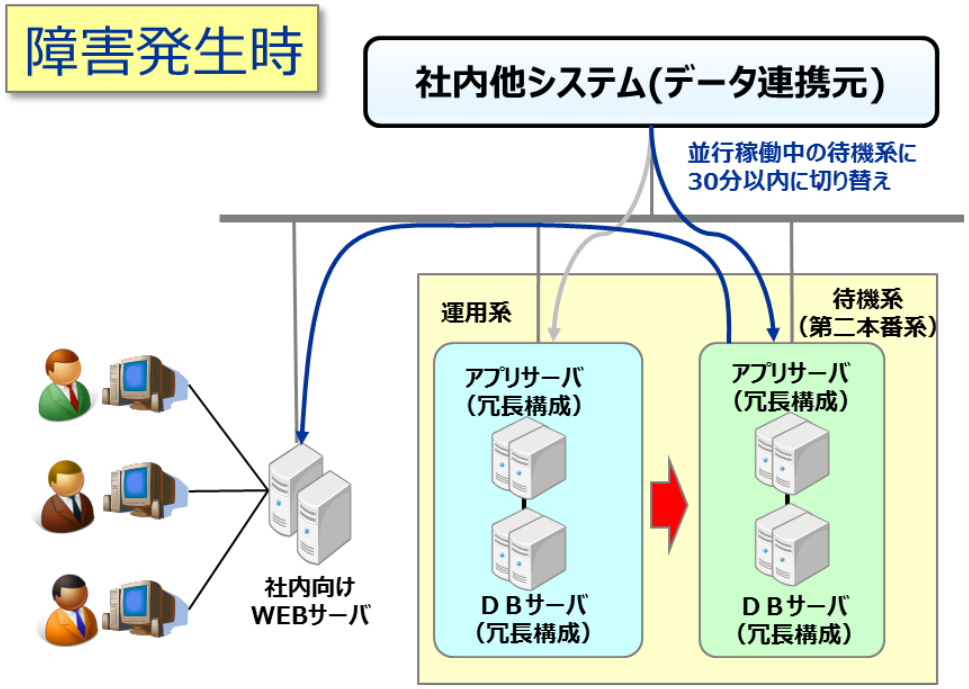
〔原因〕代替機への切替が失敗した直接の原因は通信制御ミドルウェアの潜在的な不具合であったが、このエラーで業務に影響が出た原因は、業務が許容できるシステム停止時間を明確に決定できておらず、その時間内に回復というゴールの設定と実現手段の準備ができていなかったこと。

〔対策〕業務サービス停止の抑止を第一義に、仮に停止した場合に業務側が許容できる時間の上限の設定と、上限までに切り替えられる第二本番環境の構築、仮にシステムが復旧しなかった場合の現場でのシステムを利用しないで実施できる対策・エンドユーザへの被害の軽減策を準備。

〔教訓〕ITシステムトラブルからの回復策の検討の要点

①システムが停止してから回復までの許容時間（業務遅延をリカバリできる時間）をサービス利用部門と合意し、その時間内での回復手段を構築する

②許容時間内の回復見込みがない場合に備えて、システムを使用しないでも現場で実施できる範囲の作業を決定し、周知・訓練する



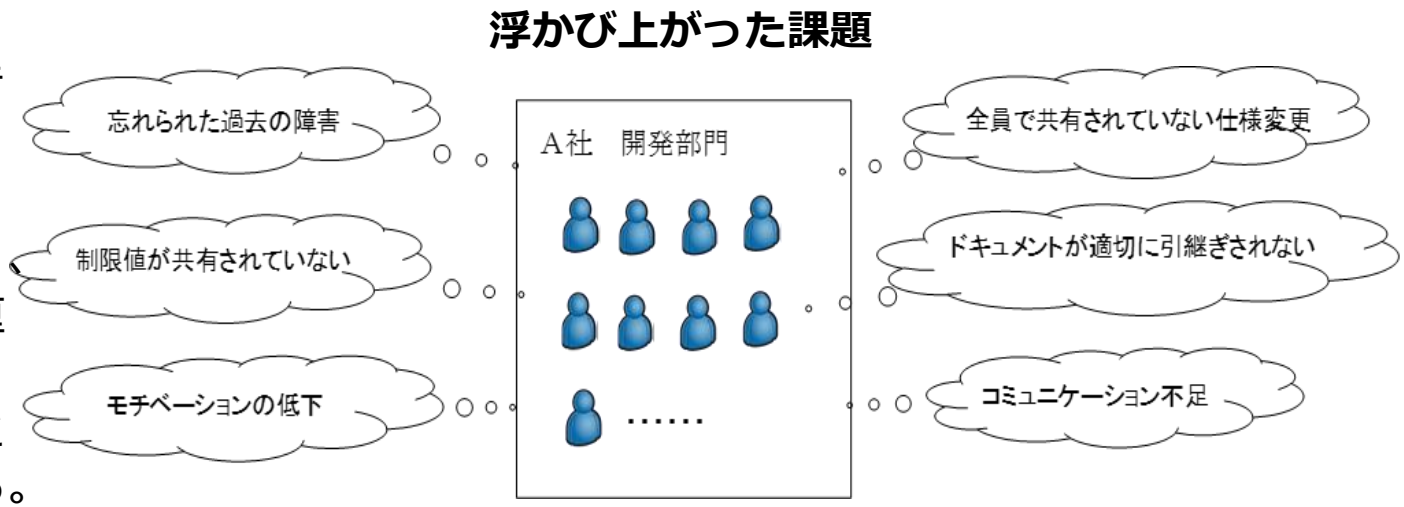
- 障害発生時には許容時間を設けて調査
- 時間内の対応見込みなしと判断した時点で第二本番系に切り替えて業務を再開
- 時間内に業務が回復できない場合には、あらかじめ用意しておいた、システムを使用しないで実施可能な範囲の業務を開始

G19: みんなで唱和！障害減らす教訓共有

〔問題〕制御装置の稼働系が障害になり、システム監視端末が、異常を検知した。システム運用者は、端末の表示メッセージからハードウェアの問題と思い、すぐに、制御装置を待機系に切り替えたが、切替え後も障害となった。システムを一旦リセットすることにより稼働系の制御装置は正常に戻った。

〔原因〕制御装置が持つ制限値は、システム構築当初から存在していたものだったが、この制限値があることを知っていたのは、この個所の設計、製造を受け持った一部のメンバだけ。そのため、この制限値が開発チーム全体で共有されず、障害復旧が遅れてしまった。

〔対策〕「担当者しか知らない」仕様があったために起きた障害を2度と起こさないために、A社の開発チームは、「どこに問題があったのか」「対策は何か」等、議論を重ね、アイデアを出し合って、原因、対策を教訓としてまとめ、毎回全員で唱和している。



〔教訓〕システム障害を未然に防ぐのは、そのシステムに参画しているひとりひとりにかかっている。全員で唱和することは、チーム内のコミュニケーションの円滑化とモチベーション向上のために有効である。

G20：「システム運用環境変更時の品質向上」は正攻法の成功事例に学べ!

NEW!

〔問題〕ある年、基幹業務システムの運用環境の変更作業のミスにより、提供するサービスを一時的に停止させた結果、自社および販売店の業務に大きな影響を与え、お客様にご迷惑をおかけする事態が複数回発生

〔原因〕これまでに発生したトラブルの分析結果から、発生原因は単一ではなく、①システム変更リリース手順の審査・承認プロセス、②過去と同一原因のトラブルの再発防止策、③システム変更担当者の作業に対する意識、のそれぞれに問題があることが判明

〔対策〕各々の問題に全社で取り組むこととし、品質保証室を設置して専任で担当させた
① システム変更の品質確保に向け作業計画の承認プロセスを再構築
② 過去の失敗を教訓として再発防止するシステム部門全体での品質向上サイクルを確立
③ 障害の原因になるような行動をとらないよう作業者の意識醸成活動を開始

システム変更時トラブルの原因分類

分野	原因分類
1.ハードウェア	① システムリソース不足
	② 機器障害
2.ソフトウェア	③ システム変更が複雑化し誤りを誘発
	④ 人為的な単純ミスによる作業誤り
	⑤ 変更内容に対する組織的なチェック不足
	⑥ 熟練した基盤要員の不足による実施内容の誤り
	⑦ 万が一システムが停止した時の対策準備不足
	⑧ 過去と同じ原因によるトラブルの再発

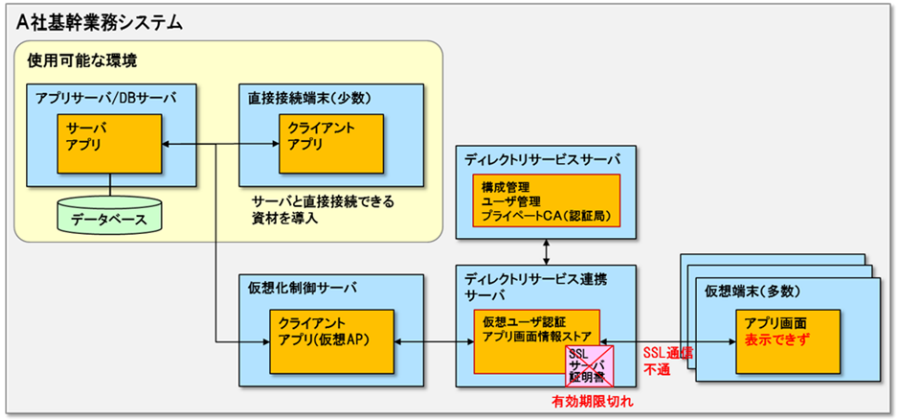
施策例：システム変更審査の標準化

	従来の実施内容の問題点	改善点
1	作業手順など技術的な観点でのレビューが中心	システム開発・運用責任者と品質保証室をレビュアとして、システム変更実施タイミングの妥当性、連絡体制、トラブル時のリカバリ計画など、サービス運用環境への影響をチェック
2	チーム間でレビュー品質にバラツキがある	実施規準にしたがって漏れなく実施し、関係者がそろって統一されたチェック項目を審査 A) 案件の概要 B) 目的・経緯 C) リリース日時 D) 実施スケジュールと体制 E) 実施内容（過去実績、システム変更内容、対応期限、災害対策有無、検証方法、初日の監視） F) 影響範囲（性能、キャパシティ、業務への影響、業務停止） G) トラブル発生時のリカバリ計画（作業時間、影響、トラブルであるかどうかの判断規準、判断者、サービス開始後にリカバリする場合の内容）
3	実施タイミング、作業により影響を受けるシステムの範囲、トラブル発生時のリカバリ計画などが網羅的にチェックできない	
4	レビュー関係者の判断によっては、作業を実施することが部門内で止まり、報告する必要があるレベルの作業が全体を管理する部門に伝達されないことがある	

G21：サーバ証明書等の有効期限の確認方法を工夫せよ

〔問題〕ある朝、Ａ社の窓口業務の現場で、すべての仮想端末上で基幹業務が起動しないという現象が発生。トラブル収束までの間、仮想端末以外で業務を開始したが、処理の順番待ちが多数発生し、一部の顧客が順番を待ちきれずに帰る

〔原因〕直接原因は、仮想化端末認証サーバのSSLサーバ証明書の有効期限切れだが、根本原因は、システム開発／保守委託先のサーバ構築担当者が、2年後に更新が必要な証明書を問題になったサーバに組み込んだことを、Ａ社にも委託先の保守担当者にも引き継ぎをしていなかったこと



〔対策〕自社および開発・保守委託先会社の両当事者を集めて根本原因の分析と採り得る再発防止策の検討を行い、以下の再発防止策を実施

- サーバ証明書を期限切れを待たず毎年定期的に更新するよう運用変更
- 他のサーバ証明書やパッケージライセンスの監視に漏れはないか確認
- すべての証明書の有効期限、管理担当者等を台帳管理し、定期的に確認
- サーバ証明書の定期更新を保守委託先の作業として契約時の仕様書に明示
- 今後、新たにシステムを構築する際には、開発委託先に対して委託元からも自主的に引継ぎ事項の有無を確認（引継ぎ事項チェックリストにサーバ証明書の組み込み有無を確認事項として提示し、調査結果を相互確認することにより漏れを抑止）

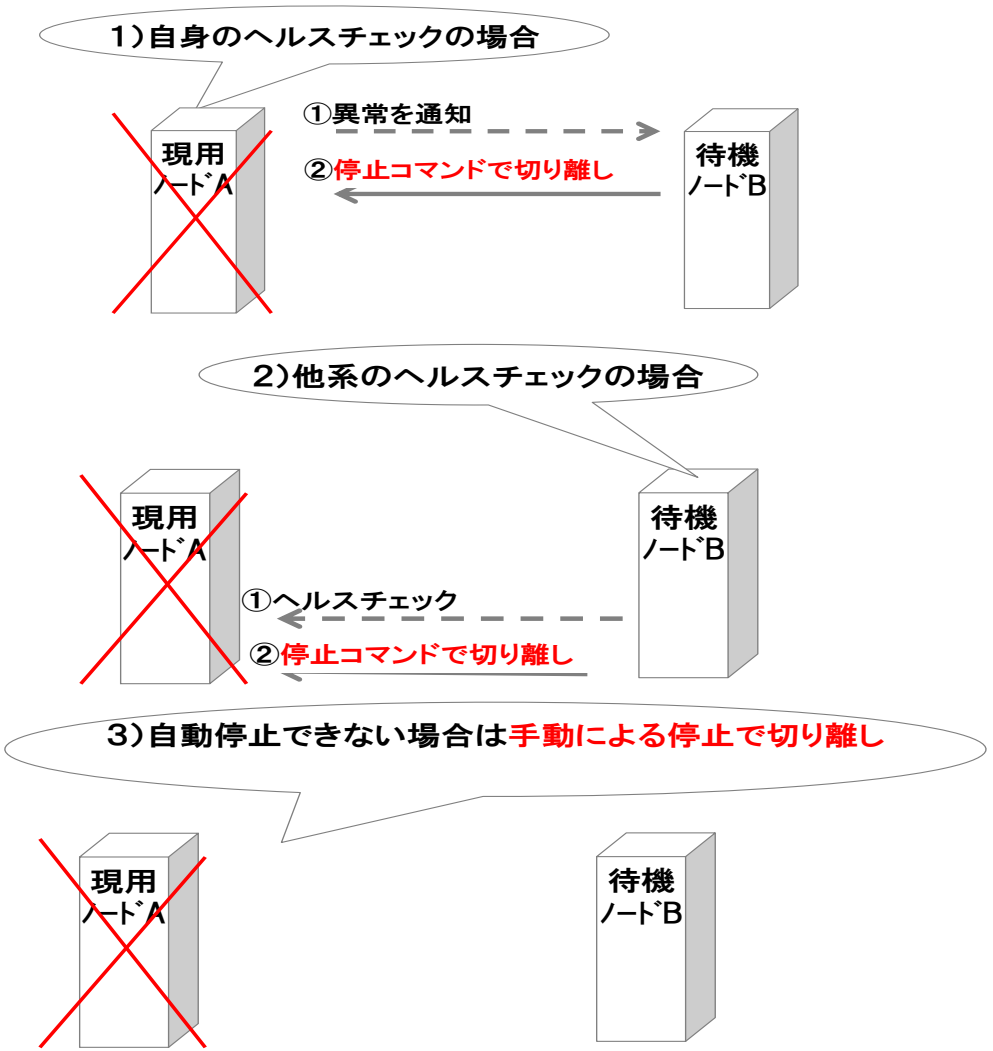
T1：サービスの継続を優先するシステムにおいては、疑わしき構成要素を積極的にシステムから切り離せ（“フェールソフト”の考え方）

業務内容に基づいて、システム毎にポリシーを作成した上で、フェールソフトの考え方を適用。

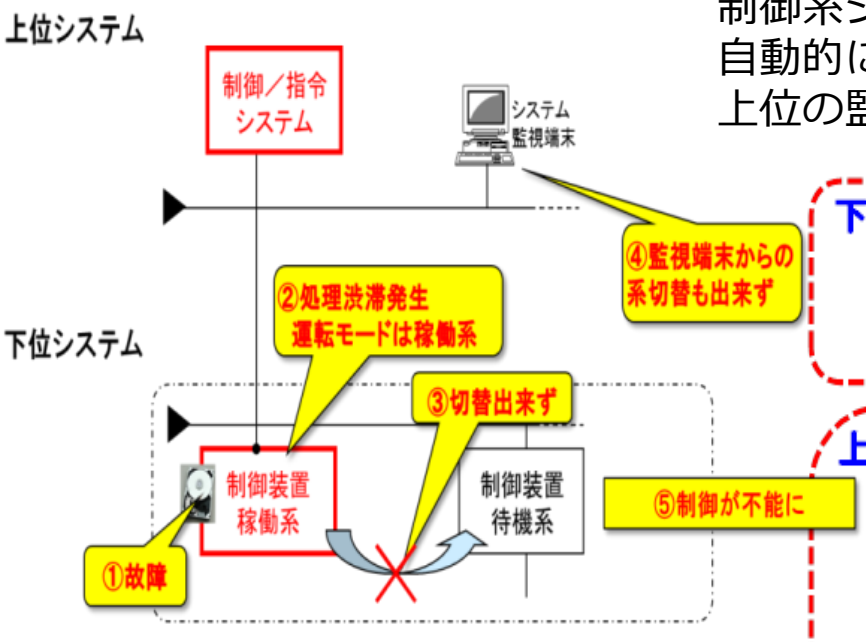
ハードウェア機器の故障、ソフトウェアの処理プロセスの異常等があった場合には、その部位を積極的に停止させることでシステムから切り離す、場合によってはその系全体を放棄するといった考え方のもとに処理・対応する。

一方、そのような状況下で一部の部位や系をシステムから切り離しても、システム全体としてのサービスは継続できるように、フェールソフトの考え方に基づいて設計・運用する。

というのは、機器やソフトウェアそれぞれの動作継続を優先しすぎてしまうと、予期せぬ障害の場合にサービスへの影響がかえって大きくなってしまう場合があり、サービスの継続を優先させるためには、むしろ積極的に関連する部分をシステムから切り離す方が良い場合が多いからである。



T2：蟻の目だけでなく、システム全体を俯瞰する鳥の目で総合的な対策を行うべし！



制御系システムの下位にある制御装置の稼働系に故障が発生し、自動的に待機系に切り替わるところが切り替わらず、上位の監視端末からの指示による系切替えを実施したが、失敗。

下位システム対策（蟻の目）

対策A-1	システムで故障検知（下位の制御装置）	・待機系で故障検知しシステム停止 ⇒待機系が稼働系となる

上位システム対策（全体を俯瞰した鳥の目）

対策A-2	システムで故障検知（上位の制御装置）	・上位装置による故障検知 ⇒稼働系を停止させ待機系に切替え指示
対策B	ソフトで故障検知（上位の情報システム） ・プログラム異常 ・処理渋滞 ・メモリのリソース不足	異常現象を判断するソフト機能の追加 ・周期監視より異常を検知 ・動作状態より異常判断 ⇒構成制御の実施
対策C	手動による系切替	・リモートでの手動による異常系停止機能を強化

障害が下位で起きた場合、障害を起こした下位だけの対策を考えがちであるが、蟻の目（下位）の対策だけでなく、システム全体の視点で、鳥の目（上位）対策を活用することでシステムの安定稼働が保たれる。

T3：現場をよく知り、現場の知識を集約し、現場の動きをシミュレートできるようにすべし！

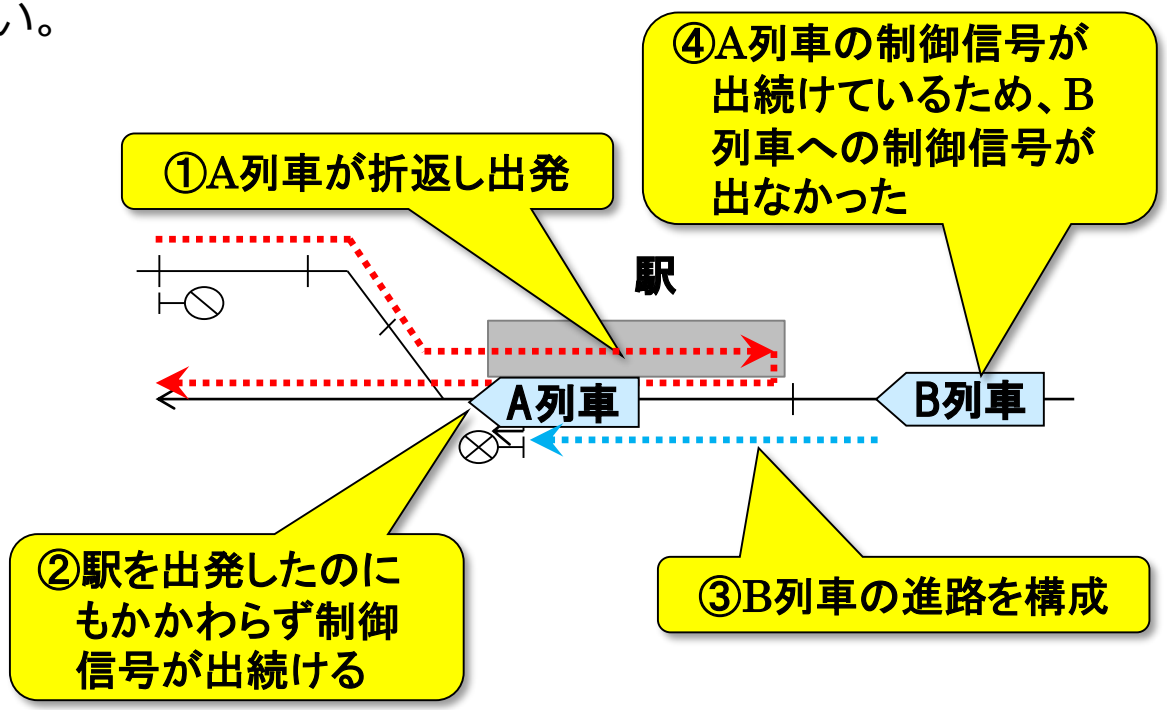
特定ケースで制御信号が正しく出力されず、列車が停止

【原因１】有識者（ベテラン社員を含む）による以下の機能確認を行っても、まだ洗い出せていない機能が存在する。（機能要件漏れ）

【原因２】列車の動き、システムの動作などを総合的にテストできる環境、つまり、組込みソフトウェアを持った制御システムと列車などの動作の全てのテストが行えるテスト環境ができていない。

【原因１】については、一度設計された「機器の動き（列車の運転）」のパターンを知識データベースとして蓄積し、そこに、追加登録していく。

【原因２】については、制御系システムのシミュレーション・システムの開発を行うためには、現実の制御装置のプロセスを分かりやすく可視化し、プロセスの骨子を見極めて「モデル」化（モデリング）する。

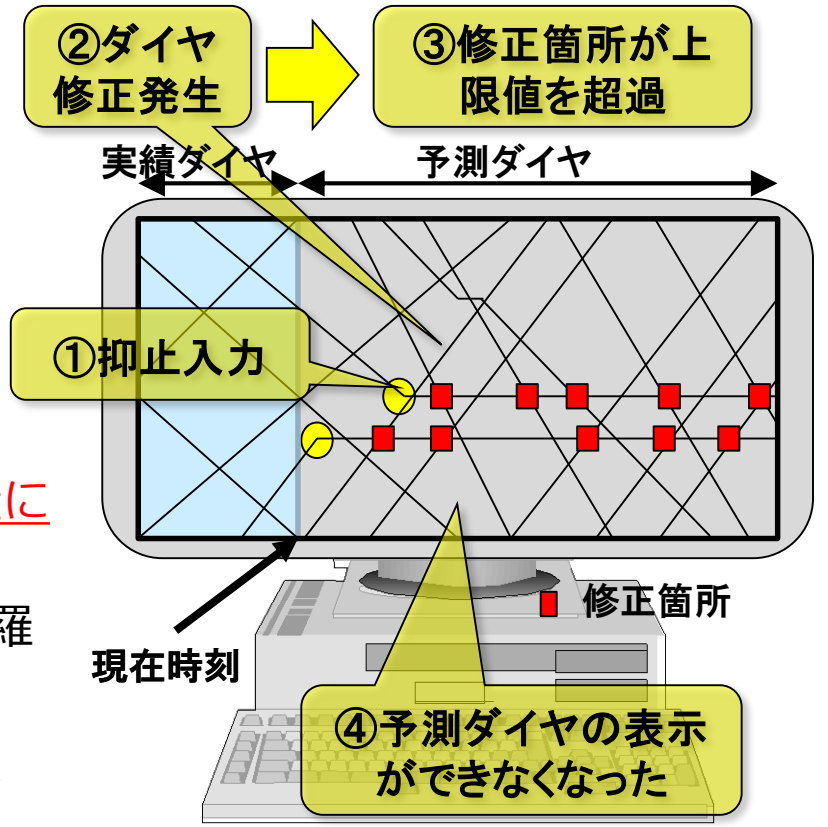


T4：システム全体に影響する変化点を明確にし、その管理ルールを策定せよ！

表示項目数がシステムの上限値を超えたため、全画面表示が消え，オペレータが混乱
システム構築当初から決まっていた上限値について、外部仕様変更に伴う見直しを未実施
原因の本質は、全体に影響する変化点（この場合、予測時間、列車運転本数）が不明確

【原因１】予測時間を4H⇒24Hに変更した際、そのような要件変更があったにもかかわらず、「修正箇所数」の上限値の増加などシステム全体の機能要件変更を未実施

【原因２】列車の本数が年々増加しており、本来ならば（運転本数の増加の都度）上限値を超えた際のシステムの挙動を見直す必要があったにもかかわらず、未実施



【対策】 制御系システムの変化点の管理ルールを明確にし、そのルールを守る仕組みを構築

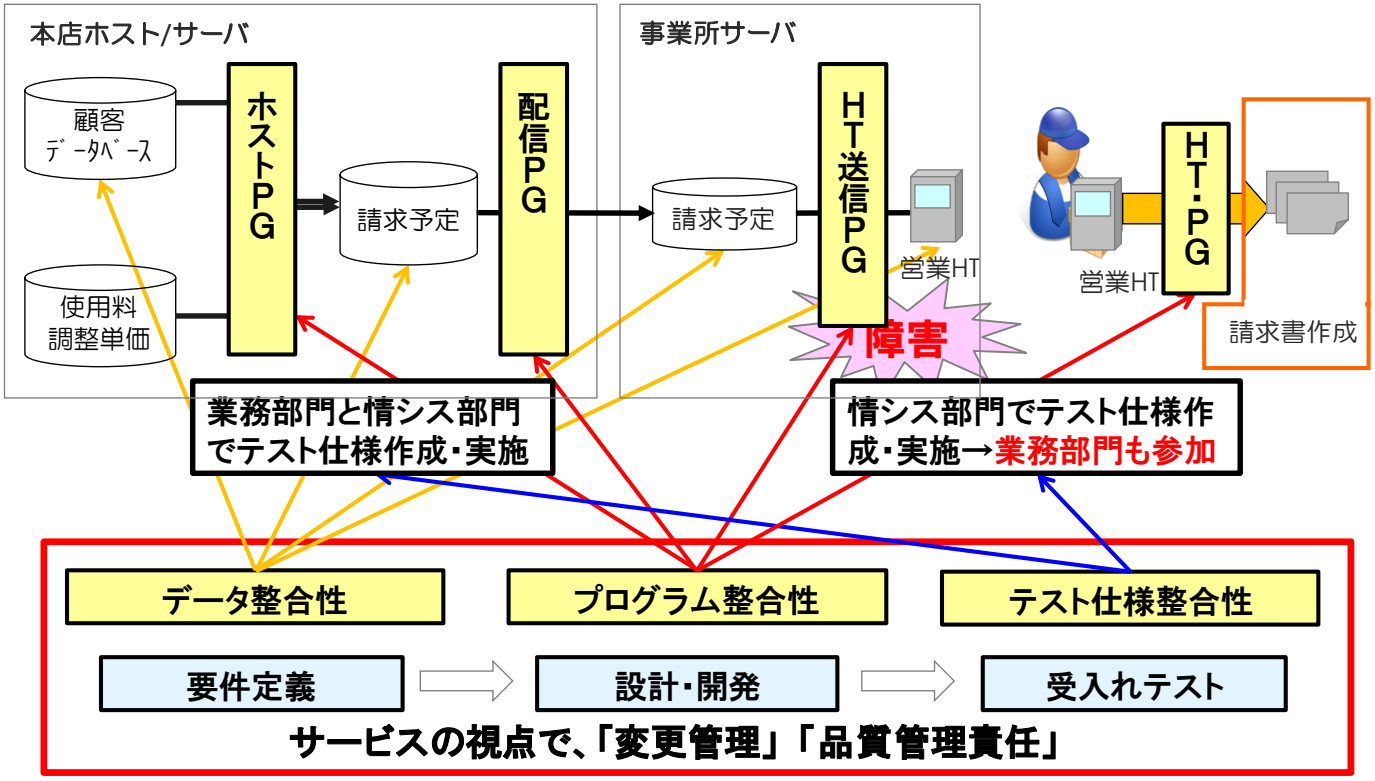
- システムが監視・制御する対象と仕様の変化点を網羅
- 変化点管理のルールとそれを守る仕組みを構築
- 変化点管理で使用する管理指標を関係部門で共有し、「変化点の見落とし」を防止

T5：サービスの視点で、「変更管理」の仕組み作りと「品質管理責任」の明確化を！

本店ホスト／サーバから請求データを端末に転送し請求書を印刷するシステムにおいて、端末として、営業員が持ち歩くHT（ハンディ・ターミナル）を新規に導入したところ、そのシステムから出力される請求書の金額が誤ったまま顧客に渡ってしまい、個別謝罪・請求書の再発行に追われた.

システムへの新たな要件追加、使用方法の変更があると、今まで正常に稼働していたシステムが突如障害となる（追加により未使用・未確認のロジックが使われ，不具合が顕在化）

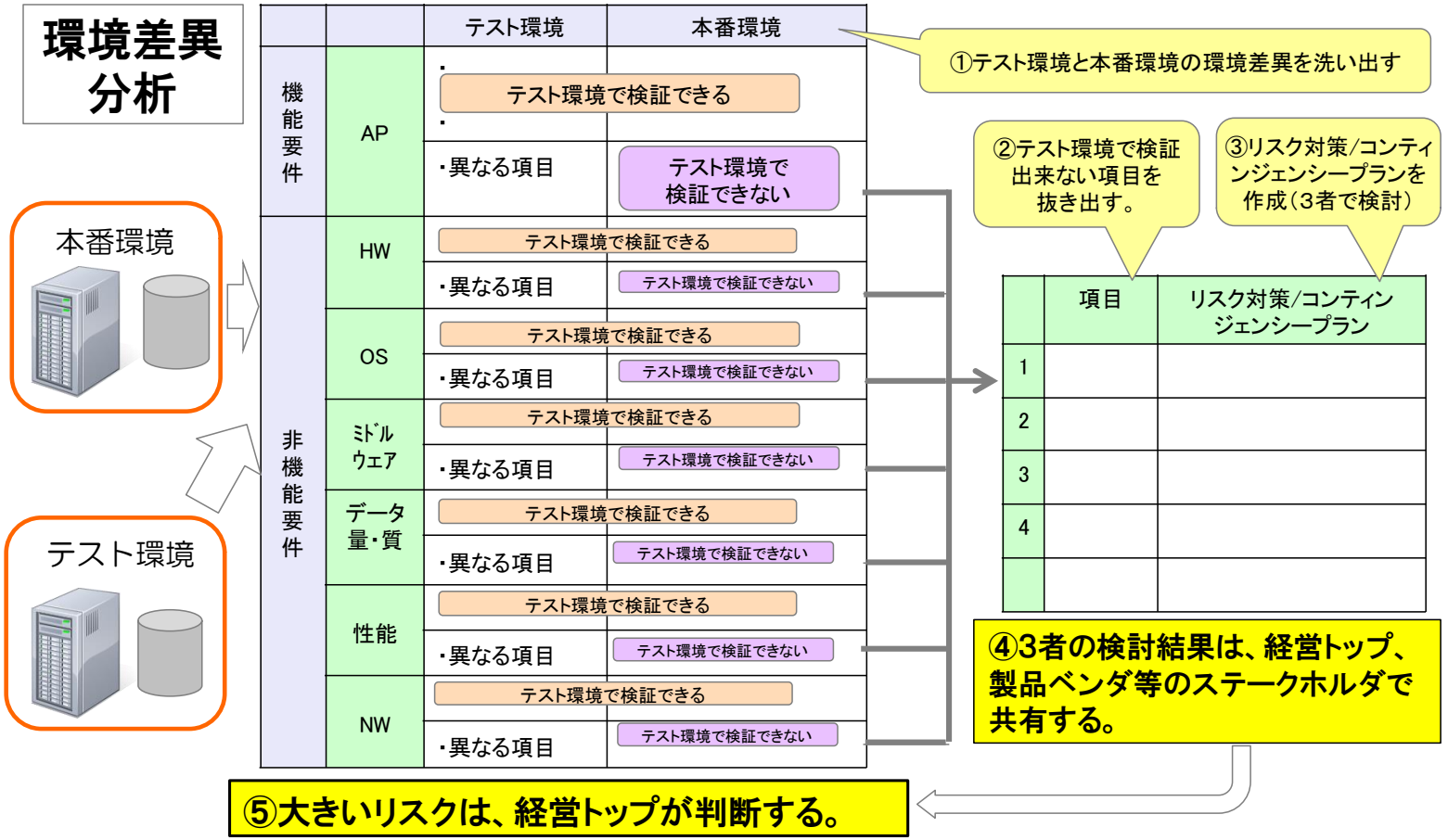
変更があった時にシステム全体のプログラム、データ、テスト仕様の整合性を保つための変更管理を確実に実施
システム全体の整合性を確認する人を決め、品質管理責任を明確にし、開発フェーズ毎の検証を実施



T6: テスト環境と本番環境の差異を体系的に整理し、障害のリスク対策を練る

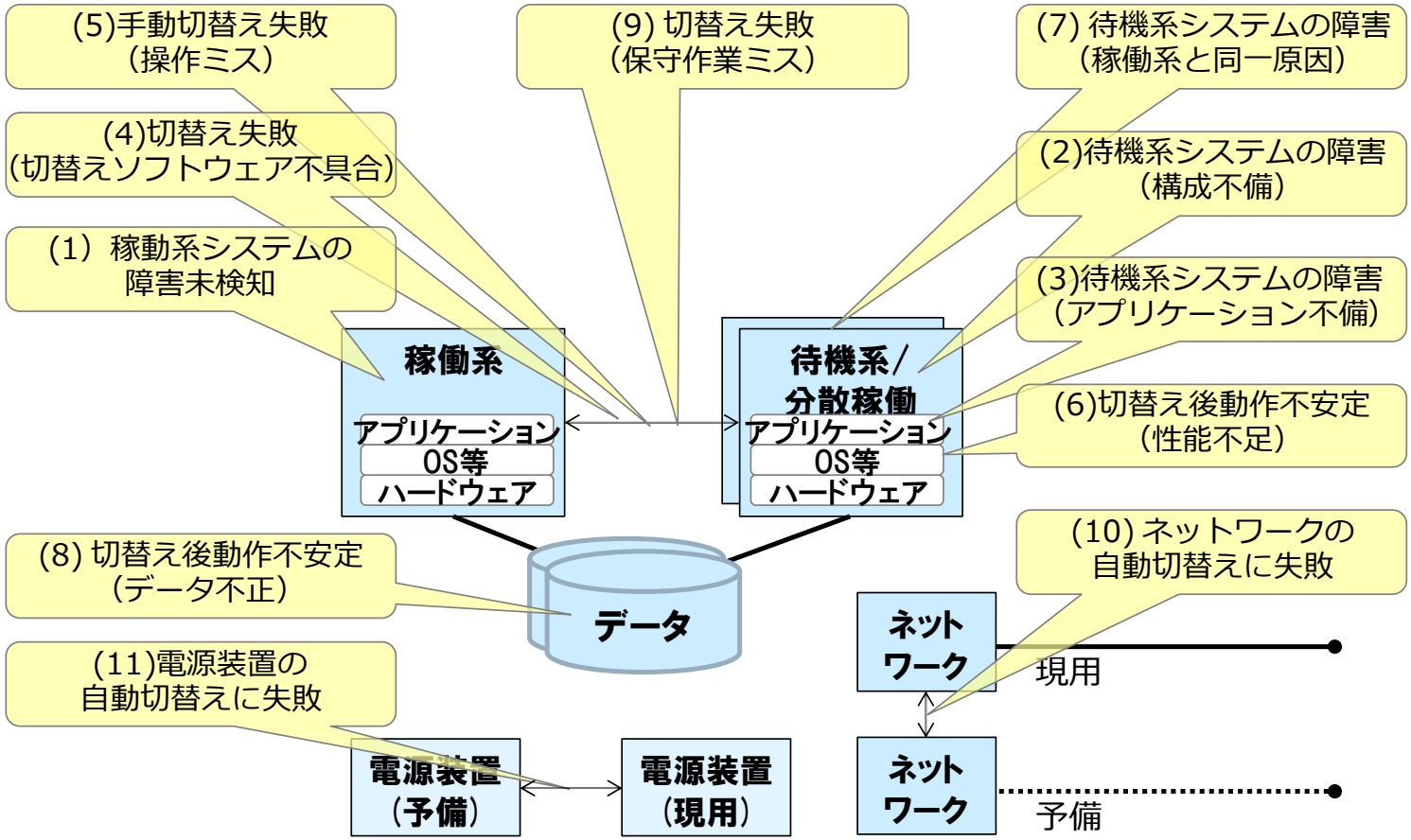
テスト環境と本番環境とに相違があり、テスト環境でうまくいったソフトウェアのリリースであるが、本番環境で障害が発生

①テスト環境と本番環境の差異分析
②テスト環境で確認できない項目、機能に対し、関係者でリスク分析
③リスク分析結果を基に、コンティンジェンシープランを作成
④本番環境のリスクをステークホルダで共有
⑤大きいリスクは経営トップが判断



T7：バックアップ切替えが失敗する場合を考慮すべし

冗長構成を取っているにも関わらず、バックアップ切替えが失敗しシステム障害となるケースが多い。下図に示されるさまざまな失敗原因にあらかじめ配慮してシステムの開発・運用を行うことにより、過去に発生した障害と類似の障害の発生を防ぐことができる。

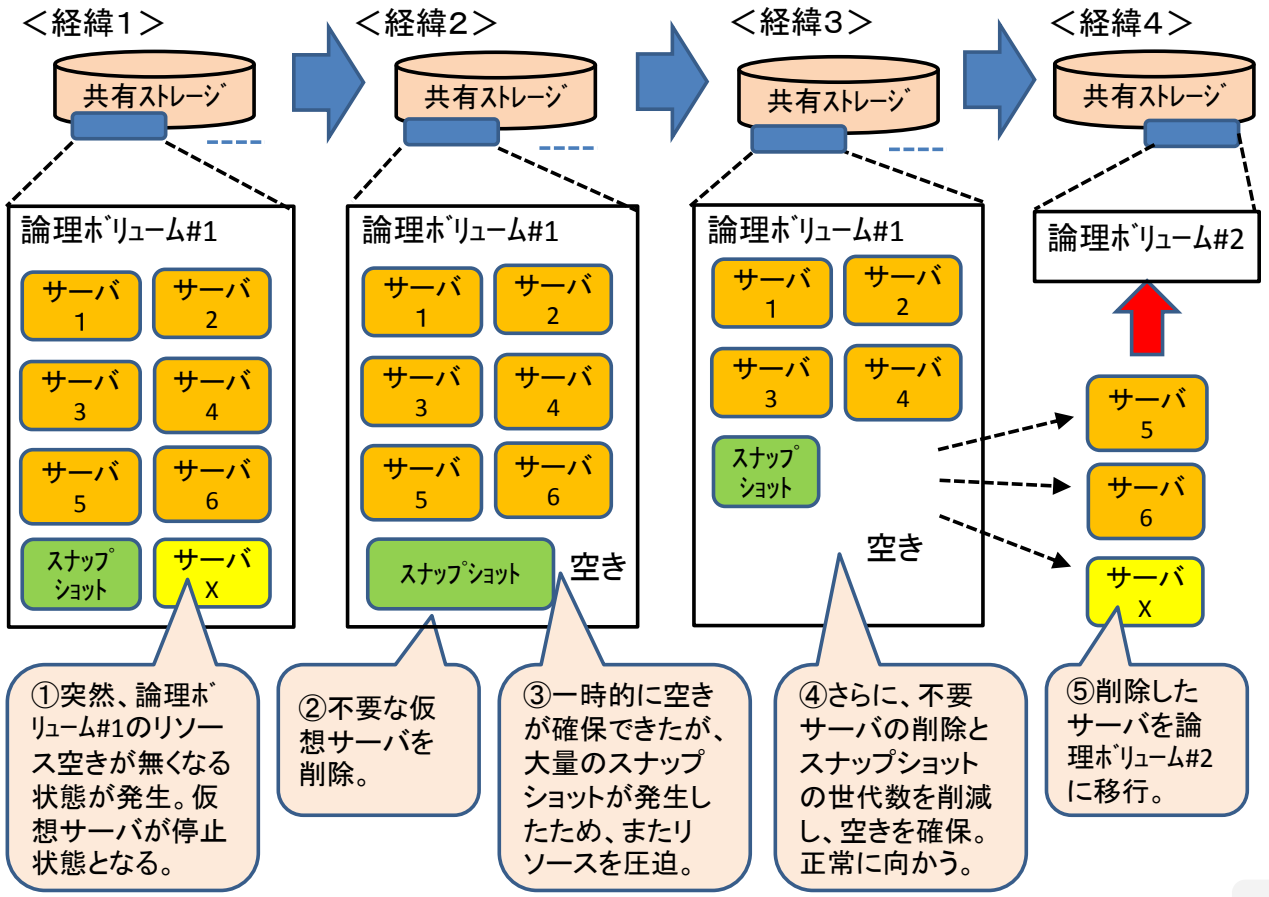


T8: 仮想サーバになってもリソース管理、性能監視は運用の要である

【問題】プライベートクラウドシステムのサービスが数時間停止

【原因】直接原因は，運用担当者のボリューム割り当てミス．根本原因は，仮想サーバ環境での運用要件の未整理．リソース管理，性能監視が不十分、かつ復旧時の対応方法も十分に把握せず．

【対策】
- 物理サーバを稼働サーバに移行する際のリソース管理、性能監視プロセスの策定
- 仮想化によるオーバーヘッド増大の見積り徹底
- 障害対応マニュアル等の整備と，要員の教育・訓練

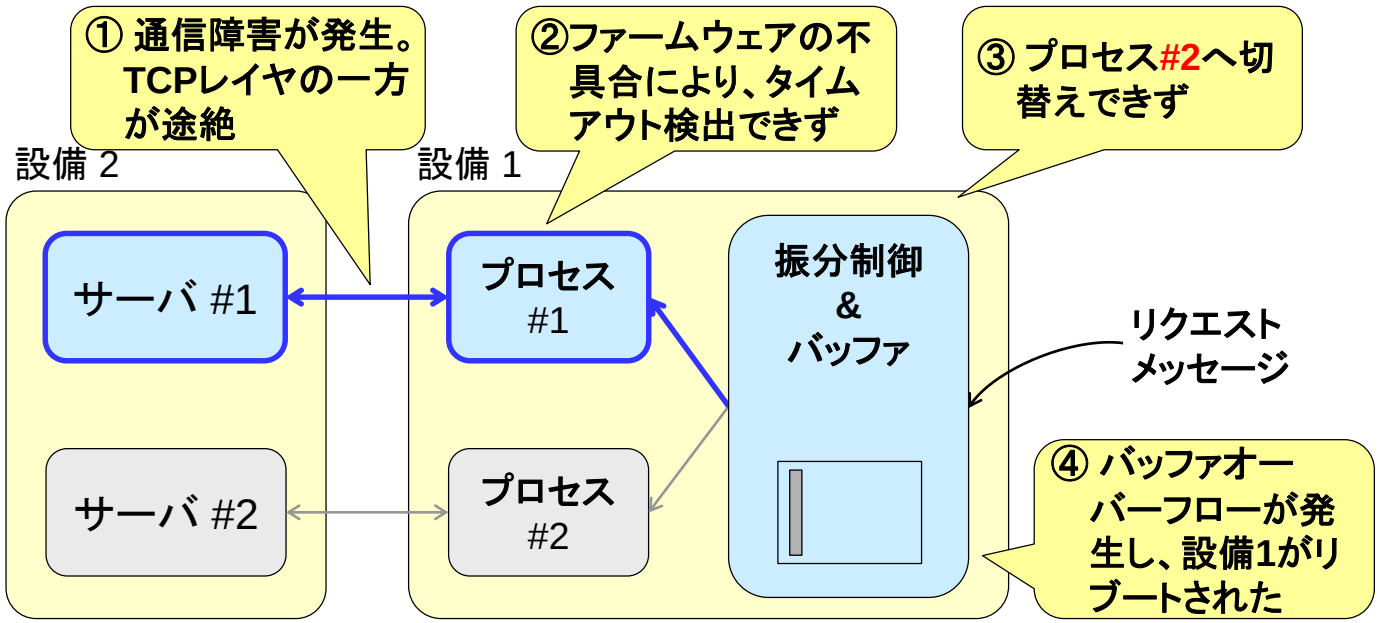


T9: 検証は万全？それでもシステム障害は起こる。回避策を準備しておくこと

【問題】2設備間の通信障害発生時に片方の設備上のプロセスの不具合が発現して動作を停止し、これを検知した同設備上の振分制御が他のプロセスへ切り替えようとしたがうまくいかず、処理待ちバッファのオーバーフローにより同設備がリブートされた

【原因】当該通信障害のケース（稀有なもの）を検証しておらず，システムの検収時点では、調達側、供給側とも当該ケースの存在を認識していなかった.

【対策】・タイムアウト前にオーバーフローしないよう，バッファサイズを拡大。
・サービス復旧のための作業手順を整備。
・対象システムの重要性に応じ、検証に費やすことのできる時間と労力は制約される。常に不具合が潜在しているとの前提に立ち、業務の継続性を確保することが重要

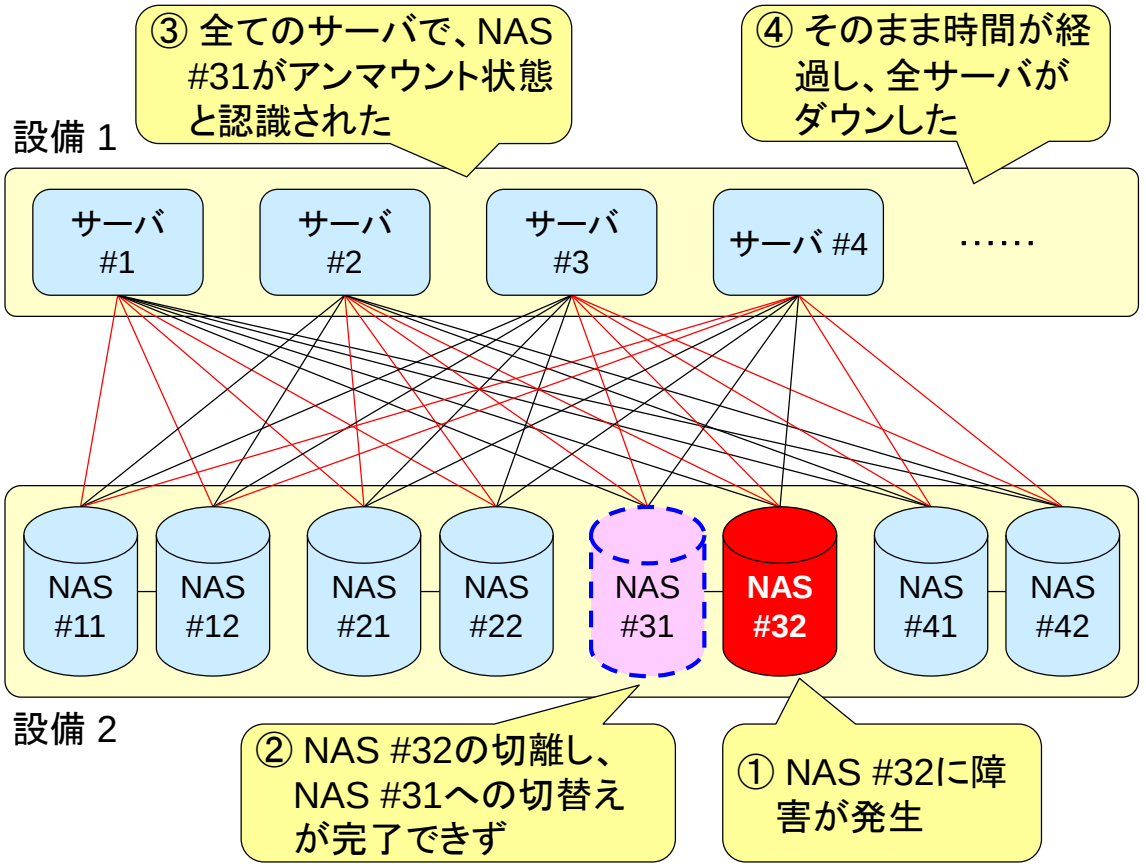


T10: メッシュ構成の範囲は、可用性の確保と、障害の波及リスクのバランスを勘案して決定する

【問題】各サーバが4ペアのNASと接続されたフルメッシュ構成のシステムにおいて、ある1台のNASの故障に起因して、全サーバがダウンした。

【原因】NAS制御ファームウェアの不具合により、故障したNASの切り離しに失敗し、この局所的な障害が、フルメッシュ構成のシステム全体に波及。根本には、フルメッシュ構成のリスクについての検討不足。

【対策】メッシュ構成を見直し、1サーバにつき2ペアのNASのみを接続するグルーピング化（図の赤色の接続）し、併せてトランザクションの振分け論理も変更。また、可用性の担保のため、迂回時の性能劣化防止用のBCP設備を増強。

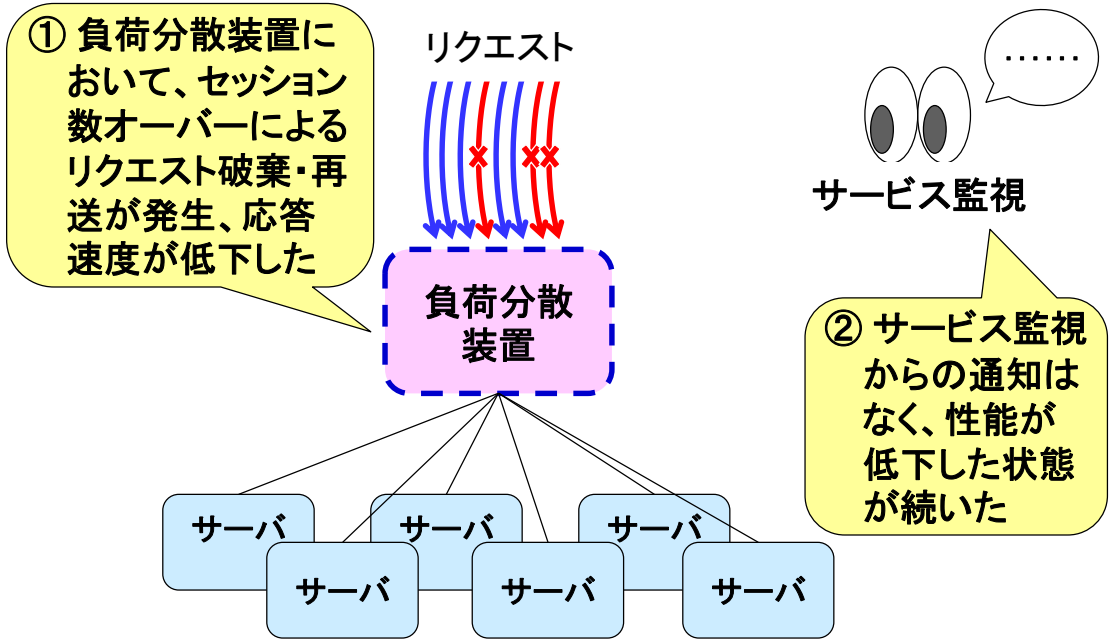


T11： サイレント障害を検知するには、適切なサービス監視が重要

【問題】Webサービスでサイレント障害（明示的に障害が検出されていないにもかかわらず性能が劣化）が発生した。外部からの指摘があるまで発見できず、利用者は長時間にわたり、応答作度低下の影響を受けた。

【原因】直接には、負荷分散装置のファームウェアの不具合が原因。発見が遅れたのは、サービス監視の条件設定が最適でなかったため。（サービス監視はリクエストが一定回数連続して廃棄された場合にアラームを発するよう設定されていたが、今回は閾値を超えるまでには至らなかったため、サービス監視からの通知はなかった。）

【対策】負荷分散装置のファームウェアを更新するとともに、サービス監視の条件を変更した。
サービス監視等の基本的な取組みを実践した上で、さらなる早期発見、対処を望む場合には、早期障害検知・分析のための技術・製品（SNS上のつぶやき監視、インバリアント分析、ビッグデータ分析、等）が用いられるようになっている。

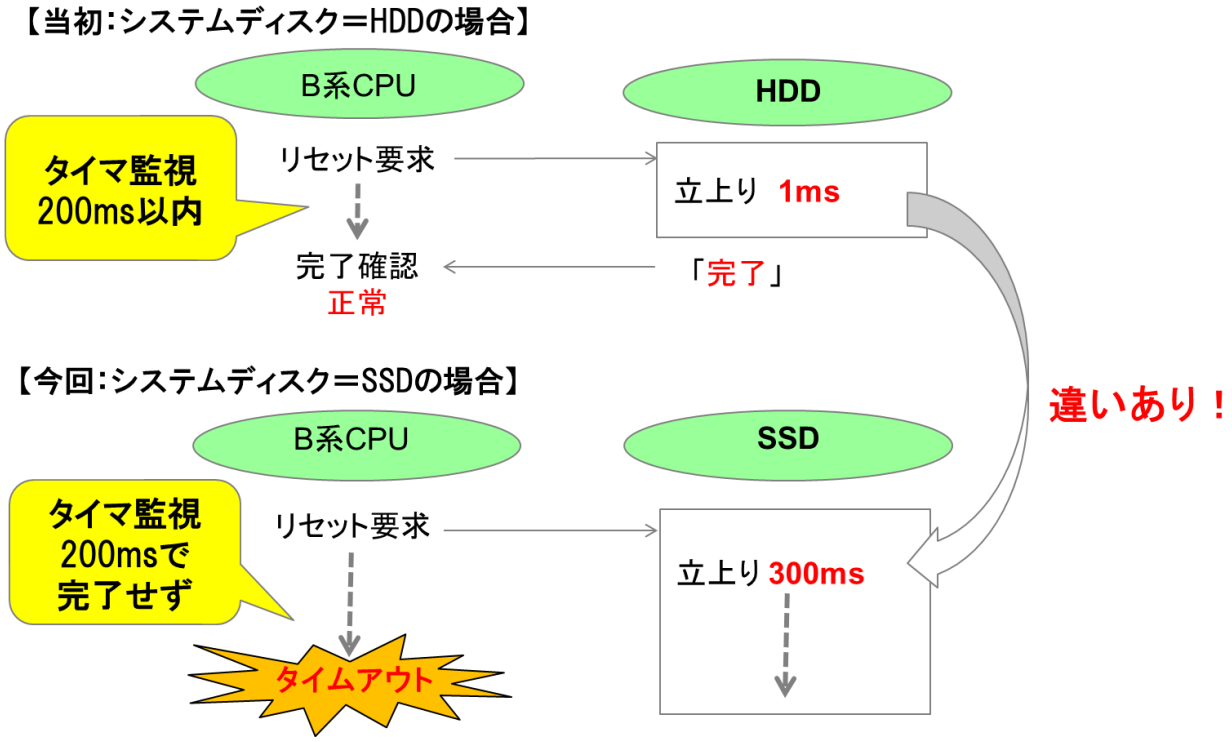


T12: 新製品は、旧製品と同一仕様と言われても、必ず差異を確認！

【問題】2重化された制御系システムにおいて、部品交換の保守作業時にシステム全体の動作が停止し、短時間で復旧できずに、サービス利用者が終日影響を受けた。

【原因】システムのディスク装置が、数年前に，当初構築時のHDDからSSDに交換されていた。部品交換作業でA系を切り離した時にB系OSから両系のディスク装置にリセット要求が発せられるが、SSDのリセット要求処理時間は、HDDのそれよりかなり長く，OSのタイマ監視においてタイムアウトが発生した。その後のリカバリ処理もうまくいかなかった。

SSDへの交換時に、HDDと完全に互換性があると誤認し、検証・テストが不十分であった。



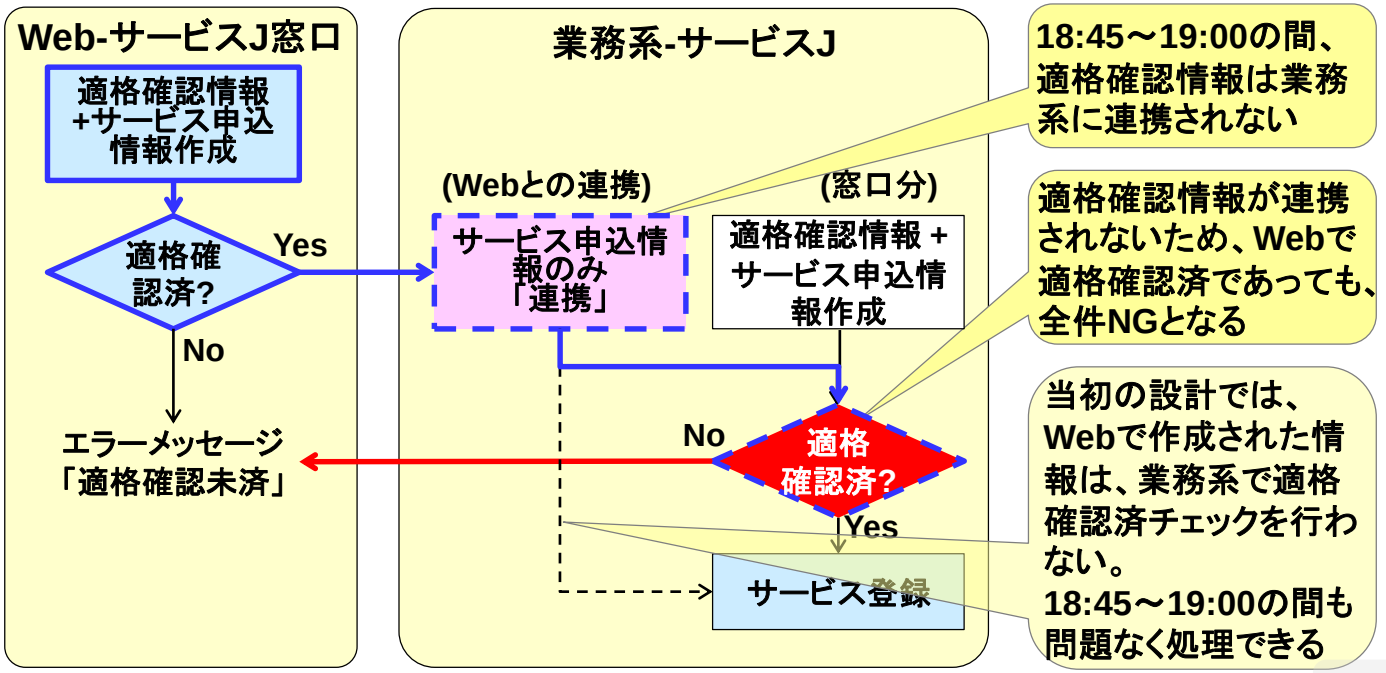
【対策】
- 仕様上の互換性を過信せず、差異分析を必ず実施
- ベンダとユーザの双方が相手の役割分担を支援し合う（ユーザ側でハザード分析を行う）

T13: 利用者の観点に立った、業務シナリオに即したレビュー、テストが重要

【問題】オフラインでの申込みのみサポートしていたところを、追加でWeb経由での申込みを可能としたサービスにおいて、特定の時間帯に限り、Webサイトからのサービス申込みが全て不備とみなされ、登録できなかった。顧客からの連絡で判明した。

【原因】オフライン/Web経由の2系統のサービス申込みを処理するロジックにおいて、各系統の処理間でのデータの連携に誤りがあった。根本原因としては、全体設計が個別システム設計に正しく引継がれなかったことと、業務シナリオに即した確認が行われず、設計後のレビューでも発見されず、対応するテストも行われなかったこと。

【対策】処理ロジックを正しく修正した。また、要件定義・設計段階でウォークスルー等により関係者相互で確認するとともに、利用者の観点に立った、業務シナリオに即した検証を行うようにした。



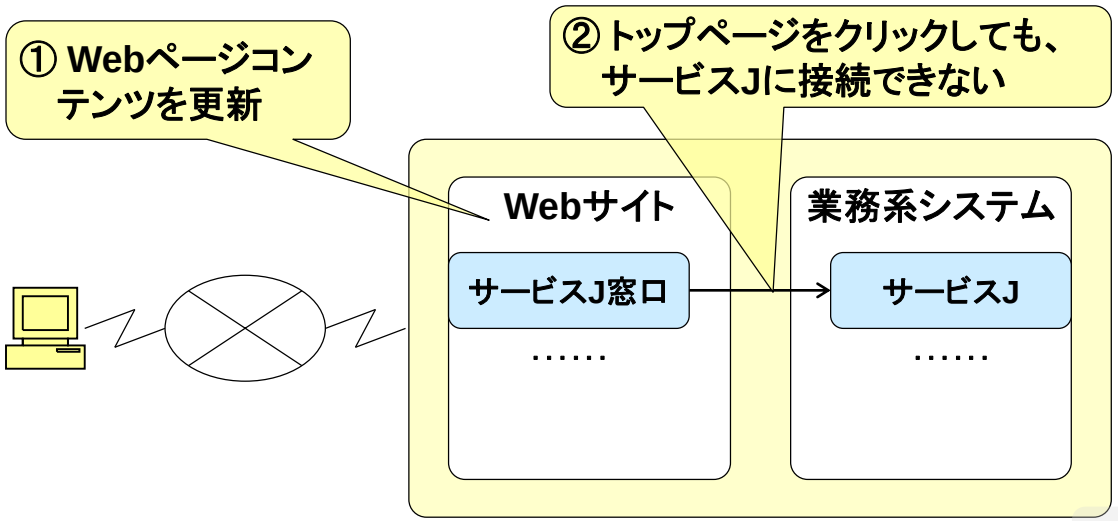
T14: Webページ更新時には、応答速度の変化、性能面のチェックも忘れずに

【問題】Webサイト上のあるサービスのトップページをクリックすると、応答に長時間を要し、目的のサービスに接続できないケースが多発した。

【原因】業務部門がトップページのコンテンツを更新した結果、1顧客当りのダウンロードサイズが4倍になったが、応答速度への影響を確認しないままリリースした。
業務部門はダウンロードサイズと応答性能との関連を意識せず、それに関するIT部門による技術的な確認がルール化されていなかった。

【対策】業務部門がWebページコンテンツを更新する際には、IT部門が技術的な観点で確認を行うことを手順書に明記するとともに、IT部門が必要と判断した場合、業務部門に対しリリース中止を指示できるようルールを改めた。

次の直接的対策も実施：
- 当該トップページへのアクセスを高速ネットワークサービス経由に変更
- コンテンツ変更量の自動チェック機能を導入し、最新のコンテンツ量とアクセス量を可視化

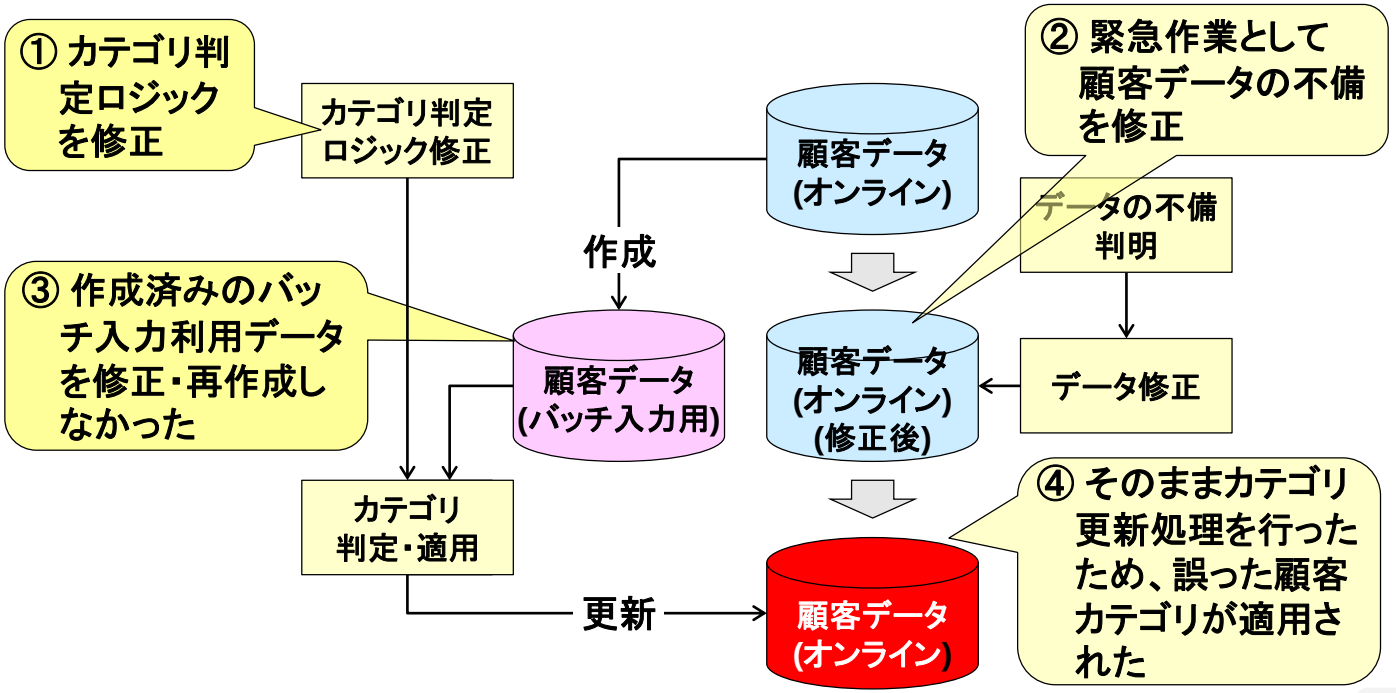


T15：緊急時こそ、データの一貫性を確保するよう注意すべし

【問題】毎月末に、顧客のカテゴリ判定バッチ処理を、あらかじめ作成しておいた顧客データ（マスタ）のコピーを用いて行う運用において、ある時、緊急の要請により、マスタを修正して対応したことがあった。その後のオンライン処理において、誤った顧客カテゴリが適用された。

【原因】緊急対応後に、マスタから顧客カテゴリ判定用コピーの再作成を行わなかったため、マスタとコピーとの不整合が発生していたにも関わらず、そのまま、カテゴリ判定処理を行ったため。

【対策】緊急時対応の影響範囲を見極め、対応結果が平常時のシステム運用の流れに確実に繰り込まれるよう、特に意識するよう周知するとともに、作業ルール・手順書を明確化。

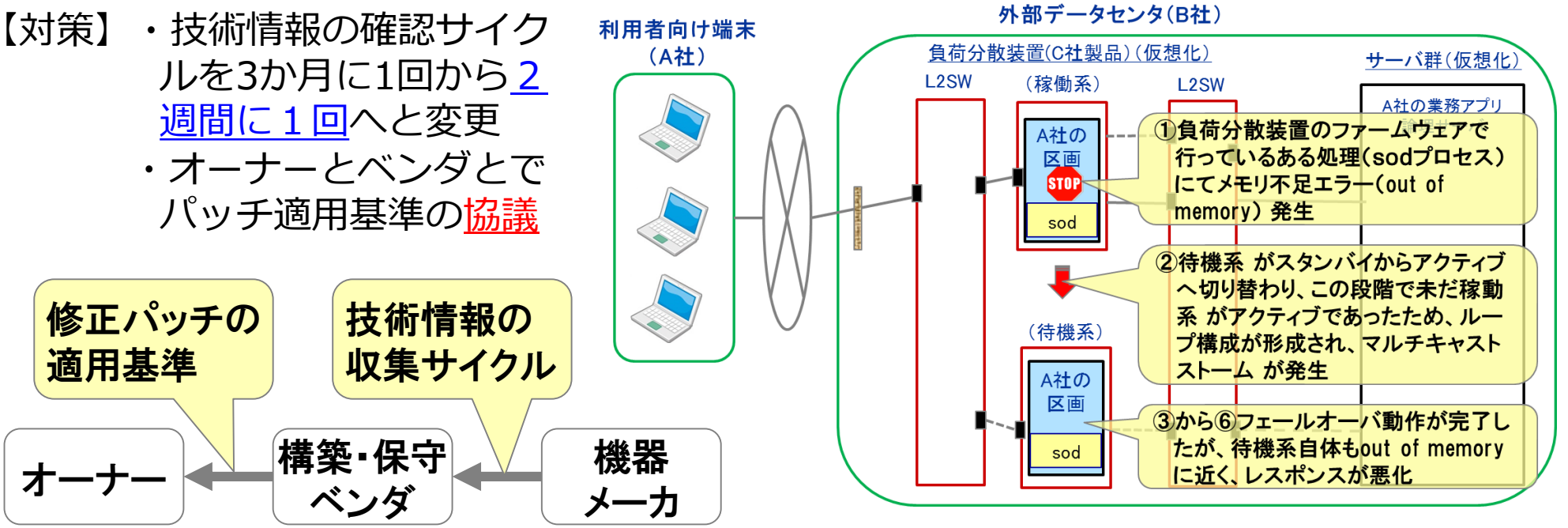


T16: システム構成機器の修正パッチ情報の収集は頻繁に行い、緊急性に応じて計画的に対応すべし

【問題】システムの通信機器（負荷分散装置）に障害が発生し、丸１日間業務が停止

【原因】システム構築・保守ベンダが外部メーカから調達した負荷分散装置のファームウェアの既知の不具合が直接の原因であり、その修正パッチが1ヵ月前に公表されていたが、ベンダによる修正情報の収集間隔が３ヶ月に１回程度と非常に粗く設定していたため、その適用が間に合わなかった。システムのオーナーは、技術情報が時々公表されていることを認識していなかった。

【対策】
・技術情報の確認サイクルを3か月に1回から２週間に１回へと変更
・オーナーとベンダとでパッチ適用基準の協議

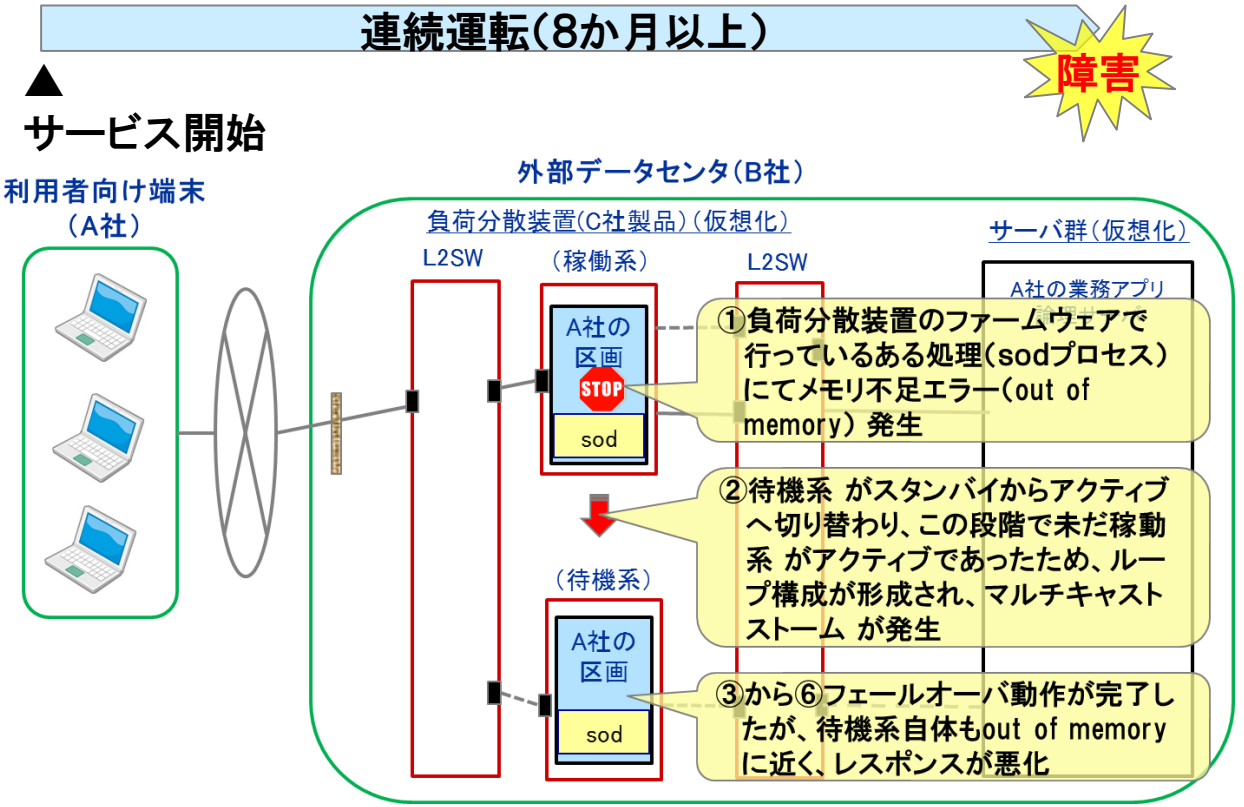


T17: 長時間連続運転による不安定動作発生の回避には定期的な再起動も有効！

【問題】稼働開始以来8か月以上連続運転のシステムの通信機器に障害が発生し、丸１日間業務が停止

【原因】直接の原因は、あるプロセスのメモリ資源を時間の経過とともに消費し続けるという負荷分散装置のファームウェアの不具合（装置を定期的に再起動をしていれば顕在化しなかった）

【対策】システムの再起動のサイクルを検討し、毎月の定期保守日に状況を見て再起動することを決定
（ネットワーク機器について、定期的に再起動することにより、長時間連続運転による不具合の顕在化が回避できることが経験的に知られている）

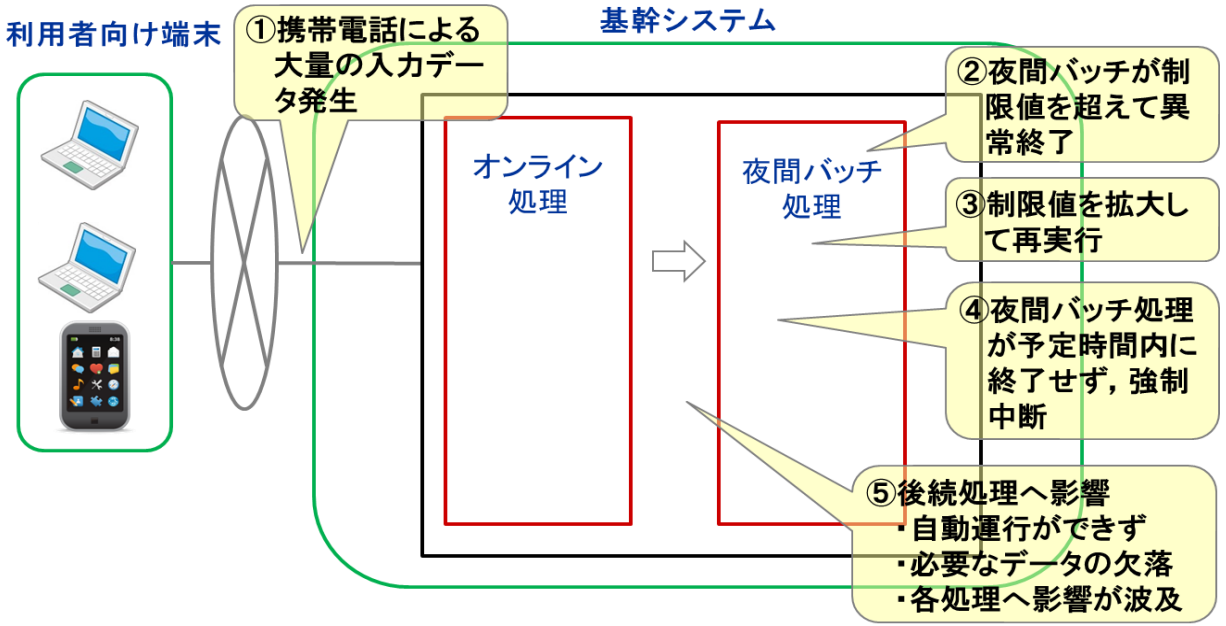


T18: 新たなサブシステムと老朽化した既存システムとを連携する場合は両者の仕様整合性を十分確認すべし

【問題】特別な事象を契機に、オンラインで大量の入力が集中して連携するバッチ処理がオーバーフローし、その後の対応も誤って10日間程度オンラインサービス停止等

【原因】携帯電話に対応した新しいシステムと既存システムとを接続した際の全体整合を十分チェックできておらず、携帯電話からのバースト的なサービス要求が無制限に受け付けられて後続のバッチシステムに連携され、夜間バッチの処理能力の制限値を超えたため異常終了。その結果、膨大な作業が発生し、処理失念や誤処理による多数の副次的障害も発生。

【対策】
- 既存システムの要件定義内容を再度チェックして連携するシステム間の整合性を確認。これらをルール化してマニュアルに反映。
- システムが異常終了しても途中から再開始可能な仕組みの導入。

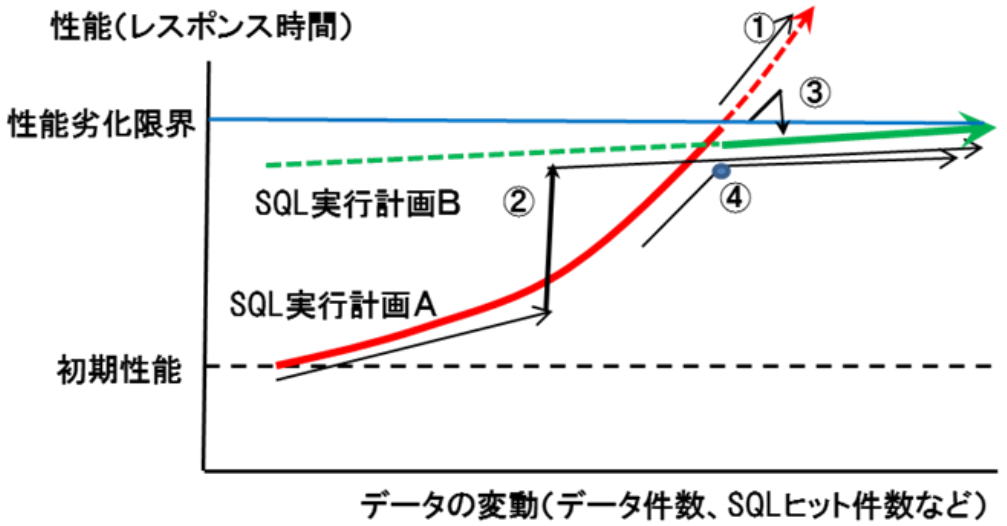


T19: リレーショナルデータベース（ＲＤＢＭＳ）のクエリ自動最適化機能の適用は慎重に！

【問題】ある朝、オンラインシステムのタイムアウトが多発し運用監視コンソール端末にトランザクション異常終了のメッセージが大量に出力された。

【原因】広範囲にわたる検索条件入力が投入された(①)ため、ＤＢサーバで稼働するRDBMSの最適化機能により全件フルスキャンを行うSQL実行計画が適用されSQL処理が長時間実行された(②)。APサーバのSQL監視タイマのタイムアウト（3分）となりアプリケーションが異常終了した(③)。以降のトランザクションも同じSQL実行計画が適用され続けたため。

【対策】・RDBMSの再起動で回復
・フルスキャンが発生しにくいようインデックスキーを複数追加した



①SQL実行計画Aが切り替わらない場合(徐々に性能劣化限界へ)
②性能劣化限界以前にSQL実行計画AからBに切り替わったケース(突然性能劣化発生)
③性能劣化限界を過ぎた後でSQL実行計画AからBに切り替わったケース(徐々に性能劣化限界へ)
④性能劣化限界でSQL実行計画AがBに切り替わったケース(理想)

T19: リレーショナルデータベース（ＲＤＢＭＳ）のクエリ自動最適化機能の適用は慎重に！

【問題】ある朝、オンラインシステムのタイムアウトが多発し運用監視コンソール端末にト

SQL実行計画の変更による性能劣化発生への対策

案	対策内容	メリット	デメリット	備考
1	インデックスを追加	クエリ最適化機能は活用できる	更新負荷の増大による性能劣化の恐れあり	データ量の変動が予測される場合
2	SQL実行計画の固定化（利用比率の高いSQLのみ）	案3に比べて効率的	利用比率が均等だと効果は小さい	システム全体に影響させたくない場合
3	全てのSQL実行計画を固定化	SQL実行計画の自動切替は発生しなくなり極端な劣化は発生しない。	データ量の変化に追随しないため高速処理は望めない。	極端に遅くなるのを防止したい場合

全ひにくいようインデックスキーを複数追加した

データの変動（データ件数、SQLヒット件数など）

①SQL実行計画Aが切り替わらない場合（徐々に性能劣化限界へ）
②性能劣化限界以前にSQL実行計画AからBに切り替わったケース（突然性能劣化発生）
③性能劣化限界を過ぎた後でSQL実行計画AからBに切り替わったケース（徐々に性能劣化限界へ）
④性能劣化限界でSQL実行計画AがBに切り替わったケース（理想）

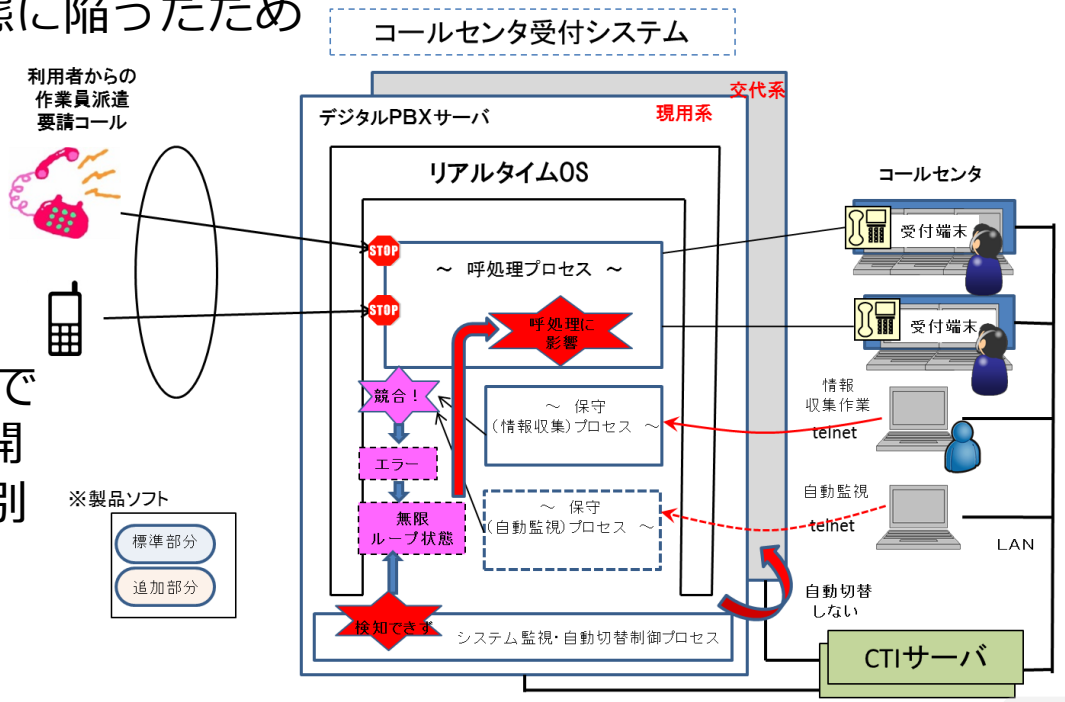
T20: パッケージ製品の機能カスタマイズはリスクを認識し特に必要十分なチェック体制やチェック手順を整備して進めること

【問題】コールセンタのオペレータ受付端末のタッチパネルが反応しなくなった。本事象は同じ構成を持つ交代系システムへの切替を実施するまで約10分間継続していた。

【原因】　新システムに更改して1週間連続稼働していた。新システムはベンダのパッケージ製品ソフトを導入し構築されたが、Ａ社の要件により機能追加＊1のカスタマイズ対応がされていた。システムダンプを取得し解析した結果、障害情報収集のためのtelnet接続と呼処理を自動監視するためのtelnet接続が競合し無限ループ状態に陥ったためと判明した。また、異常を検知して交代系システムに自動的に切替える機能もソフトウェアの不具合から無限ループを検知できなかった

＊1 呼処理の自動監視機能

【対策】
- ファームウェアの不具合、手動で交代系に切替えてサービスを再開
- 障害情報収集をtelnet方式から別の方式に変更
- ファームウェアの修正
- FTA分析による総点検実施



T20: パッケージ製品の機能カスタマイズはリスクを認識し 特に必要十分なチェック体制やチェック手順を整備して進めること

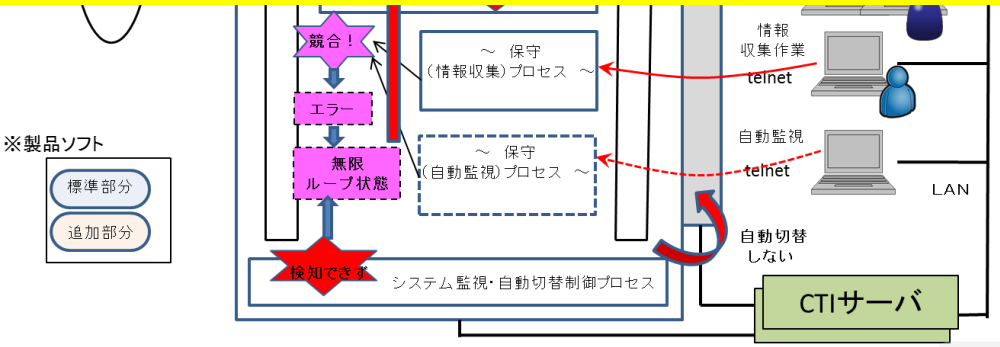
【問題】コールセンタのオペレータ受付端末のタッチパネルが反応しなくなった。本事象は同じ構成を持つ交代系システムへの切替を実施するまで約10分間継続していた。

【原因】　新システムに更改して1週間連続稼働していた。新システムはベンダのパッケー

パッケージ製品のカスタマイズをする場合
カスタマイズした製品は稼動実績はないと認識

改造部分のテストに加えて、
レグレッションテストも必ず行い
リスクを低減する

【対策】
- ファームウェアの不具合、手動で交代系に切替えてサービスを再開
- 障害情報収集をtelnet方式から別の方式に変更
- ファームウェアの修正
- FTA分析による総点検実施



パッケージ製品利用時の留意点

【参考】パッケージ製品導入によるトラブルは事例が多い

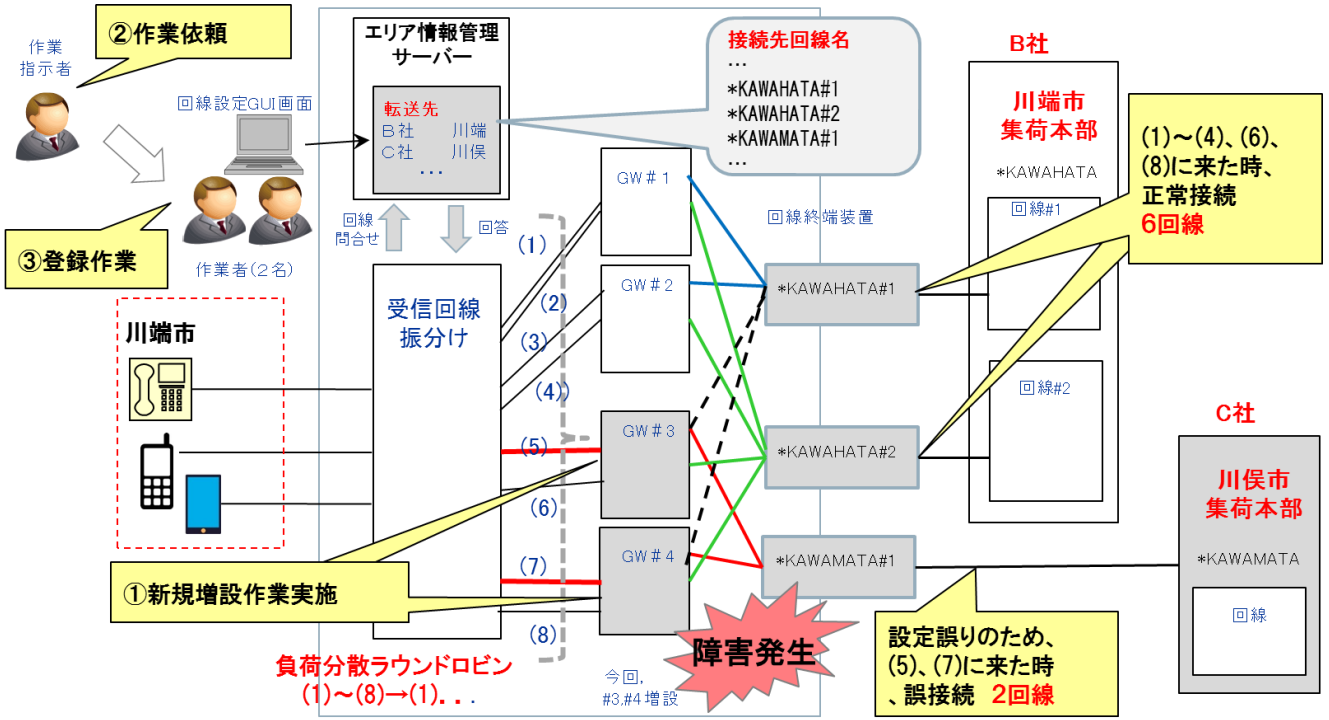
- パッケージに業務をあわせることが前提
- パッケージのセールストークを信じて内容確認を怠ると後で痛い目に
- FIT＆GAPには有識者の確保が必要
- カスタマイズ量の管理はタイムリーに実施
- パッケージ機能の確認だけでなく性能評価も忘れずに
- 稼動実績があってもバージョンが異なる場合は初物と思うこと
- サポート体制の確立が重要

T21：作業ミスを減らすためには、作業指示者と作業者の連携で漏れのない対策を！

【問題】A社は、4回に1回の割合でB社への集荷依頼をC社集荷本部に転送していた。現場は混乱し、集荷作業漏れが多発し、顧客からの苦情が殺到していた。

【原因】障害の直接原因は、作業者の些細な誤りであったが、根本的には、誤りを見逃しやすい作業環境と最後の砦となるべき作業指示者の確認不足によるものであった。

【対策】
- **作業者の観点から、個人、環境、ハードウェア、ソフトウェアの視点で、作業ミスの原因、対策を考える。**
- **作業指示者の観点から、作業指示者は、システムの問題を仕組みや組織として改善することに主眼を置く。**



T21：作業ミスを減らすためには、作業指示者と作業者の連携で漏れのない対策を！

【問題】A社は、4回に1回の割合でB社への集荷依頼をC社集荷本部に転送していた。現場は混乱し、集荷作業漏れが多発し、顧客からの苦情が殺到していた。

【原因】障害の直接原因は、作業者の些細な誤りであったが、根本的には、誤りを見逃しやすい作業環境と最後の砦となるべき作業指示者の確認不足によるものであった。

【

川端本部とすべきところ川俣本部と設定誤り
○ KAWAHATA　× KAWAMATA
再監しても日本語読みで誤読したため気付かず
→アルファベット1文字ずつ確認に変更

※海外パッケージ（日本語化未対応）

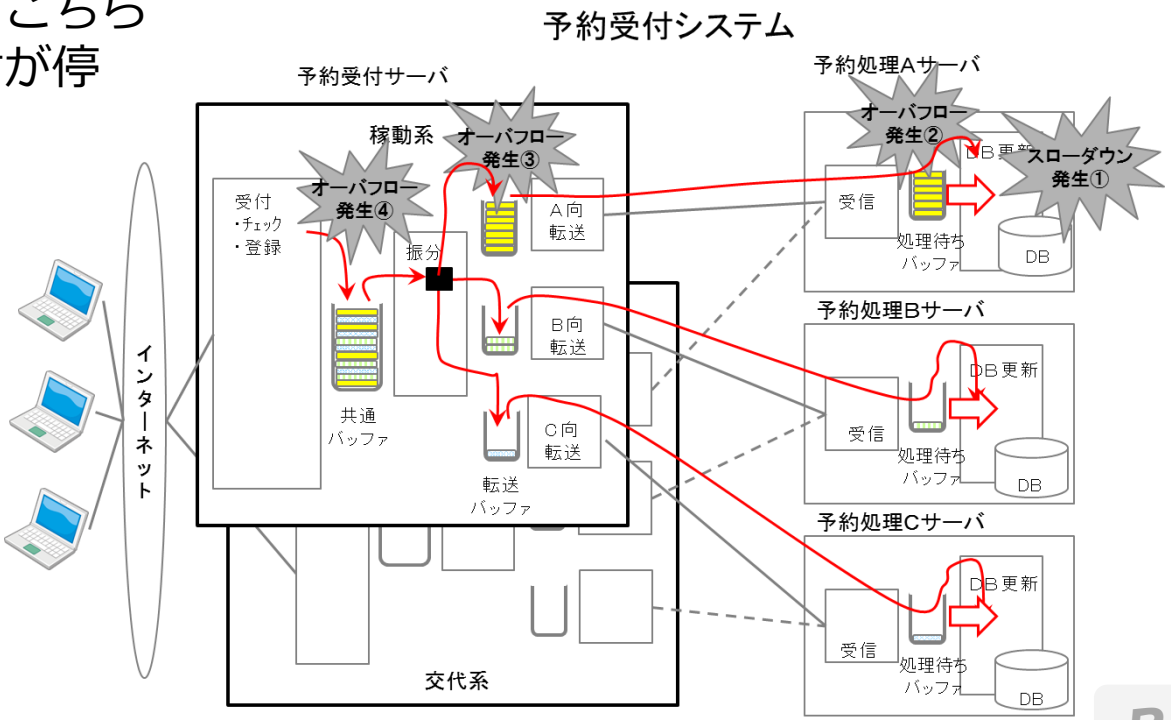
組の外組織として改善することに主眼を置く。

T22: 隠れたバッファの存在を把握し、目的別の閾値設定と超過アラート監視でオーバフローを未然に防止すること

【問題】予約受付システムが特定の予約業務の停止から始まり、全予約業務の停止に拡大し、再開まで２時間程度サービス不可となった。

【原因】　予約処理Ａサーバの処理がメモリ不足により、スローダウン状態となったため処理待ちバッファに徐々に入力データが滞留し、上限の1,000件を越えた。これにより、予約受付サーバ側の転送バッファに未送信データが滞留して満杯となり、共通バッファから予約処理Ａの入力データが取り出されなくなり、こちらも満杯となり全予約処理の受付が停止した。

【対策】
- 予約業務Aのスローダウン状態を回復
- 滞留データを順次処理し共通バッファを空にする
- 予約受付システムを再開
- **バッファの洗い出し、閾値設定とアラート監視の改善**



T22: 隠れたバッファの存在を把握し、目的別の閾値設定と超過アラート監視でオーバフローを未然に防止すること

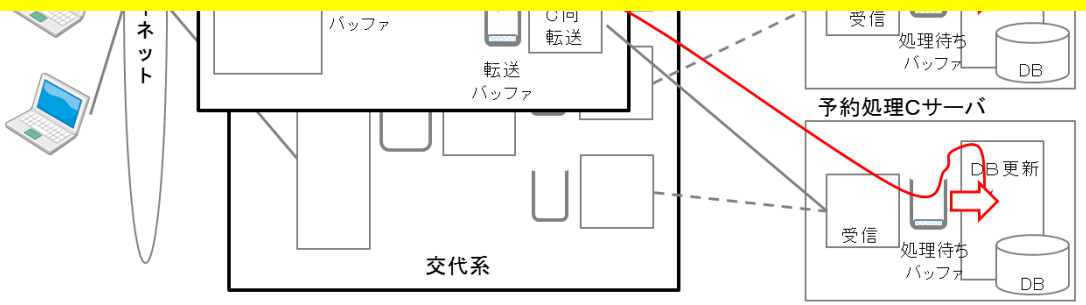
【問題】予約受付システムが特定の予約業務の停止から始まり、全予約業務の停止に拡大し、再開まで２時間程度サービス不可となった。

【原因】　予約処理Ａサーバの処理がメモリ不足により、スローダウン状態となったため処理待ちバッファに徐々に入力データが滞留し、上限の1,000件を越えた。これ

各種バッファの存在を認識し、構成情報としてその最大収納件数を認識･管理する。

とくにパッケージ製品を利用する場合は明示されていない内部バッファの存在を把握しておく必要がある。

共通バッファを空にする
・予約受付システムを再開
・**バッファの洗い出し、閾値設定とアラート監視の改善**



T23: 障害監視は、複数の観点から実装し、障害の見逃しを防げ！

〔問題〕Ａ社の基幹システムは24時間365日稼働のオンラインシステムであり、業務の特性から瞬時の停止も許されない。DBサーバは4重化されており、それぞれ障害監視機能を持っている。ある日、DBサーバ4台全てが順番に停止した。

〔原因〕サーバ停止の直接の原因は、ネットワークスイッチのキャッシュメモリ故障であった。このスイッチは、本件に関してエラーメッセージを出力しなかったので障害検知ができなかった。タイミング悪く２つの監視機能（データ同期更新、自ノード監視）の実行タイミングの重複によって全DBサーバが停止してしまった。

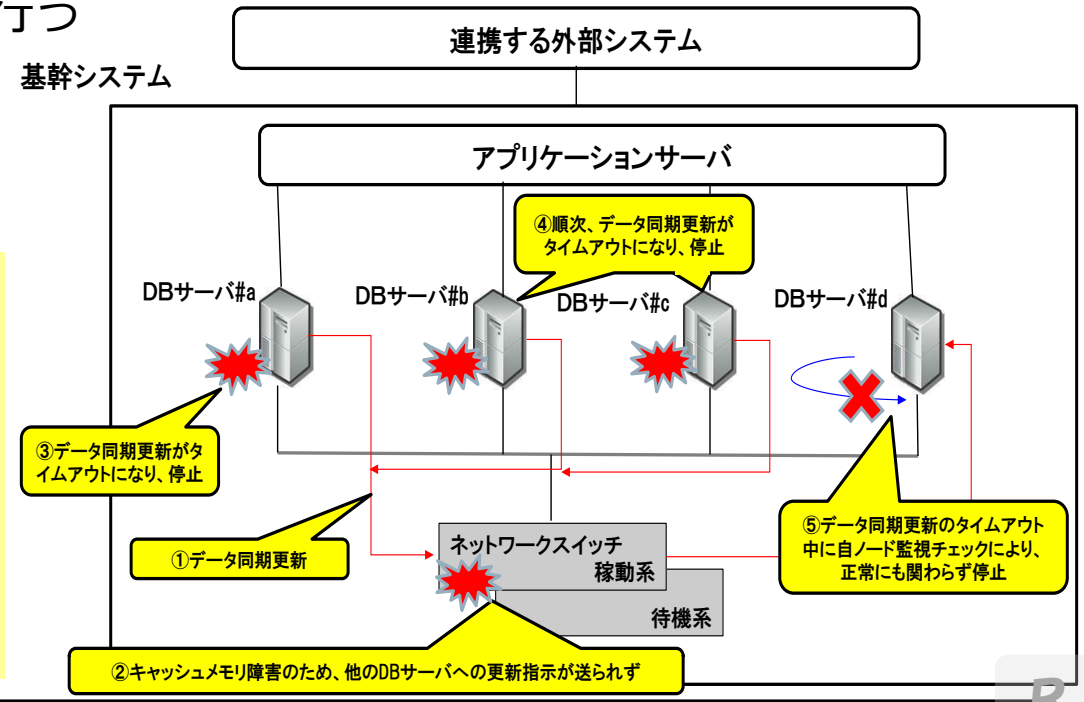
〔対策〕

【１】個々の機器の監視を複数のツールで行う

【２】複数の監視機能の組合せで動作に問題がないか確認する

【３】十分な保守時間の確保

〔教訓〕監視技術が重要度を増している一方で、監視の不具合による障害も増えている。この事例は、監視ミス（自ノード監視がエラーでないものをエラーとした）と監視もれ（ネットワークスイッチのエラーを見過ごした）が同時発生した事例である。



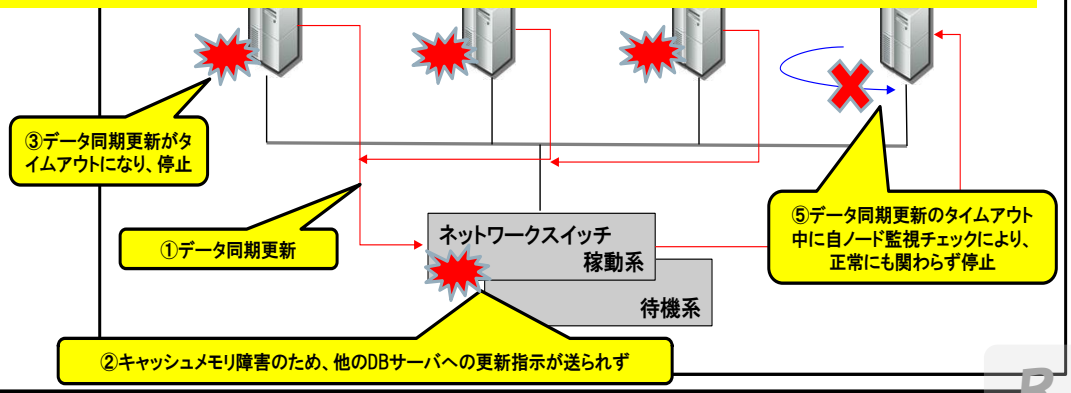
T23: 障害監視は、複数の観点から実装し、障害の見逃しを防げ！

〔問題〕Ａ社の基幹システムは24時間365日稼働のオンラインシステムであり、業務の特性から瞬時の停止も許されない。DBサーバは4重化されており、それぞれ障害監視機能を持っている。ある日、DBサーバ4台全てが順番に停止した。

〔原因〕サーバ停止の直接の原因は、ネットワークスイッチのキャッシュメモリ故障であった。このスイッチは、本件に関してエラーメッセージを出力しなかったので障害検知ができなかった。タイミング悪く２つの監視機能（データ同期更新、自ノード監視）の実行タイミングの重複によって全DBサーバが停止してしまった。

機器のエラーメッセージを監視するだけではメッセージが出ない場合は検知できない。能動的に正常動作を監視することも組み合わせる必要がある。

一方で、監視の不具合による障害も増えている。この事例は、監視ミス（自ノード監視がエラーでないものをエラーとした）と監視もれ（ネットワークスイッチのエラーを見過ごした）が同時発生した事例である。



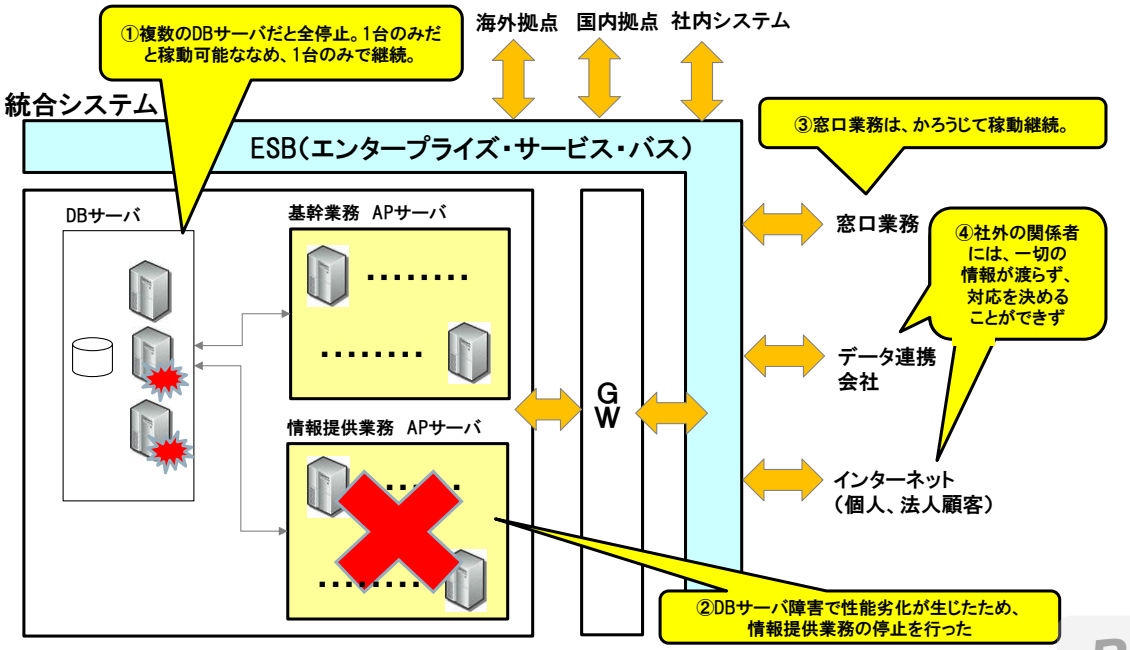
T24: サービス縮退時の対策を考慮せよ

〔問題〕 A社の統合システムの3台でクラスタリング構成を組んだDBサーバが順次停止し、全DBサーバが停止。1台だけ稼動させ、情報提供業務を停止させて基幹業務だけに専念させてサービスを継続。情報提供業務の長時間のサービス停止により混乱は終日続いた。

〔原因〕統合システムが障害になりサービス縮退となった場合、対外システムやエンドユーザに「障害情報と復旧見込み」などの情報を発信することができなかったため、障害の影響を大きくしてしまった。根本原因は、サービス縮退が長期化することによって、問題が大きくなることを見逃していたことである。

〔対策〕 【1】エンドユーザの要求の変化を検証する
【2】ディペンダビリティ を追求したシステム構成を考慮する

〔教訓〕設計時にリスク分析を行いディペンダビリティの確保に対応する最適なサービス縮退を考慮することにより、最適対策を取ることができる。
サービス縮退のあり方から、「システムの集中」が良いか、「システムの分散」が良いかなど、システム障害のリスクを軽減する構成を決めることも検討できる。



T24: サービス縮退時の対策を考慮せよ

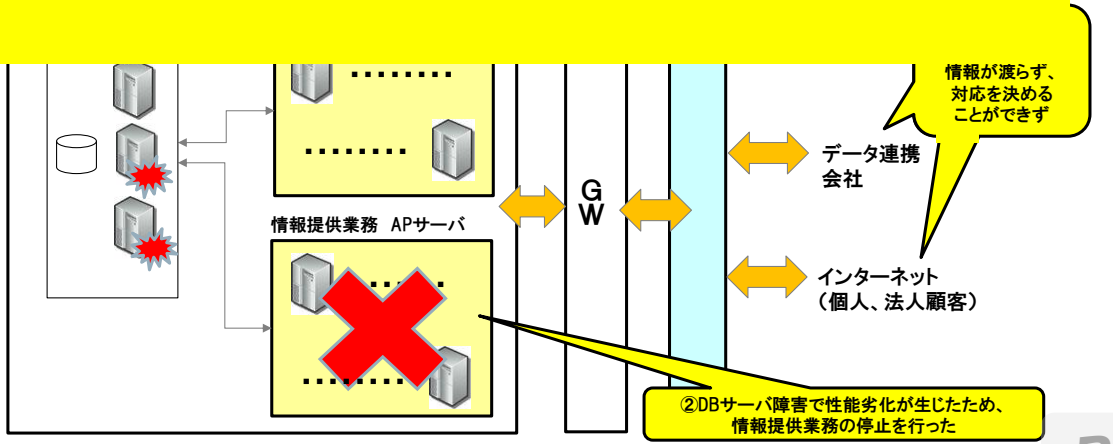
〔問題〕 A社の統合システムの3台でクラスタリング構成を組んだDBサーバが順次停止し、全DBサーバが停止。1台だけ稼動させ、情報提供業務を停止させて基幹業務だけに専念させてサービスを継続。情報提供業務の長時間のサービス停止により混乱は終日続いた。

〔原因〕統合システムが障害になりサービス縮退となった場合、対外システムやエンドユーザに「障害情報と復旧見込み」などの情報を発信することができなかったため、障害の影響を大きくしてしまった。根本原因は、サービス縮退が長期化することによって、問題が大き

一部のサーバ障害が発生し、縮退運転を余儀なくされる場合を想定し、縮退運転時の最適なサービス構成をコンティンジェンシプランとして準備しておく。

る。

サービス縮退のあり方から、「システムの集中」が良いか、「システムの分散」が良いかなど、システム障害のリスクを軽減する構成を決めることも検討できる。



T25：障害原因が不明でも再発予防と発生時対策はできる

〔問題〕仮想化基盤上の各サーバからストレージへの通信が不安定になり、サーバ上に構築したすべてのサービスを内部/外部の利用者に終日提供できなくなった。

〔原因〕仮想化サーバとストレージを接続するスイッチに異常が発生した。当該スイッチ他各層のスイッチ、ストレージの再起動による復旧を順に試みたが効果がなく、当日の運用再開は断念した。その後、スイッチ目視によりLED点灯異常を発見し、今度は電源抜線・再投入を実施した結果、通信は回復した。ハードかファームかなど原因や発生条件は不明のままであり、今後も**再発リスクあり**。

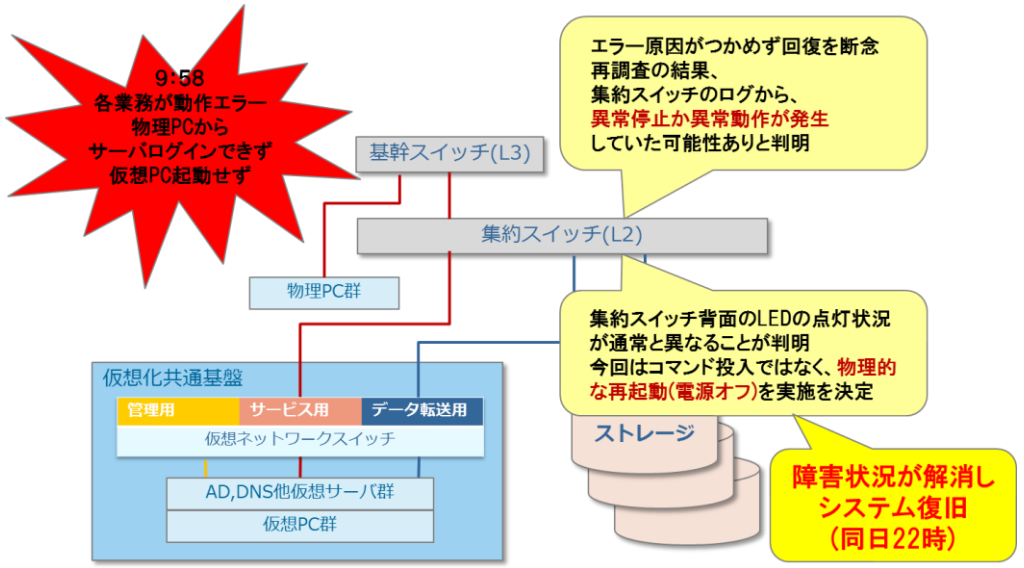
〔対策〕予防保守のため障害発生したスイッチを交換するとともに、本件の事象が再発した際の**リスク対策として**スイッチ点灯状況の目視と異常時の電源抜線・再投入を**障害対応手順に加えた**。

〔教訓〕システムに障害発生リスクがあると判明したら、そのシステムの稼働要件に応じたリスク対策を構築してトラブル対応手順の最新化と周知を怠らないこと。

障害発生のリスクがあっても、運用要件に応じた対策を手順化し周知しておけば、再発時に慌てず安全確実にシステムを回復できる。

障害対応手順(危機管理マニュアル)
- **障害対応方針(縮退運転継続、全再起動、障害調査優先等)**
- **発生時の連絡体制、対応体制**
- **発生時の具体的手順(障害判定、ログ取得、エラー回復)**

得られた経験は必ず次回の対策に役立てる



T25: 障害原因が不明でも再発予防と発生時対策はできる

〔問題〕仮想化基盤上の各サーバからストレージへの通信が不安定になり、サーバ上に構築したすべてのサービスを内部/外部の利用者に終日提供できなくなった。

〔原因〕仮想化サーバとストレージを接続するスイッチに異常が発生した。当該スイッチ他各層のスイッチ、ストレージの再起動による復旧を順に試みたが効果がなく、当日の運用再開は断念した。

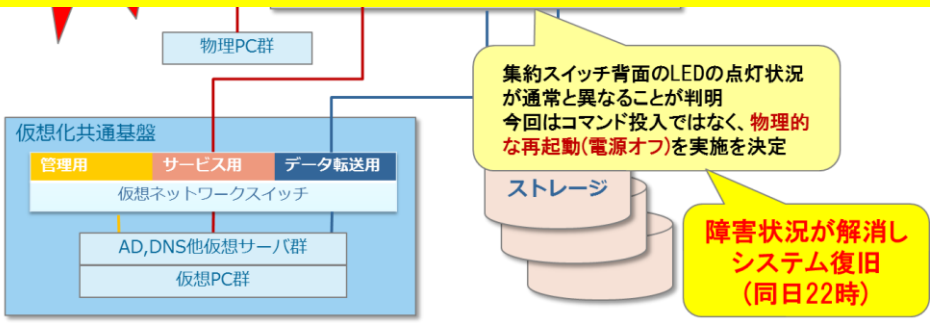
障害対応の経験は貴重な財産

新たな障害発生のリスクが発生しても、運用要件に応じた対策を障害対応手順に追記し周知しておけば、再発時に慌てず安全確実にシステムを回復できる。

に慌てず安全確実にシステムを回復できる。

障害対応手順(危機管理マニュアル)

- **障害対応方針(縮退運転継続、全再起動、障害調査優先等)**
- **発生時の連絡体制、対応体制**
- **発生時の具体的手順(障害判定、ログ取得、エラー回復)**



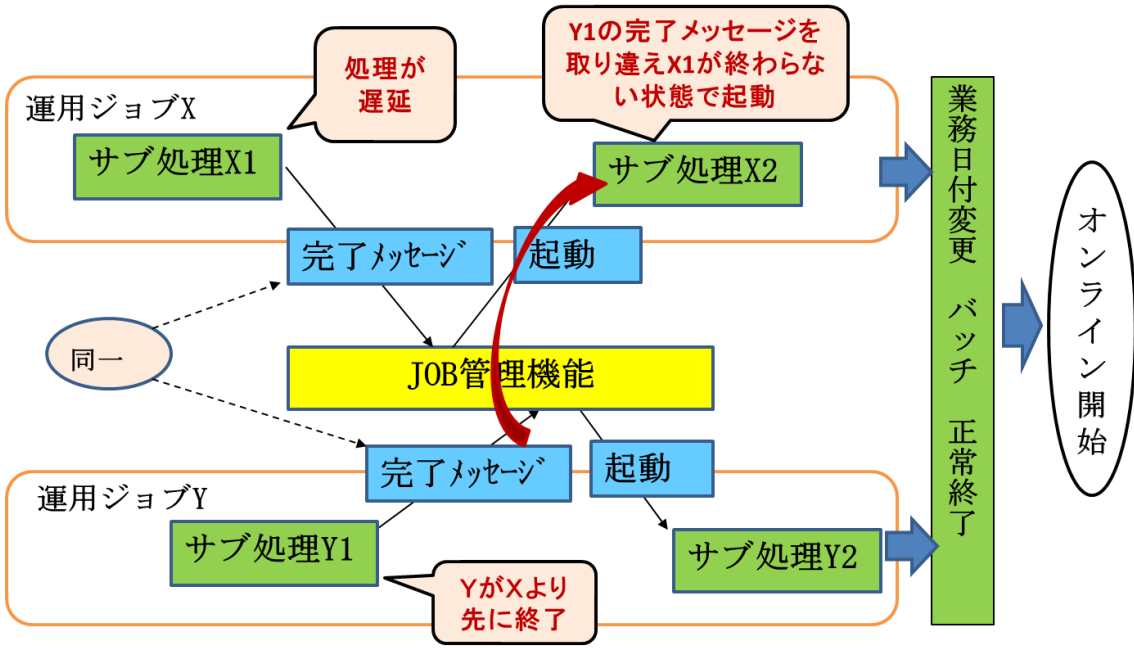
T26: 既存システムの流用開発はその前提条件を十分把握し、そのまま利用可能な部分と変更する部分を調査して実施する

〔問題〕翌日オンラインサービスに向けた夜間の業務日付変更バッチ処理が異常終了し、オンライン開始が大幅に遅延した。

〔原因〕バッチ処理のJOBネットは先行JOBの完了メッセージを受けて後続のJOBが起動される仕組みであるが、別のJOBネットを追加する際に既存を流用して開発した。完了メッセージが同一となっていたため、その取違いが発生し先行JOBが終了する前に後続のJOBが起動されて異常終了した。

〔対策〕稀なタイミングで発生したものであり、原因が判明するまでに時間を要した。後続JOBの再ランを実行しようやくオンライン開始となった。
　2つのJOBネットを並行処理から逐次処理に変更した。

〔教訓〕
**　追加機能の開発は既存の似た処理があれば、安易にコピー、流用してモデファイし対応しがちであるが落とし穴があることが多い。**
**　既存システムの背景・前提条件を十分把握し変更すべき個所を洗い出し、レビューすることが大切。**



T26: 既存システムの流用開発はその前提条件を十分把握し、そのまま利用可能な部分と変更する部分を調査して実施する

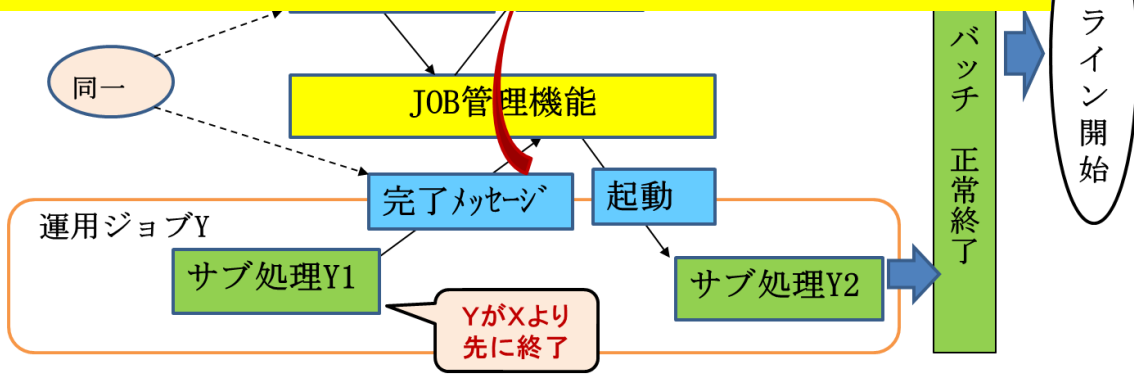
〔問題〕翌日オンラインサービスに向けた夜間の業務日付変更バッチ処理が異常終了し、オンライン開始が大幅に遅延した。

〔原因〕バッチ処理のJOBネットは先行JOBの完了メッセージを受けて後続のJOBが起動される仕組みであるが、別のJOBネットを追加する際に既存を流用して開発した。完了メッセージが同一となっていたため、その取違いが発生し先行JOBが終了する前に後続のJOBが起動されて異常終了した。

〔対
ラン
2

既存のシステム・プログラムを流用して開発する場合、メッセージやファイル名が重複するなど思わぬ落とし穴がある。既存を熟知している人にレビューするなどして修正漏れなどを排除！

とし穴があることが多い。
**　既存システムの背景・前提条件を十分把握し変更すべき個所を洗い出し、レビューすることが大切。**



T27：パッケージはサポートを買え

NEW!

〔問題〕海外パッケージで構築した24時間運用中のオンラインサービスが深夜に突然サーバダウンした。開発元と連絡したが復旧まで8時間かかり、その間利用者へのサービス提供ができなかった。

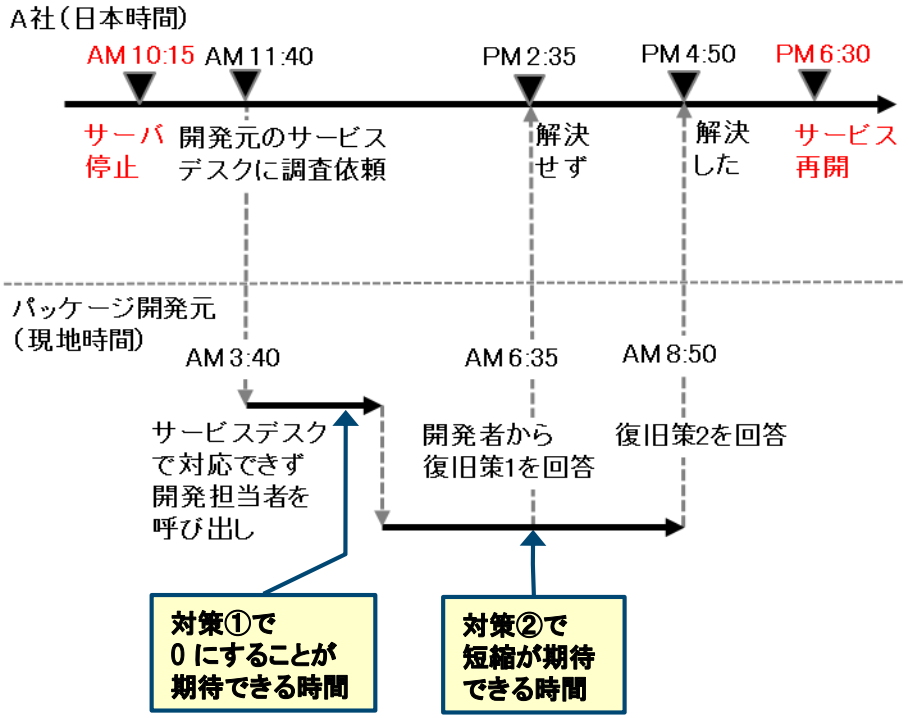
〔原因〕直接の原因はパッケージの潜在的な不具合であったが、復旧までに長時間を要した原因は、開発者でないと対応できない内容のトラブル、発生が現地の深夜であり開発担当者と連絡をとって対応を開始するまでに時間を要した、および対応初期の回答誤りであった。

〔対策〕海外から導入したパッケージについても、国内パッケージと同様にトラブルが発生した際に開発担当者に24時間連絡できる体制の確保を骨子とした、開発元との保守契約の見直し（連絡体制の常備、連絡がつかない時や的確な対応が早期回答できないときのペナルティ等のSLA化）を行った。

〔教訓〕運用中のシステム停止が多大な影響をもたらす重要業務にパッケージを導入する場合には、トラブル発生に即応できる体制をパッケージ開発元も巻き込んで構築すること。

パッケージを導入して構築したシステムの課題
- **トラブル即応力が自主開発に比べて劣る**
- **トラブル難度が高い場合には、サポートデスクでは対応できない場合が多い**

パッケージを基幹業務に組み込む際の留意点
1. **開発者による２４時間直接サポート（緊急連絡ルート）の確保**
2. **回答の即時性担保（回答までの時間、正確性をサービスレベル化し、契約で合意）**



T28：パッケージを更新する時は、変更内容の詳細確認と回帰テストで二重に安全を確保せよ

NEW!

〔問題〕パッケージを導入して安定稼働中のシステムが、バージョンアップ2週間後にサーバダウンした。関連システムからの入力データ連携を遮断後にサーバを再起動させて復旧させたが、データ連携再開後の再処理などのため、サービス再開までに3時間を要した。

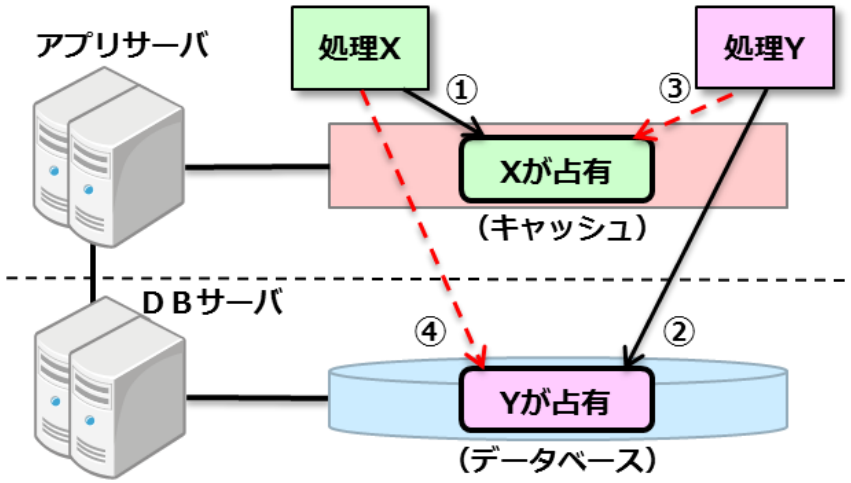
〔原因〕サーバダウンの直接の原因はパッケージ修正モジュールの障害（アプリサーバ上のキャッシュとDBサーバ上のDBの占有順番の不統一によるデッドロック）であったが、リリースノートにはこの修正のことは記載されておらず、適用可否の検討や事前の動作テストは実施されていなかった。

〔対策〕今後、新バージョン提供時には、すべての変更を報告させレビュー会議で実施内容を確認することと、新バージョンがこれまでと同じ動作をすることを回帰テスト（シナリオ再構築と自動化）により確認することをルール化した。

〔教訓〕パッケージを導入して業務システムを構築する場合は、パッケージ開発元から提供された修正モジュールを適用することにより新たな障害が混入するリスクへの対策が必要

① 修正箇所と内容を詳細に確認し、変更部分を漏れなく確認できるよう動作テストを計画

② 変更箇所以外の部分がアップデート前と同じ動作をすることを回帰テストにより確認



・パッケージ開発元がアプリサーバのキャッシュを使用した更新制御を追加した際、処理Xはキャッシュ、DBの順で資源を確保し、処理Yはその逆の順で資源を確保するという、不統一な作りになっていた。

T29：単位などの定義が異なる制限値、連携するシステム間で使っていませんか？

NEW!

〔問題〕個別配送監視システムが停止、拠点ターミナルでは配送状況が分からなくなった。

〔原因〕配送ルート数として持つ制限値としては、当日分の配送ルート数と、翌日分の配送ルート数が存在していた。その制限値が２つあるにも関わらず、個別配送監視システムの開発ベンダは、当日「1,000 ルート」、翌日「1,000 ルート」とあるべきところ、当日、翌日併せて「1,000 ルート」と理解して、システムの設計、製造を行ってしまった

〔対策〕制限値の管理を厳密に定義する、つまり、システムで持つ制限値は、数値だけでなく、「単位」、「単位当たり」、サイン（符号）、上限値、下限値など、正確な定義をした管理を行う。連携システム間の不整合によって起きるシステム障害の発生を防ぐためにそのような制限値を一元管理する。

〔教訓〕連携システム間で持つ制限値は、その定義を明確にすることにより、複数のシステムの制限値が同じ意味をもつことが確認でき、さらに、そのような制限値は、一元管理することが重要である。

制限値に関する他の関連する教訓
G13,T4,T5,T18　など