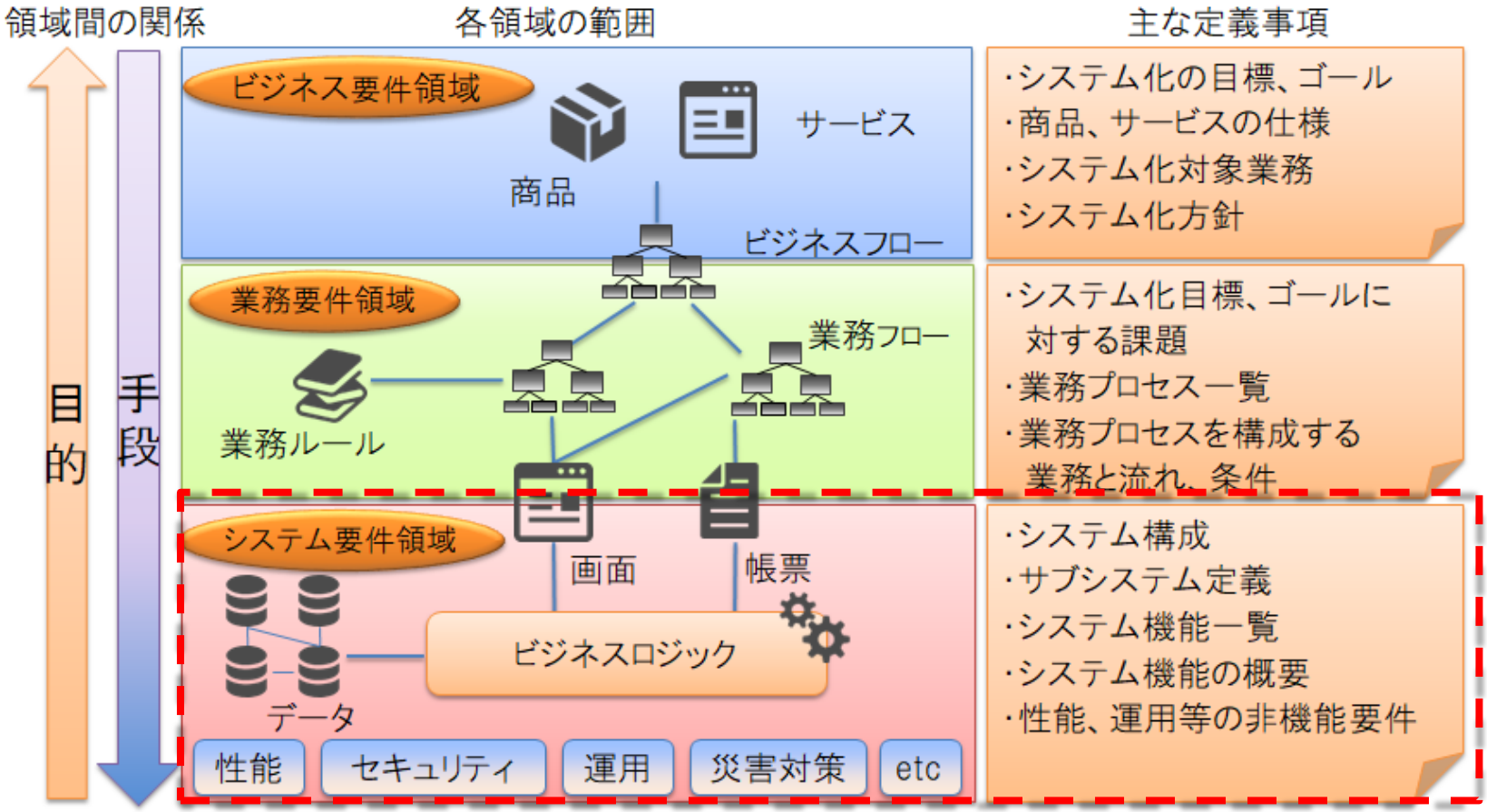
システム要件定義

Ver1.4.0

システム要件定義	1. システム要件定義プロセスの概要
	2. システム要求の収集と整理
	3. 機能要件の定義
	4. 非機能要件の定義
	5. 全体要件の精査、合意と承認
	6. 引継ぎ

『システム要件定義プロセス』とは？

- システム要件定義プロセスは、業務要件実現に必要なシステム機能要件とそれに付随する非機能要件業務を定義するプロセス。
- あるべき業務という”目的” を実現する”手段”としての『十分性』、ITシステムとしての『実現性』の観点から、必要なレベルまで具体化する。

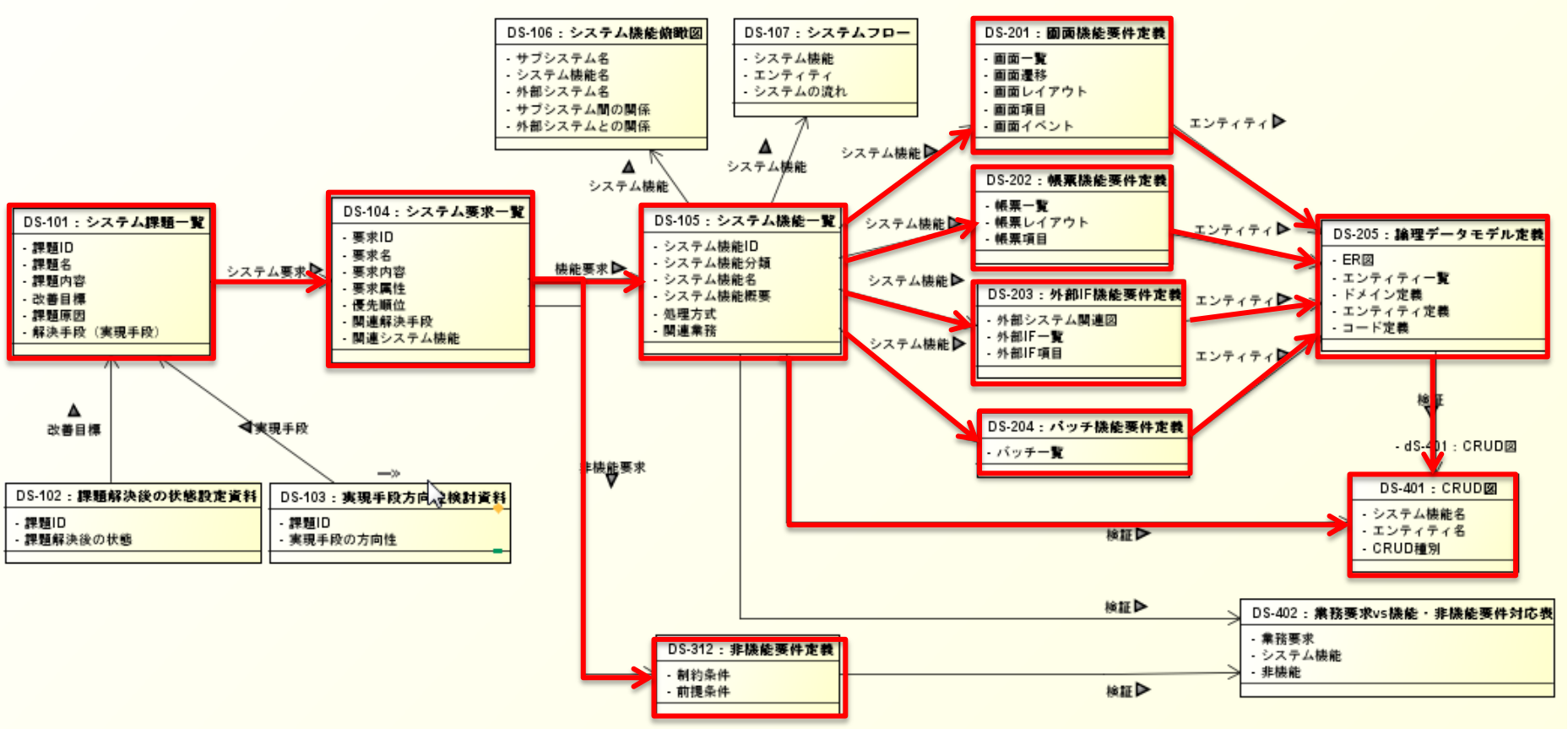


システム要件定義のアウトプット

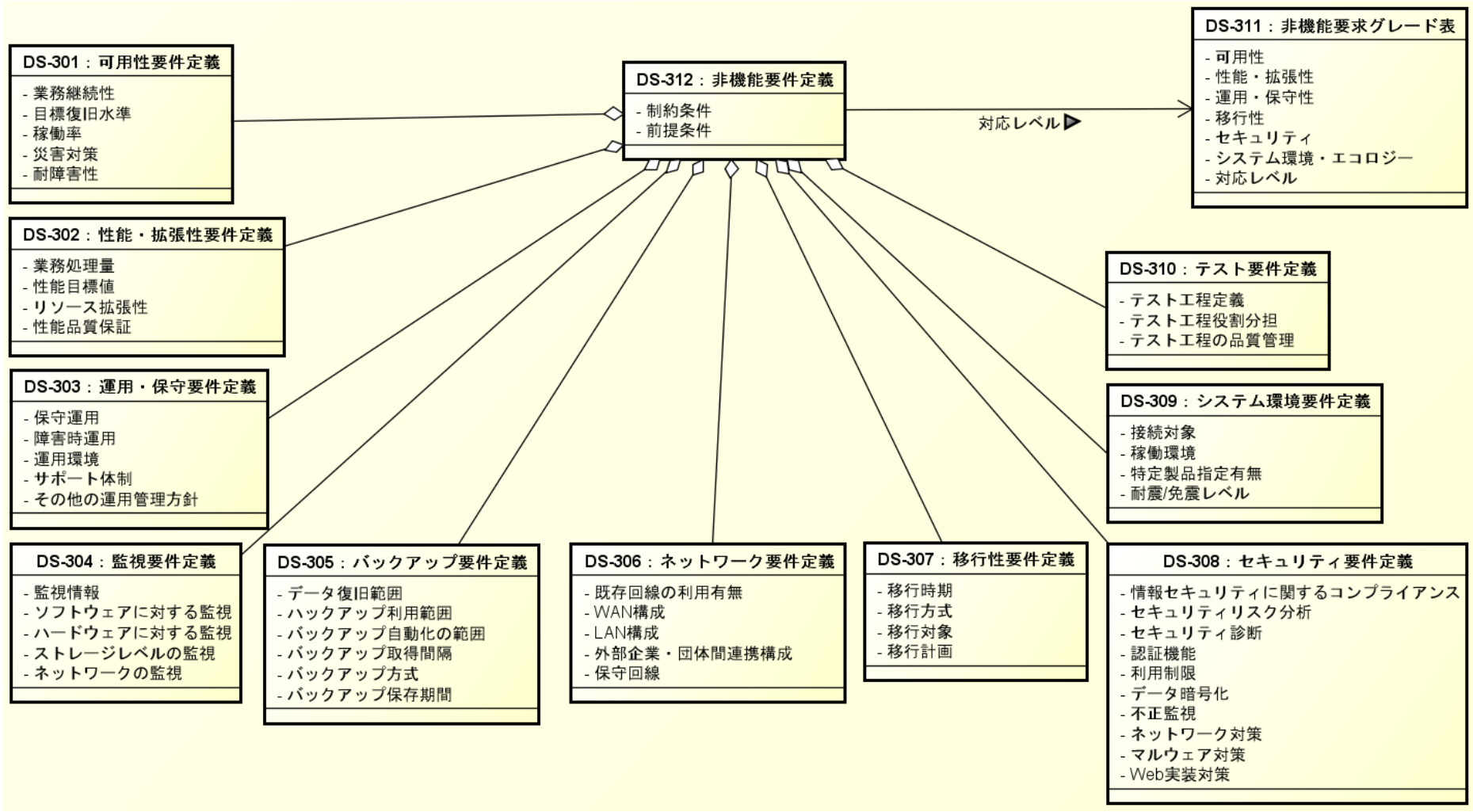
システム要件定義成果物　※抜粋

成果物ID	成果物名	成果物の目的
DS-106	システム機能俯瞰図	新システムのシステム境界とサブシステム定義を含めた 新システムの機能全体像を明確にする。
DS-107	システムフロー	一連のシステム処理の流れを、 システム機能と主要なデータを紐付けて可視化する。
DS-201	画面機能要件定義	画面機能全体に関わる要件(画面一覧、画面フロー、サイトマップ等)と 画面機能個別要件(レイアウト、項目定義、イベント定義等)を明確にする。
DS-202	帳票機能要件定義	帳票機能全体に関わる要件(帳票一覧、帳票処理方式等)と 帳票機能個別要件(帳票項目定義、帳票レイアウト等)を明確にする。
DS-203	外部IF機能要件定義	各外部システムに関わる要件(IF一覧、IF方式等)と IF機能個別要件(IF項目定義、IF手順、データ量、サイクル等)を明確にする。
DS-204	バッチ機能要件定義	バッチ機能全体関わる要件(ジョブネットフロー、ジョブネット一覧、バッチ処理 方式等)とバッチ機能個別要件(入出力定義、処理概要定義等)を明確にする。
DS-205	論理データモデル定義	各機能要件定義にて定義した機能にて利用する エンティティ、項目を明確にする。
DS-312	非機能要件定義書	非機能要件の分類ごとに設けた、各種メトリクスの値・内容を明確にする。
DS-401	CRUD図	論理データモデルに定義したエンティティと機能要件に定義した機能を 紐付け、エンティティと機能の関係を明確にする。

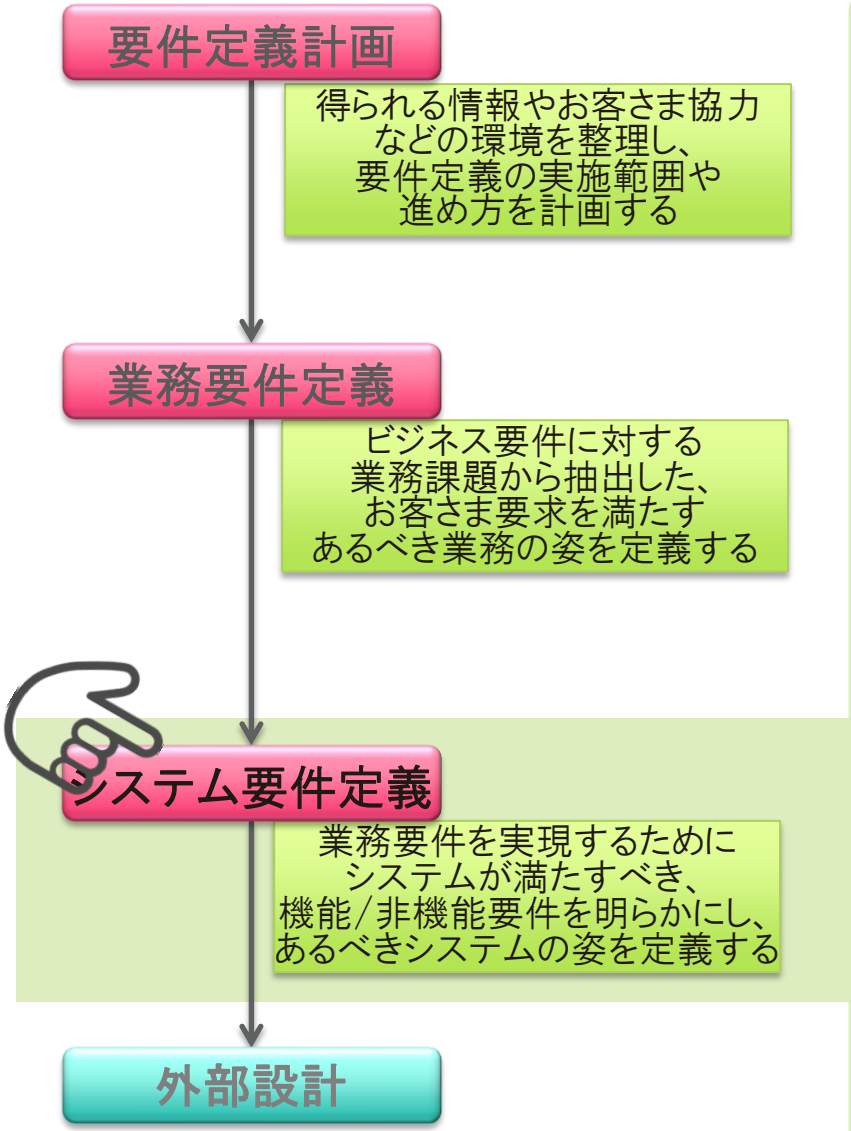
システム要件定義のアウトプット

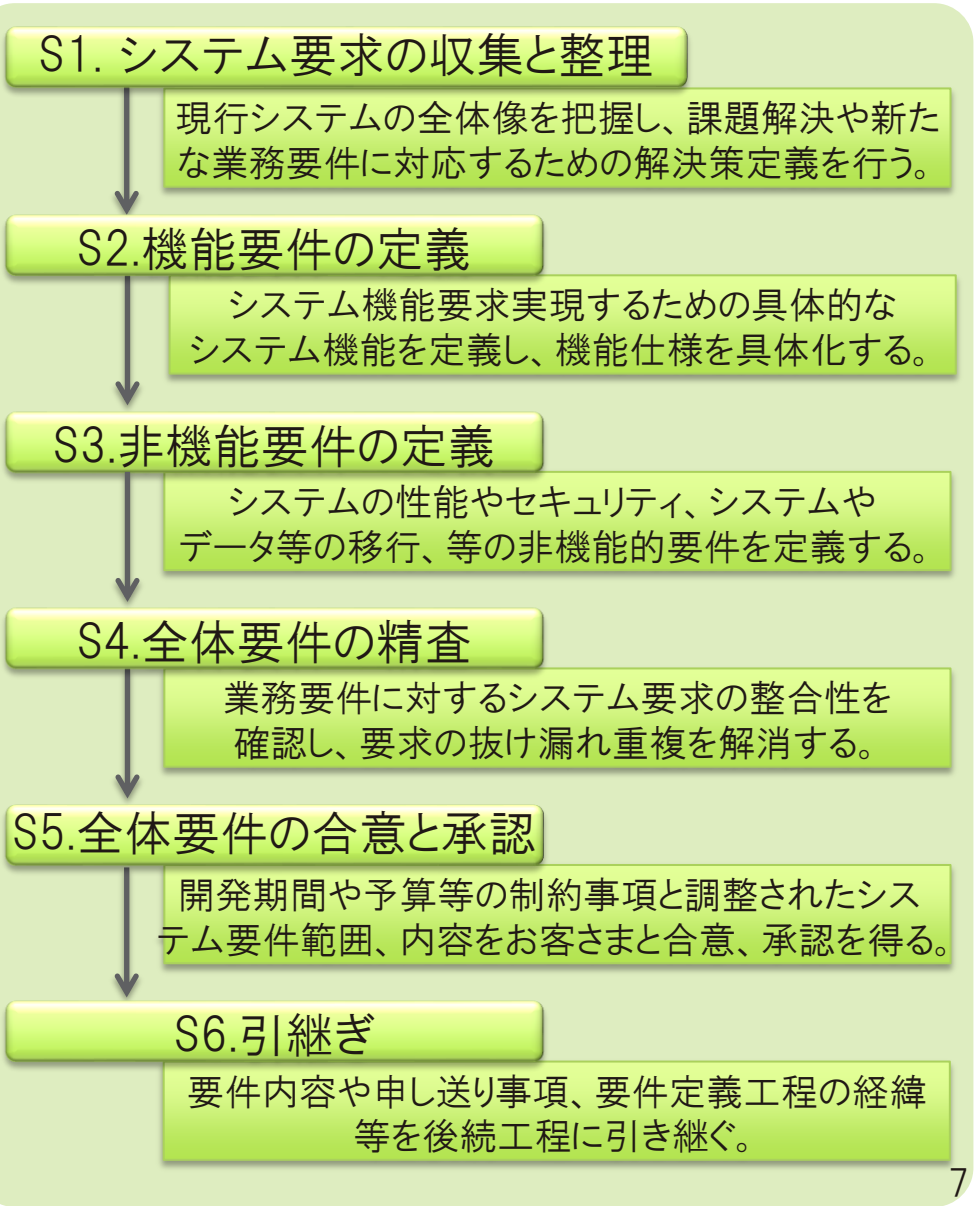


システム要件定義のアウトプット



要件定義プロセス全体におけるシステム要件定義プロセスの位置づけ





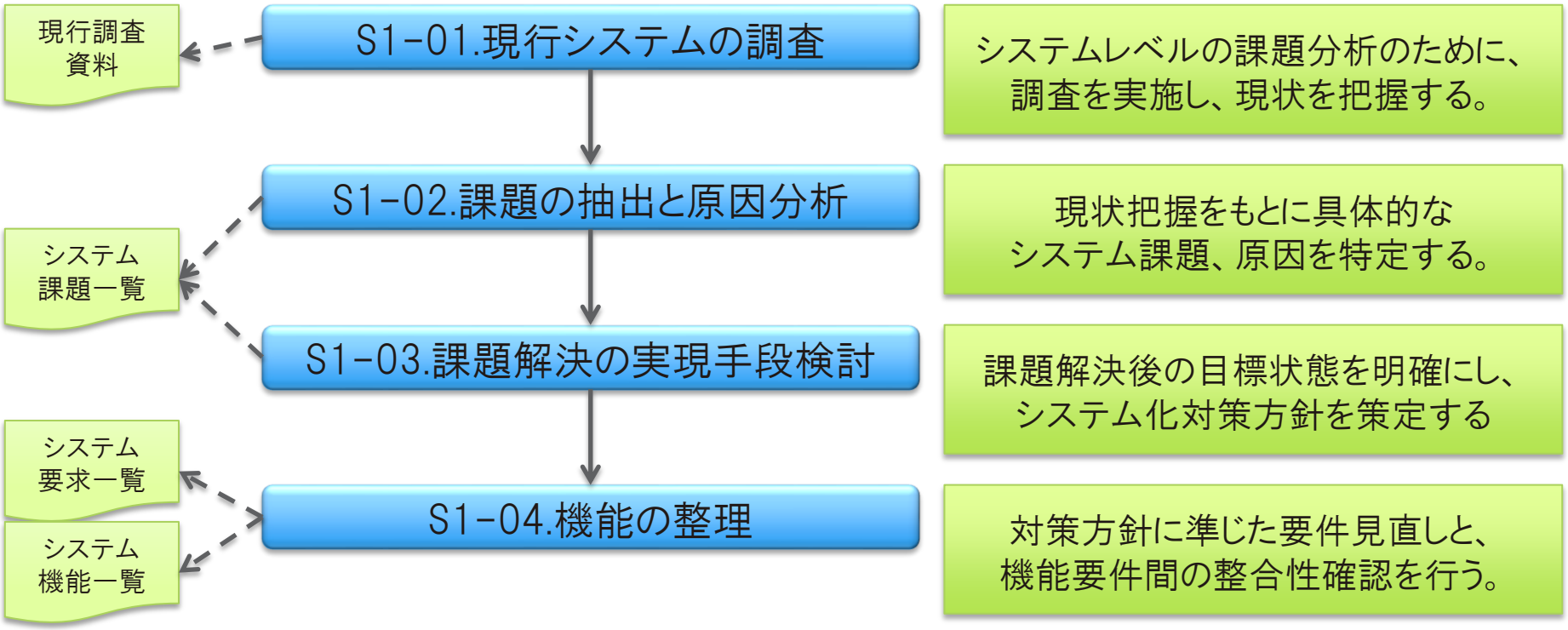
システム要件定義	1. システム要件定義プロセスの概要
	2. システム要求の収集と整理
	3. 機能要件の定義
	4. 非機能要件の定義
	5. 全体要件の精査、合意と承認
	6. 引継ぎ

S1. システム要求の収集と整理

■ このプロセスでの到達目標

現状システムの課題定義、原因分析、解決策(=システム要求)を定義する。
システム要求の内容と目的を明確化し、お客さまと認識を合わせる。

■ サブプロセスフロー



S1. システム要求の収集と整理

■ ポイント

【現行システムの調査】

- 業務要件定義段階での調査結果を踏まえて、
  現行システム課題の理解や原因分析に必要な範囲と詳細度で行う。

【機能の整理】

- 「現行システム課題の原因」、「業務要件実現に関するシステム検討事項」に対する解決手段を、システム要求一覧やシステム機能一覧に反映する。
- システム機能全体での整合性、実現性を確保するために、
  システムフロー図、CRUD図で機能やエンティティの関連を整理する。
- NLPのメタモデル、避けるべき曖昧な用語、等のテクニックを使って、
  要求の曖昧さを排除する。（「曖昧さの排除」は業務要件定義編で解説）

S1. システム要求の収集と整理

■ アンチパターン①

すべてのシステム要求実現を前提としてシステム要件定義を進めてしまう。

■ ポイント

- 業務要件とのトレーサビリティを分析し、必要なシステム要求に絞り込む。
- 改善テーマを設定し、関連するシステム要求に絞り込む。

S1. システム要求の収集と整理

■ アンチパターン②

“遅れた“業務要件確定に伴い、システム要求変更が多発する。

■ ポイント

- 要件定義計画時点で、難易度が高いビジネス要件や業務要件を把握する。
  →該当部分は要件確定遅れに備えた要件定義計画を合意する。
  例)確定遅れの影響取込期間の設定、影響規模の上限合意)
- 業務機能や業務要件ごとに、要件の合意確度や変更影響を見極める。
  →新規のサービス/業務に関わる部分はリスク大
  →お客さまのお客さまが強い要件決定力を持つ場合はリスク大

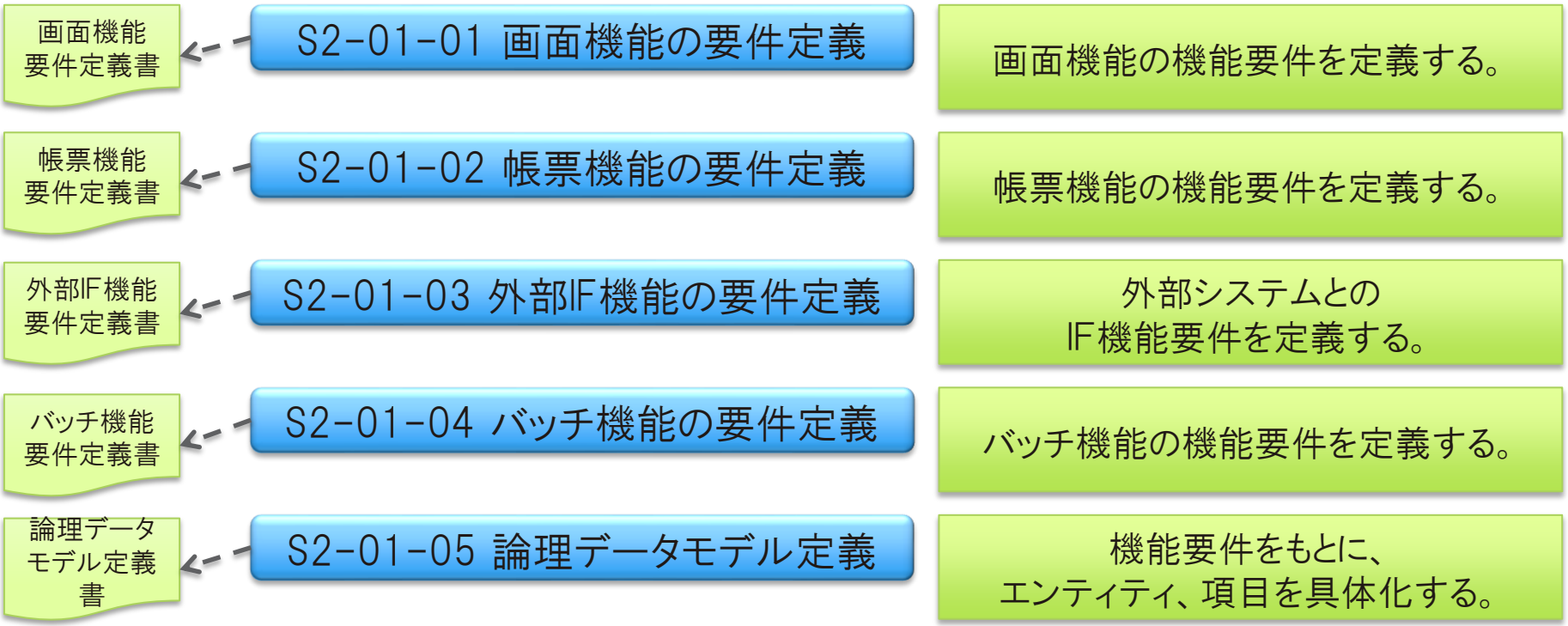
システム要件定義	1. システム要件定義プロセスの概要
	2. システム要求の収集と整理
	3. 機能要件の定義
	4. 非機能要件の定義
	5. 全体要件の精査、合意と承認
	6. 引継ぎ

S2. 機能要件の定義

■ このプロセスでの到達目標

システム機能一覧に整理した各機能要件で実現する機能内容や関係情報を、
設計以降のスコープが明確で、適切な工数見積が可能な状態に詳細化する。

■ サブプロセスフロー



S2. 機能要件の定義

■ ポイント

【定義内容の詳細度】

- 外部設計工程の契約形態や検討内容と整合させる。

  「外部設計から一括請負」の場合は、見積確度維持のため詳細化が必要。
  「外部設計まで準委任」の場合は、一括請負に比べ詳細化の必要性は低い。
  →要件定義フレームワークのサンプルを参考に詳細化する。

- 決定した要件内容の判断理由を文書に残す

  要件に複数の選択肢がある場合や、重要な意図がある場合、決定した要件の判断理由や背景を記録し、後続作業にその判断意図を継承する。
  機能が必要な業務的理由や機能要件の妥当性を示すために、依拠する業務要件を示す。

  保守開発で、既存の要件や機能を理解するための重要なインプットになる。

要件定義フレームワーク
https://github.com/Fintan-contents/requirement-definition-fw

S2. 機能要件の定義

■ ポイント

【認識齟齬、誤解の低減】

- お客さま業務用語、一般用語で記述する。(用語集を遵守)

  システム/開発者寄りの記述を避け、業務内容とシステム活用をお客さまがイメージしやすく

×望ましくない例(プログラムよりの表現)

アクションの処理概要	アクションの処理詳細
入力されたデータをもとに顧客情報テーブルにUpdateする	①入力データの入力チェック
	② Search処理 入力されたデータをもとに顧客情報テーブルをSearchする
	③アンマッチの場合 データをUpdateし、顧客登録画面をクリア表示する
	④Matchするデータが存在した場合
	⑤Exceptionが発生した場合はロールバックする

○望ましい例(業務上の表現)

アクションの処理概要	アクションの処理詳細
入力された情報をもとに顧客情報を登録する	①顧客情報の入力チェック
	②検索処理 入力された情報をもとに顧客情報テーブルを検索する
	③合致する顧客情報が存在しない場合 顧客情報を登録し、顧客登録画面を初期表示する
	④合致する顧客情報が存在した場合
	⑤例外が発生した場合は初期表示する。

※IPA 情報処理推進機構「機能要件の合意形成ガイド」から引用
http://www.ipa.go.jp/sec/softwareengineering/reports/20100331.html

- 機能を使用する状況(シナリオ、利用者、目的、頻度、制約、等)を記述する。
- 正常系、異常系の判別を明確に記述する。(画面遷移、ロジック、等)
- 曖昧さを排除する(モデリングの活用、NLPのメタモデルの活用、等)

S2. 機能要件の定義

■ ポイント

【効率化、品質向上】

- 共通要件は別冊にまとめるなどし、個々の要件記述から分離する。
  - ✓ 画面設計標準、共通画面部品、方式標準、ドメイン定義、用語集等
- 他の要件や機能への影響度が高い部分からレビューを進め、手戻りを減らす。
- レビューでお客さまに確認頂く項目を用意する。
- 当該機能と関連する業務の担当者に、業務要件の充足を確認してもらう。
  - ✓ 業務フローや業務ルール定義と、システム機能要件の内容を突き合わせ
  - ✓ 業務階層定義上の業務と、システム機能一覧上の機能を突き合わせ
- 重要機能を見極めて、かける時間/工数のメリハリをつける。
  - ✓ 業務フロー図、業務ルールをもとにクリティカルな機能をお客さまと確認する。
  - ✓ [保守開発の場合]新規・変更・既存を記述凡例で明確にする。

S2. 機能要件の定義

■ アンチパターン①

処理方式やアプリ基盤との不整合、実現不可能な画面レイアウトなど、
技術的実現性が未検証の機能要件が定義される。

■ ポイント

- ソフトウェア方式担当(アーキテクト)との連携を密に。
  - ✓ アプリケーション機能の実現方法、難易度、複雑度
  - ✓ アプリケーション処理方式を先行して固め、機能要件を方式と整合させる
- ソフトウェア方式要件検討と機能要件検討のスケジュール整合性を確保。

S2. 機能要件の定義

■ アンチパターン②

複雑度が高い機能要件を定義し、設計/製造/テストの効率を下げてしまう。

■ ポイント

- できるだけシンプルな機能要件にまとめる。
  - ✓ お客さまは実装の複雑度を考慮せずに、複数機能の集約を求める場合がある。
  - ✓ 設計・実装・テストの複雑化で、逆にコスト増になることも。
- 「設計」しない。
  - ✓ 要件はお客様への提供を約束するもので設計はシステムとしての実現方法を検討するものである。
  - ✓ 要件定義工程ではお客様との要件合意に注力するべきであって、不必要に設計レベルの事柄を合意するべきではない。

S2. 機能要件の定義

■ アンチパターン③

外部接続先システムの洗い出し漏れにより、後続工程で大きなロスを発生する。
（接続処理方式の追加検討、インフラ方式・設計の見直し等）

■ ポイント

- ステークホルダー分析とCRUD分析で外部IF抽出漏れを防ぐ。
  - ✓ 接続先のニーズによるIFは、ステークホルダー分析で関連外部システム特定から。
  - ✓ 自システムのニーズによるIFは、CRUDでのデータライフサイクル分析から。

S2. 機能要件の定義

■ アンチパターン④

外部接続先との取り決めが不十分で、後工程のコスト・スケジュールに影響する。

■ ポイント

- お客さま、接続先システムの担当を巻き込み、以下を明確にする。
  - ✓ IF仕様、接続方式
  - ✓ 責任境界
  - ✓ プロトコル、セキュリティポリシー
  - ✓ エラーリカバリ方法
  - ✓ テスト方法、環境
  - ✓ 移行・切替のリハーサル、本番実施方法
  - ✓ 費用負担

システム要件定義	1. システム要件定義プロセスの概要
	2. システム要求の収集と整理
	3. 機能要件の定義
	4. 非機能要件の定義
	5. 全体要件の精査、合意と承認
	6. 引継ぎ

要件の分類

分類	説明	例
機能要件	利用者が目的を遂げるために ITシステムが提供するサービス。 システムが実現する画面機能や 出力帳票、バッチ処理機能、外部 システム連携機能を明確化する。	前日に発生したクレジットカード利 用実績をカード番号単位で集計し、 ◯◯センターへ電送する。
非機能要件	ITシステムの機能要件に付随して 必要となる品質要件や制約事項。 システム機能に求められる性能、業 務処理量、セキュリティ、稼働時間 などを明確化する。	クレジットカード利用実績集計は、 3万件/日の実績データを 00:00から01:00の間で集計する。 ※非機能要求グレードの中項目 「業務処理量」観点の要件

非機能要件は、ユーザーの関心が部分的(例えばセキュリティ)で曖昧になりやすい。
業務特性に応じた適切な非機能要件を定義するには、非機能要件に含まれる要素を体系的に理解した上で、お客さまをリードして具体化することが欠かせない。

非機能要件の大分類

■ 網羅的な非機能要件定義には、「非機能要求グレード」等のフレームワークが有効

大項目	説明	例
可用性	システムサービスを継続的に利用可能とする要求	運用スケジュール(稼働時間・停止予定など) 障害、災害時における可動目標
性能・拡張性	システムの性能および将来のシステム拡張に関する要求	業務量および今後の増加見積 システム化対象業務の特性(通常時/ピーク時)
運用・保守性	システムの運用と保守サービスに関する要求	運用中に求められるシステム稼働レベル 問題発生時の対応レベル
移行性	現行システム資産の移行に関する要求	新システムへの移行期間、移行方法 移行対象資産の種類および量
セキュリティ	情報システムの安全性の確保に関する要求	利用制限 不正アクセスの防止
システム環境・エコロジー	システムの設置環境やエコロジーに関する要求	耐震/免震、温度/湿度などのシステム環境 CO2排出量などのエコロジー関連

IPA/SEC 非機能要求グレード:
https://www.ipa.go.jp/sec/softwareengineering/std/ent03-b.html

非機能要求は曖昧なことが多い

要件に即して非機能要求を調整する

■ 要求調整例：バックアップの必要性・頻度・保存期間

非機能要求メトリクス	ユーザ見解	ベンダ見解
バックアップの必要性	コストを安価に抑えたい	障害時のことを考えると必要
バックアップの頻度	月次で収集	日次で収集
バックアップデータの保存期間	できるだけ長く	保管期間は3年

BCPに鑑み
以下の通り調整

非機能要求メトリクス	調整後
バックアップの必要性	バックアップを取る
バックアップの頻度	週次で収集
バックアップデータの保存期間	保管期間は3年

BCP：Business Continuity Plan（事業継続計画）

非機能要求レベルと、コスト・リスクのトレードオフ関係

■ 非機能要求とコスト・リスクの関係

一般的に非機能要求レベルを高くすると、導入コストは大きく、リスクは小さくなる。

【データのバックアップ】

	レベル		導入コスト		リスク	
バックアップ保管期間	長期間	大	コスト大	大	低	小
	短期間	↑	コスト小	↑	中	↓
	保存せず	小	コスト＝0	小	高	大
バックアップ取得間隔	短周期	大	コスト大	大	低	小
	長周期	↑	コスト小	↑	中	↓
	取得せず	小	コスト＝0	小	高	大

- 保管期間が短い場合、アーカイブに使えない、社内規程や法律違反になるリスクがある。
- バックアップが未取得の場合、データ破壊発生時にシステムを復旧できないリスクがある。
- 取得間隔が長いとシステム復旧時に古いデータに戻るリスクがある。

非機能要求メトリクス間の矛盾

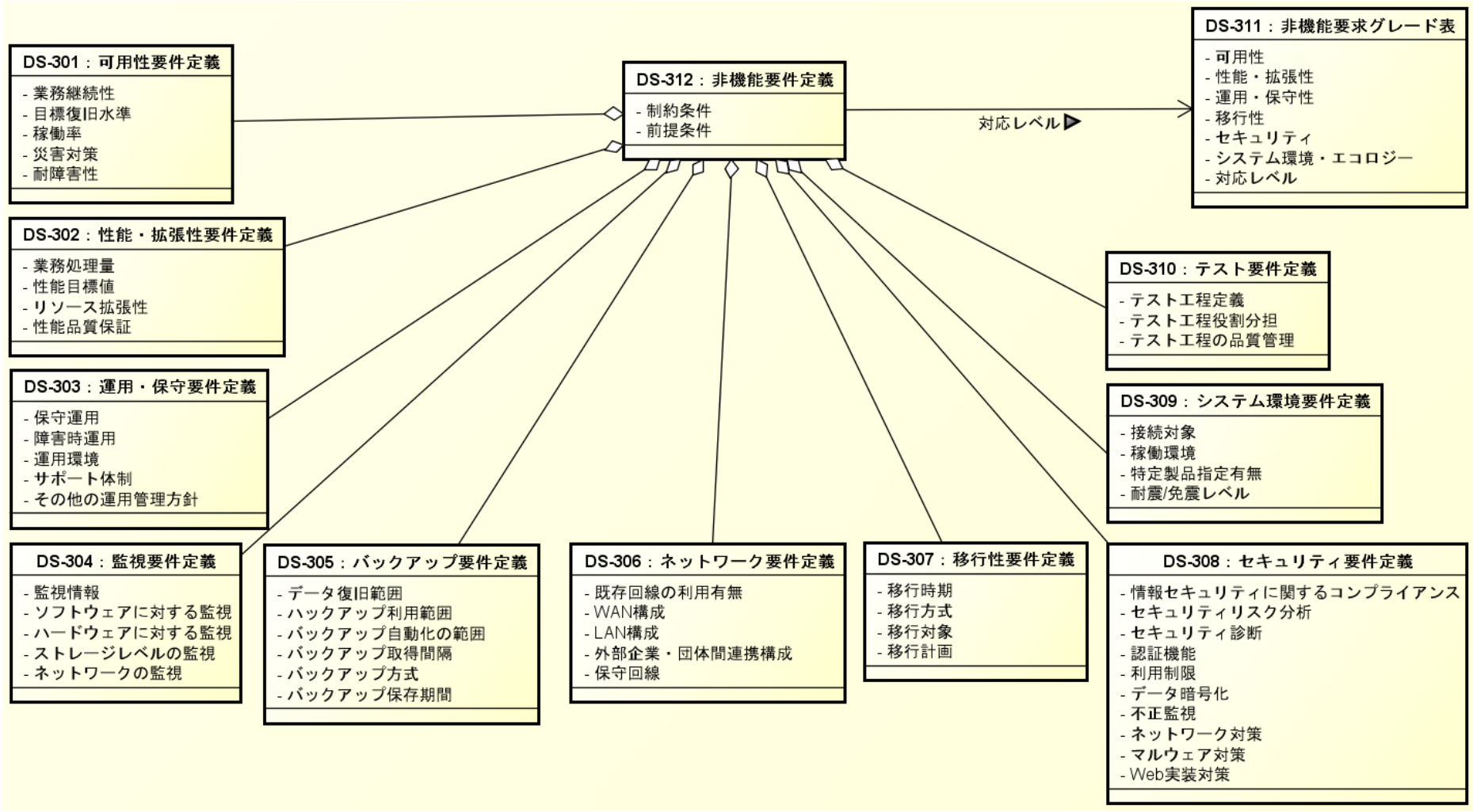
■ 一部の非機能要求メトリクス間には依存関連があり、矛盾がないよう調整が必要。

【例】
① RTOをレベル3の「6時間以内」、サーバ冗長化はレベル0の「非冗長構成」を決定
② 非機能要求全体を確認し、サーバ故障時のRTO「6時間以内」は困難なことが判明
③ RTOを満たすために、冗長化のレベルを1ないし2にレベルアップすることを決定

※非機能要件と機能要件間でのトレードオフが行われる場合もある

項番	大項目	中項目	小項目	メトリクス（指標）	レベル				
					0	1	2	3	4
A.1.3.2	可用性	継続性	目標復旧水準（業務停止時）	RTO（目標復旧時間）	1営業日以上	1営業日以内	12時間以内	**6時間以内**	2時間以内
： ：		： ：	： ：	： ：					
A.2.1.1		耐障害性	サーバ	冗長化（機器）	**非冗長構成**	特定のサーバで冗長化	全てのサーバで冗長化		

システム要件定義のアウトプット

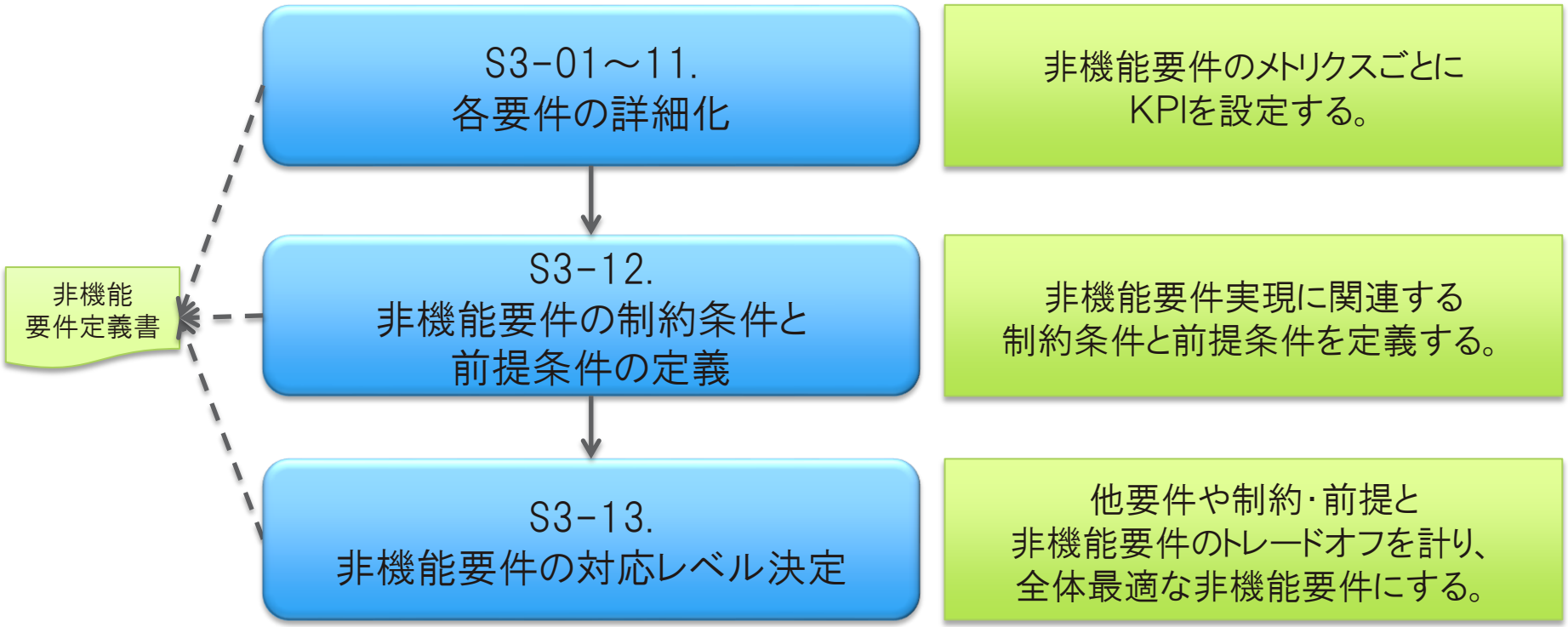


S3. 非機能要件の定義

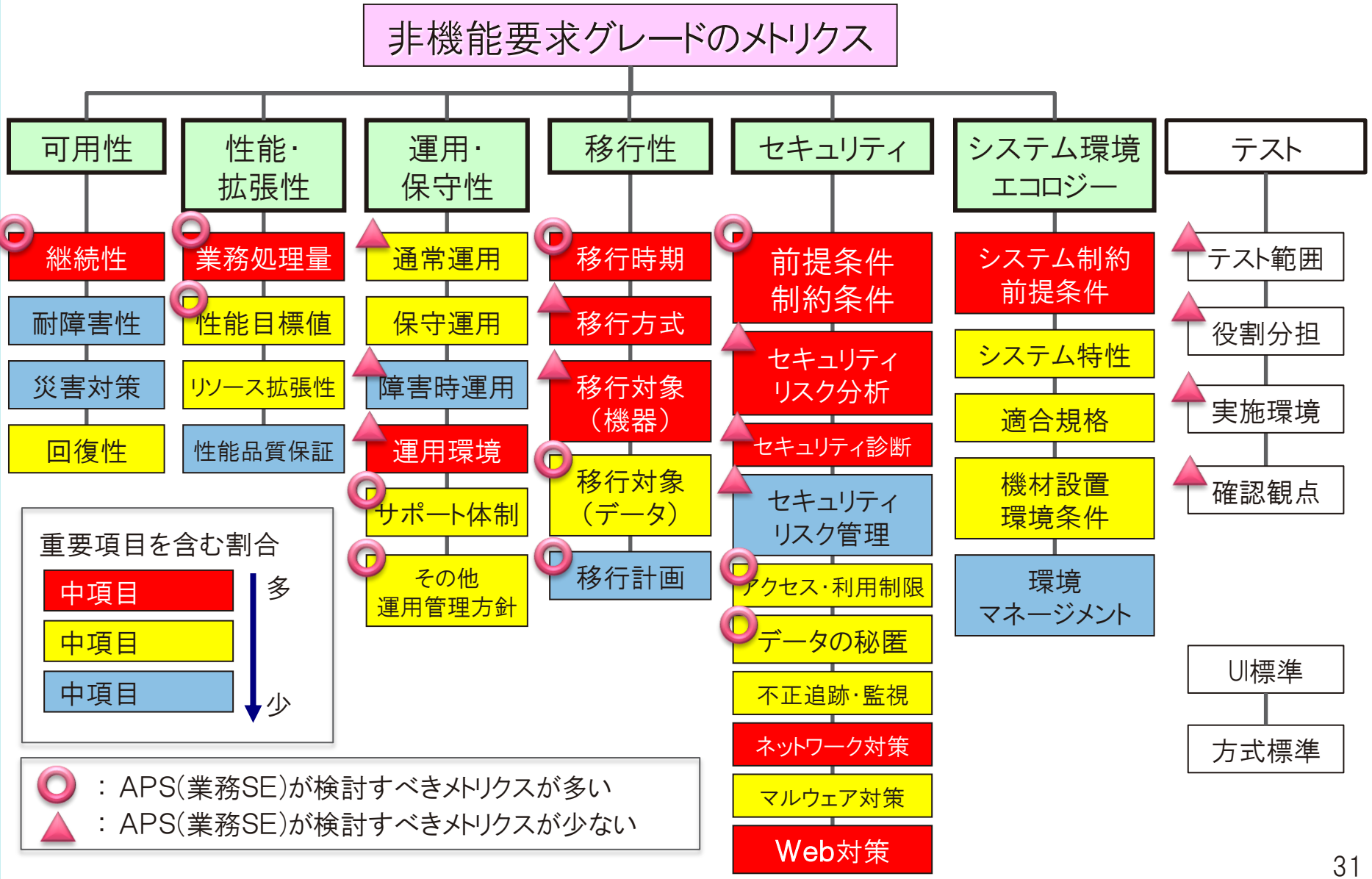
■ このプロセスでの到達目標

システムの性能やセキュリティ、システムやデータの移行等の非機能要件
メトリクスごとに具体的な実現要件を明確化する。

■ サブプロセス



S3. 非機能要件の定義


非機能要求グレードのメトリクス
可用性
継続性
耐障害性
災害対策
回復性
性能・拡張性
業務処理量
性能目標値
リソース拡張性
性能品質保証
運用・保守性
通常運用
保守運用
障害時運用
運用環境
サポート体制
その他運用管理方針
移行性
移行時期
移行方式
移行対象（機器）
移行対象（データ）
移行計画
セキュリティ
前提条件制約条件
セキュリティリスク分析
セキュリティ診断
セキュリティリスク管理
アクセス・利用制限
データの秘匿
不正追跡・監視
ネットワーク対策
マルウェア対策
Web対策
システム環境エコロジー
システム制約前提条件
システム特性
適合規格
機材設置環境条件
環境マネージメント
テスト
テスト範囲
役割分担
実施環境
確認観点
UI標準
方式標準
重要項目を含む割合
中項目
中項目
中項目
多
少
：APS(業務SE)が検討すべきメトリクスが多い
：APS(業務SE)が検討すべきメトリクスが少ない

S3. 非機能要件の定義

■ 非機能要件メトリクス

大項目	中項目	小項目	メトリクス
可用性	継続性	運用スケジュール	運用時間(通常、休日／祝祭日)、計画停止の有無
		業務継続性	対象業務範囲、サービス切替時間、業務継続要求度
		目標復旧水準 (業務停止時)	目標復旧地点(RPO)、目標復旧時間(RTO)、目標復旧レベル(RLO)
		目標復旧水準 (大規模災害時)	システム再開目標
		稼働率	稼働率
	耐障害性	サーバ	冗長化(機器)、冗長化(コンポーネント)
		端末	冗長化(機器)、冗長化(コンポーネント)
		ネットワーク機器	冗長化(機器)、冗長化(コンポーネント)
		ネットワーク	回線の冗長化、線路の冗長化、セグメント分割
		ストレージ	冗長化(機器)、冗長化(コンポーネント)、冗長化(ディスク)
		データ	バックアップ方式、データ復旧範囲、データインテグリティ
	災害対策	システム	復旧方針
		外部保管データ	保管場所分散度、保管方法
		付帯設備	災害対策範囲
	回復性	復旧作業	復旧作業、代替業務運用の範囲
		可用性確認	確認範囲

S3. 非機能要件の定義

■ 非機能要件メトリクス(つづき)

大項目	中項目	小項目	メトリクス
性能・拡張性	業務処理量	通常時の業務量	ユーザ数、同時アクセス数、データ量、オンラインリクエスト件数、 バッチ処理件数、業務機能数
		業務量増大度	ユーザ数増大率、同時アクセス数増大率、データ量増大率、 オンラインリクエスト数増大率、バッチ処理件数増大率、業務機能数増大率
		保管期間	保管期間、対象範囲
	性能目標値	オンラインレスポンス	レスポンス順守率(通常時・ピーク時・縮退時)
		バッチレスポンス	レスポンス順守度合い(通常時・ピーク時・縮退時)
		オンラインスループット	処理余裕率(通常時・ピーク時・縮退時)
		バッチスループット	処理余裕率(通常時・ピーク時・縮退時)
		帳票印刷能力	処理余裕率(通常時・ピーク時・縮退時)
	リソース拡張性	CPU拡張性	CPU利用率、CPU搭載余裕有無
		メモリ拡張性	メモリ利用率、メモリ搭載余裕有無
		ディスク拡張性	ディスク利用率、ディスク増設余裕有無
		ネットワーク	ネットワーク機器設置範囲
		サーバ処理能力増強	スケールアップ、スケールアウト
	性能品質保証	帯域保証機能の有無	帯域保証の設定
		性能テスト	測定頻度、確認範囲
		スパイク負荷対応	トランザクション保護

S3. 非機能要件の定義

■ 非機能要件メトリクス(つづき)

大項目	中項目	小項目	メトリクス
運用・保守性	通常運用	運用時間	<重複>運用時間(通常・特定日)
		バックアップ	<重複>データ復旧範囲、外部データの利用可否、バックアップ利用範囲、 バックアップ自動化の範囲、バックアップ取得間隔、バックアップ保存期間、 <重複>バックアップ方式
		運用監視	監視情報、監視間隔、システムレベルの監視、プロセスレベルの監視、 データベースレベルの監視、ストレージレベルの監視、 サーバ(ノード)レベルの監視、端末／ネットワーク機器レベルの監視、 ネットワーク・パケットレベルの監視
		時刻同期	時刻同期設定の範囲
	保守運用	計画停止	<重複>計画停止の有無、計画停止の事前アナウンス
		運用負荷削減	保守作業自動化の範囲、サーバソフトウェア更新作業の自動化、 端末ソフトウェア更新作業の自動化
		パッチ適用ポリシー	パッチリリース情報の提供、パッチ適用方針、パッチ適用タイミング、 パッチ検証の実施有無
		活性保守	ハードウェア活性保守の範囲、ソフトウェア活性保守の範囲
		定期保守頻度	定期保守頻度
		予防保守レベル	予防保守レベル
	障害時運用	復旧作業	<重複>復旧作業、<重複>代替業務運用の範囲
		障害復旧自動化の範囲	障害復旧自動化の範囲
		システム異常検知時の対応	対応可能時間、駆けつけ到着時間、SE到着平均時間
		交換用部材の確保	交換部品確保レベル、予備機の有無

S3. 非機能要件の定義

■ 非機能要件メトリクス(つづき)

大項目	中項目	小項目	メトリクス
運用・保守性	運用環境	開発用環境の設置	開発用環境の設置有無
		試験用環境の設置	試験用環境の設置有無
		マニュアル準備レベル	マニュアル準備レベル(運用、保守など)
		リモートオペレーション	リモート監視地点、リモート操作の範囲
		外部システム接続	外部システムとの接続有無、監視システムの有無、 ジョブ管理システムの有無
	サポート体制	保守契約(ハードウェア)	保守契約(ハードウェア)の範囲
		保守契約(ソフトウェア)	保守契約(ソフトウェア)の範囲
		ライフサイクル期間	ライフサイクル期間
		メンテナンス作業役割分担	メンテナンス作業役割分担
		一次対応役割分担	一次対応役割分担
		サポート要員	ベンダ側常備配備人数、ベンダ側対応時間帯、 ベンダ側対応者のスキルレベル、エスカレーション対応
		導入サポート	システムテスト稼働時の導入サポート期間、 システム本格稼働時の導入サポート期間
		オペレーション訓練	オペレーション訓練実施の役割分担、 オペレーション訓練範囲、オペレーション訓練実施頻度
		定期報告会	定期報告会実施頻度、報告内容のレベル
	その他の 運用管理方針	内部統制対応	内部統制対応実施の有無
		サービスデスク	サービスデスクの設置有無
		インシデント管理	インシデント管理の実施有無
		問題管理	問題管理の実施有無
		構成管理	構成管理の実施有無
		変更管理	変更管理の実施有無
		リソース管理	リソース管理の実施有無

S3. 非機能要件の定義

■ 非機能要件メトリクス(つづき)

大項目	中項目	小項目	メトリクス
移行性	移行時期	移行のスケジュール	システム移行期間、システム停止可能日時、並行稼働の有無
	移行方式	システム展開方式	拠点展開ステップ数、業務展開ステップ数
	移行対象機器	移行設備	設備・機器の移行内容
	移行対象データ	移行データ量	移行データ量、移行データ形式
		移行媒体	移行媒体量、移行媒体種類数
		変換対象(DBなど)	変換データ量、移行ツールの複雑度(変換ルール数)
	移行計画	移行作業分担	移行のユーザ／ベンダ作業分担
		リハーサル	リハーサル範囲、リハーサル環境、リハーサル回数、外部連携リハーサルの有無
		トラブル対処	トラブル対処の規定有無

S3. 非機能要件の定義

■ 非機能要件メトリクス(つづき)

大項目	中項目	小項目	メトリクス
セキュリティ	前提条件・制約条件	情報セキュリティに関するコンプライアンス	順守すべき社内規程、ルール、法令、ガイドライン等の有無
	セキュリティリスク分析	セキュリティリスク分析	リスク分析範囲
	セキュリティ診断	セキュリティ診断	ネットワーク診断実施の有無、Web診断実施の有無、DB診断実施の有無
	セキュリティリスク管理	セキュリティリスクの見直し	セキュリティリスク見直し頻度、セキュリティリスクの見直し範囲
		セキュリティリスク対策の見直し	運用開始後のリスク対応範囲、リスク対策方針
		セキュリティパッチ適用	セキュリティパッチ適用範囲、セキュリティパッチ適用方針、 セキュリティパッチ適用タイミング
	アクセス・利用制限	認証機能	管理権限を持つ主体の認証、管理者権限を持たない主体の認証
		利用制限	システム上の対策における操作制限度、物理的な対策による操作限度
		管理方法	管理ルールの策定
	データの秘匿	データ暗号化	伝送データの暗号化の有無、蓄積データの暗号化の有無、鍵管理
	不正追跡・監視	不正監視	ログの取得、ログ保管期間、確認間隔、不正監視対象 (装置、ネットワーク、侵入者、不正操作等)
		データ検証	デジタル署名の利用の有無、確認間隔
	ネットワーク対策	ネットワーク制御	通信制御
		不正検知	不正通信の検知範囲
		サービス停止攻撃の回避	ネットワークの輻輳対策
	マルウェア対策	マルウェア対策	マルウェア対策実施範囲、リアルタイムスキャンの実施、 フルスキャンの定期チェックタイミング
	Web対策	Web実装対策	セキュアコーディング、Webサーバの設定等による対策の強化、 WAFの導入の有無

S3. 非機能要件の定義

■ 非機能要件メトリクス(つづき)

大項目	中項目	小項目	メトリクス
システム環境・エコロジー	システム制約／前提条件	構築時の制約条件	構築時の制約条件
		運用時の制約条件	運用時の制約条件
	システム特性	ユーザ数	<重複>ユーザ数
		クライアント数	クライアント数(上限の有無等)
		拠点数	拠点数(複数か否か)
		地域的広がり	広がりの範囲(同一都市内、県内等)
		特定製品指定	特定製品の採用の有無
		システム利用範囲	システム利用範囲
		複数言語対応	言語数
	適合規格	製品安全規格	規格取得必要性の有無
		環境保護	規格取得必要性の有無
		電磁干渉	規格取得の有無
	機材設置環境条件	耐震／免震	耐震震度
		スペース	設置スペース制限(マシン室、事務所設置)、 並行稼働スペース(移行時)、設置スペースの拡張余地
		重量	床加重、設置対策
		電気設備適合性	供給電力適合性、電源容量の制約、並行稼働電力(移行時)、停電対策、 想定設置場所の電圧変動、想定設置場所の周波数変動、接地
		温度(帯域)	温度(帯域)
		湿度(帯域)	湿度(帯域)
		空調性能	空調性能、空調設備の制約(カスタマイズの必要性)
	環境マネージメント	環境負荷を抑える工夫	グリーン購入法対応度、同一機材拡張余力、機材のライフサイクル期間
		エネルギー消費効率	エネルギー消費の目標値
		CO2排出量	CO2排出量の目標値
		低騒音	騒音値

S3. 非機能要件の定義

■ アンチパターン①

アーキテクト、インフラ、パッケージ(ベンダー)との間で、
非機能要件の検討やその実現に関する役割分担に齟齬が生じる。

■ ポイント

- アーキテクト・インフラとの連携を密にする
  - ✓ 非機能要件の分類やメトリクスごとに、担当範囲を整理する
- 非機能要件の実現可能性を担保する。
  - ✓ 非機能要件の実現方法やコスト等の概略に関して、断続的な相談・調整の場を設ける

S3. 非機能要件の定義

■ アンチパターン②

必要な根拠や実現性、所要コストの評価なく、非機能要件メトリクスが設定される。
オーバースペックになる。

■ ポイント

- 非機能要件の設定根拠をできるだけ明確にする
  - ✓ 業務要件から抽出(トップダウン)、個別メトリクスに対するお客さま要求(ボトムアップ)
  - ✓ 各非機能要件メトリクスの妥当性が評価・説明できるようになる

S3. 非機能要件の定義

■ アンチパターン③

非機能要求グレードのすべての非機能要件メトリクスを設定しようとする。
非機能要求グレードのメトリクスが非機能要求のすべてだと考えてしまう。

■ ポイント

- 非機能要求グレードは「モデル」。PJごとにテーラリングが必要。
  - ✓ 考慮不要な非機能要件メトリクスには“不要”と明記し、合意する。
  - ✓ 非機能要求グレードはインフラで実現する項目が対象であり、完全ではない。

S3. 非機能要件の定義

■ アンチパターン④

内容不足がおこりがちな非機能要件

■ ポイント

- 非機能要件の対象を明確にする。
  - ✓ 性能目標値は全画面なのか？全トランザクションなのか？
  - ✓ セキュリティ要件の適用対象は全機能なのか？
- 処理件数ピークの特性(発生周期や時期、発生要因)を明らかにする。
- 具体的・現実的な性能目標値を設定する。
  - ✓ 全機能一律の性能目標値を設定しない。(全画面レスポンス3秒、など)
    機能あるいは機能グループごとに、性能目標値と前提を定義する。

    目標値：機能ごとに、機能性と性能の重み付け
    前提：処理量と想定時期、通常/ピーク、計測範囲、達成率、例外条件
- 処理方式や設計/実装標準と、実装の乖離チェックを計画する。
- テスト要件として、実施範囲、観点、環境、方法、役割、前提等を定義する。

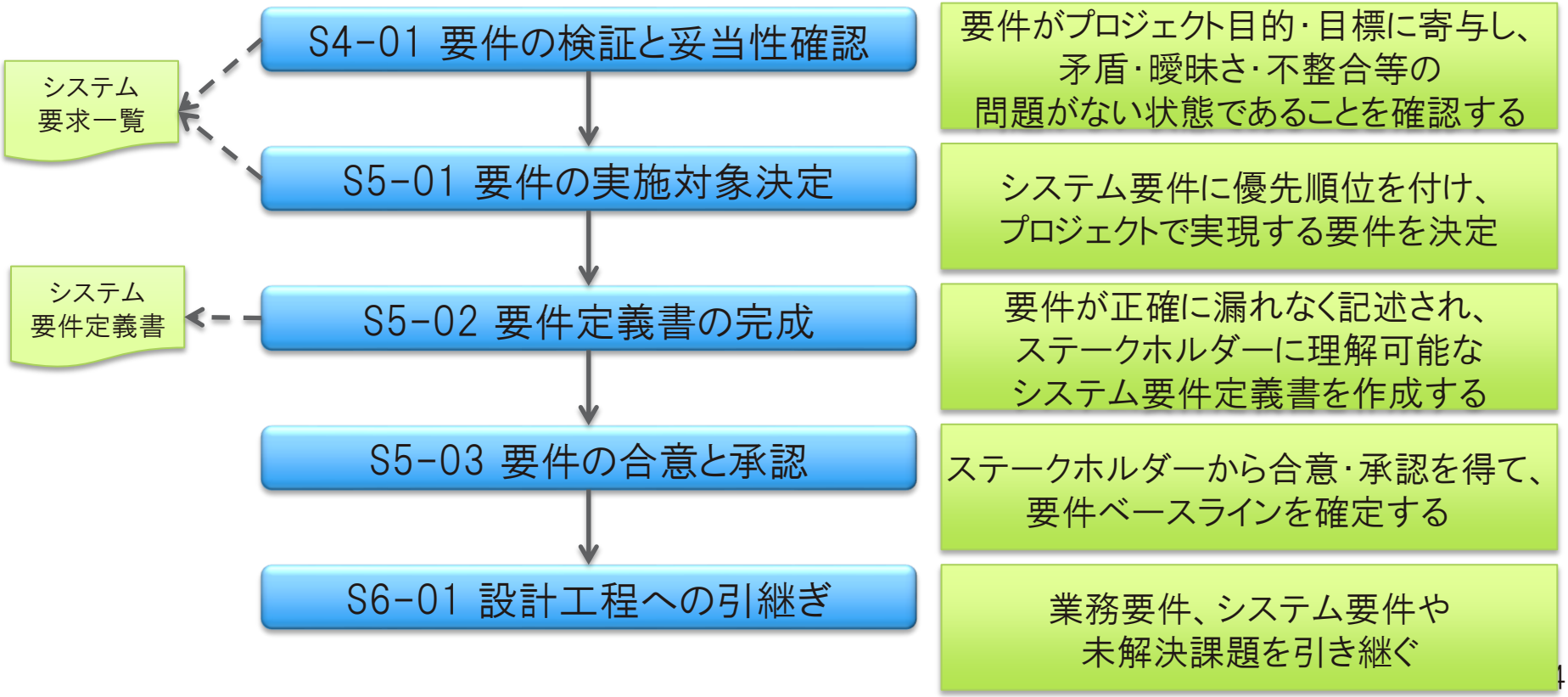
システム要件定義	1. システム要件定義プロセスの概要
	2. システム要求の収集と整理
	3. 機能要件の定義
	4. 非機能要件の定義
	5. 全体要件の精査、合意と承認
	6. 引継ぎ

S4. 全体要件の精査
S5. 全体要件の合意と承認
S6. 引継ぎ

■ このプロセスでの到達目標

システム要件がプロジェクトの目的・目標に寄与し、矛盾・曖昧さ・不整合等の問題がない状態にする。またそのシステム要件が、背景・制約・前提等とともに文書化されてベースラインとなり、ステークホルダー間で共有できる状態にする。

■ サブプロセスフロー



S4. 全体要件の精査

■ 機能要件と非機能要件の検証

機能要件、非機能要件が、要件の特性（完全性、無曖昧性、など）に照らして正しいことを検証する。

- 機能要件内の検証

  （例）機能間の関係や連携の視点で機能要件の矛盾や漏れを検証

- 非機能要件内の検証

  （例）非機能要件間で相反する要件が定義されていないことを検証

- 機能要件と非機能要件間の検証

  （例）機能要件と非機能要件に相反する要件が定義されている機能を対象に実現性を検証 （複雑な機能要件と高度な性能要件を持つ機能など）
  （例）機能が実行処理方式標準等の標準に準拠していることを検証

■ 機能要件と非機能要件の妥当性確認

業務要件の実現に足る機能要件、非機能要件が定義されていることを確認する。

※ロジックツリーを利用した妥当性確認のイメージについては、「業務要件定義編」で解説

「システム要件定義の進め方」のまとめ

プロセス	主要サブプロセス	ポイント
システム要求の収集と整理	現行システムの調査	• トップダウンでシステムを俯瞰・網羅 • 現行踏襲でも必要な調査は実施
	課題解決の実現手段検討	• トップダウンとボトムアップのアプローチ • システムの整合性、実現性確保
機能要件の定義	画面/帳票/外部IF/バッチ機能の要件定義	• 設計しない • 実現方法を担保し、外部IFを漏らさない
非機能要件の定義		• 体系的、網羅的なメトリクス定義 • 非機能要件メトリクスの要否の整理 • アーキ/インフラとの緊密な連携 • 設定根拠の明確化
全体要件の精査	要件の検証・妥当性確認	–
全体要件の合意と承認	要件の実施対象決定	–
引継ぎ	設計工程への引継ぎ	–

【補講】特定種類のPJでの要件定義

開発方式によって、要件定義のやり方は変わるか？

- 本研修の説明内容は特定の開発方式に依存しない普遍的なもの。
  しかし、開発方式固有の効果的なアプローチは存在する。

- スクラッチ新規開発プロジェクト以外の要件定義
  - ✓ 保守開発プロジェクト
  - ✓ パッケージSI開発プロジェクト
  - ✓ アジャイル開発プロジェクト

保守開発プロジェクト①

■ アンチパターン

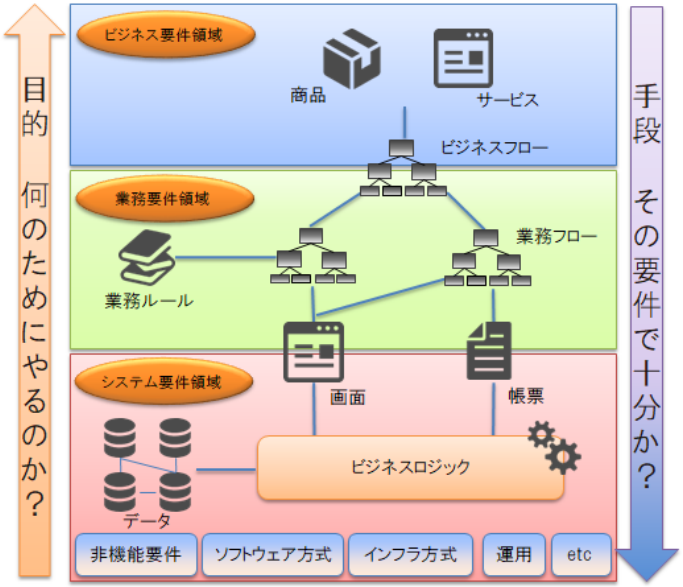
ビジネス要求達成に不必要な要求を、お客さまの要求どおりに実装してしまう。

■ ポイント

- お客さま要求の背後にある課題や上位要求を確認する。

✓ 保守開発の場合、お客さまは動いている現行システムを見て要求を出せるため、要求内容が具体的なシステム要求に偏りやすく、必要性が不明確になりやすい。

✓ 要求の背後にある課題や、上位のビジネス要求や業務要求を把握し、お客さま要求の必要性や一貫性を確認する。



保守開発プロジェクト②

■ アンチパターン

要件定義コストが限られ、新規開発と同じ成果物は作りきれない。

■ ポイント

- USDM等のモデルを活用して、最小限のコストで要求を構造化する。
  - ✓『現行システムの要件定義文書がない』、『メンテナンスされていない』などの理由で、既存資産を保守開発で活用できないケースがある。
    またコストが不足し、新規開発と同等の成果物を作ることは難しい。

- 詳細化が必要な業務要求、システム要求に注力して文書化する。
  - ✓ 文書を必要としないレベルで認識を共有できている範囲の文書化は不要。認識齟齬による影響が大きい箇所や、認識齟齬の可能性が大きい箇所に注力して文書化するのがよい。

■ アンチパターン

パッケージ機能とお客さま業務の乖離によるアドオン/カスタマイズ要件が拡大する

■ ポイント

- 当該パッケージの導入経験がある有識者を提案段階から参画させる。
  - ✓ お客さま要求や業務とパッケージ機能間の極端な乖離や、パッケージ機能に対するお客さまの誤解がないことを提案段階で担保し、要件定義段階での大きな問題発生を予防する。
- 業務要求とシステム要求を並列して整理する。
  - ✓ パッケージが前提とする業務プロセス・ルール・情報構造との整合を無視した形で、あるべき業務の姿を求める業務要件定義を進めると、業務要件を満たすために多くのアドオン/カスタマイズが必要になり、コスト超過を招く。

■ アンチパターン

要件定義が不十分な状態で開発イテレーションを開始し、
適切な開発優先順位判断や、スピード感ある開発・リリースが困難になる。

■ ポイント

- 開発イテレーション開始前に要求整理を実施する。
  - ✓『ステークホルダが実現したい業務や得たい価値』『必要な機能』を開発イテレーション開始前にプロダクトバックログにまとめる。（≒要件定義）
  - ✓ 開発イテレーション内では、スプリントバックログの『必要な機能』をプロダクトオーナーと具体化し、設計・実装・テストを行う。

アジャイル開発プロジェクト