Digital Architecture Design Center

IPA Better Life with IT

DADC アーキテクチャ成果物グリッド

解説書

独立行政法人情報処理推進機構（IPA）

デジタルアーキテクチャ・デザインセンター（DADC）

2025 年 3 月 31 日

改訂履歴

Ver	内容
1.0.0	初版公開 2025/03/31

目次

1. はじめに 4
1.1. 本書の位置付け 4
1.2. Society5.0 のアーキテクチャとその目的 4
1.3. DADC グリッドを使用する目的 6
1.4. Soceity5.0 を実現するシステムのアーキテクチャを定義する大まかな流れ 6
2. DADC グリッドの概要 11
2.1. DADC グリッドの基本構成 11
2.2. グリッドを構成する要素の関係性 12
2.3. DADC グリッドの使用方法 13
3. DADC グリッドを構成するセル 14
3.1. セルの特定 14
3.2. 各セルの記載内容 15
3.2.1. SoS コンテクストレベルの前提条件（TCn-Pre） 15
3.2.2. SoS 対象範囲（XCn-Scp） 16
3.2.3. SoS とコンテクストとの関係性（XCn- Cnt） 16
3.2.4. SoS 要求（XCn- Req） 17
3.2.5. SoS ビジョン・コンセプト（XCn- Cnc） 17
3.2.6. ロジックの定義と構成（XCn- LT） 17
3.2.7. 組織定義（TCn- Org） 17
3.2.8. SoS とコンテクストとの関係性詳細 （XSS- Cnt） 17
3.2.9. SoS アーキテクチャレベルの要求（XSS- Req） 18
3.2.10. SoS アーキテクチャレベルのコンセプト（XSS- Cnc） 18
3.2.11. SoS 機能定義と構成（XSS- LT） 18
3.2.12. SoS 機能フロー（XSS- LF） 18

3.2.13. SoS データ・アイテム（XSS- Dt）.....18
3.2.14. SoS 実現手段（HW/SW/人）（XSS- PE）.....18
3.2.15. SoS 実現手段（ルール）（XSS- Rl）.....19
3.2.16. SoS アロケーションの構成（XSS- Alc）.....19
3.2.17. SoS 実現手段間の関係性（XSS- SD）.....19
3.2.18. SoS 実現手段間のインターフェース（XSS- IF）.....19
3.2.19. SoS 実施組織定義（XSS- Org）.....19
3.2.20. SoS 価値連鎖分析（XSS- VC）.....20
3.2.21. SoS リスク分析（XSS- RA）.....20
3.2.22. SoS 設計根拠（XSS- AR）.....20
3.2.23. SoS ロードマップ（XSS- RM）.....20
3.2.24. 構成システムアーキテクチャ 前提（XCS- Pre）.....20
3.2.25. 構成システムアーキテクチャ スコープ（XCS-Scp）.....20
3.2.26. 構成システムアーキテクチャと外部システムの関係性（XCS-Cnt）.....21
3.2.27. 構成システムアーキテクチャ その他のセル.....21
3.3. As-Is に関する留意点.....21
4. おわりに.....22

1. はじめに

1.1. 本書の位置付け

「DADC アーキテクチャ成果物グリッド」(以下 DADC グリッド) は、Society5.0 のアーキテクチャをデザインするメソッドの一部として、Society5.0 を実現するシステム設計において共通的に必要となる基本的なビュー (設計結果として表現するもの) のセットを定義している。これは Society5.0 がサイバーフィジカルシステムを含む社会システムであることから、既存の情報システムやエンタープライズに対応した設計メソッドでは十分に表現できない、もしくは観点を変えて捉える必要があるとの考えから作成したものである。システムズエンジニアリング (ISO/IEC/IEEE42010 等の標準を含む) をベースに、既存の各種アーキテクチャフレームワークを参考にしつつも、適用において過度な負荷がかからないように検討した。

本書では、グリッドの各セルとセル間の関係について説明する。また必要に応じて、グリッドを活用する上で必要となる最小限のシステムズエンジニアリングに関する補足を行う。これは Society5.0 の実現に資するシステムを設計する者が参照することで、考慮すべき点の抜け漏れを抑え、異なる領域の設計者同士の会話が促進されることを狙ったものである。グリッドでは共通的に必要と考えられるビューに絞って記載している為、設計対象に応じて追加で必要なビューがある場合には、別途追加して使用することを想定している。

読者としてはビジネスエコシステムの変革を企図する者、Society5.0 を担う System of Systems の設計者、そして System of Systems を構成する個々のシステムの設計者を想定した。

1.2. Society5.0 のアーキテクチャとその目的

まず“システムアーキテクチャ”とは、システムを構成する要素とその要素間の関係性であり、システムアーキテクチャを定義し可視化することには幾つかのメリットがある。主なメリットを列挙するならば、1 点目として、設計対象となるシステムが何を、どれ程よく、どのように目的を達成しようとしているのか、根拠を持って示すことができる点がある。これは、設計者の想定範囲を示すことで、その抜け漏れの発見や、ステークホルダとの議論及び合意形成に役立つ。また、想定した実現手段に応じてリスク分析を行うことで、リスクに対応した設計を行うことにも繋がる。

2 点目として、アーキテクチャ定義においては、複数のビューポイント、ビューから、それぞれ設計対象システムを検討し、その結果を統合して 1 つのシステムとして取りまとめられる点が挙げられる。複雑なシステムの設計に際しては様々な専門性を持つ人の知見を得て設計を進めることが不可欠であるが、その専門家毎の異なる観点に基づく知見を束ねる上で、アーキテクチャ定義は有効な進め方である。

3点目として、アーキテクチャ定義は、そのシステムが達成すべき要求を、システムを構成する下位の要素（構成システムやサブシステム）に対して分解して割り当てる活動であることから、定義されたアーキテクチャに基づくことで、各構成システム・サブシステムの役割が明確となり、異なる主体がパラレルにシステム開発を進めた場合にも、全体としてシステムが整合し、目的達成が可能となる。

4点目として、アーキテクチャでは、要求、要求を満たすロジック、そのロジックを実現する手段のそれぞれを分離して定義することから、将来的な技術の進展や、環境変化に伴う要求変更が発生した際にも、どの実現手段・機能をどのように変更するかの考慮が容易になる点である。これにより継続的なシステムの発展を支援することができる。

さて、Society5.0 のアーキテクチャとは、Society5.0 という社会をシステムとして捉えたシステムアーキテクチャを指す。Society5.0 とは「サイバーとフィジカルが高度に融合されたシステムによって経済発展と社会課題の解決を両立し、人間中心の価値を実現する社会[1]」であることから、サイバーフィジカルシステム（CPS）が用いられることは自明である。また人間中心の価値を実現する上ではユーザーエクスペリエンス（UX）の最適化が必要であることから、複数の事業者が管理・運用するシステムを接続した System of Systems[2]（SoS）の形態によって実現されることも想定される。複数事業者のシステムを組み合わせ、新しい価値を生み出す上では、ビジネスエコシステム[3]の成立も必須である。Society5.0 とは産業構造のデジタルトランスフォーメーションの側面を持つことから、CPS、デジタルを活用することで既存の構造とは異なる新たなビジネスエコシステムへの変革も可能となる。そしてこれらは社会システムであることから、ルールや制度、組織も設計対象の要素となる。

Soceity5.0 実現にむけては、設計対象となるシステムの目的と合わせて、SoS、CPS の特性や、ビジネスエコシステムの在り方、社会システムの要素を踏まえたシステムアーキテクチャを定義することが必要であり、このアーキテクチャがあることで、多様なステークホルダにおける議論と合意形成を進めることが可能となる。

アーキテクチャの可視化無しに、各企業や団体、省庁などがそれぞれシステムを開発し、後から各システムの繋げ方を考える旧来のやり方で Society5.0 を目指した場合には、本来満たすべき要求に対しての過不足が不明となり、目標達成に向けた余分なコストが懸念されるだけでなく、リスクのコントロールや運用管理も難しい。まずアーキテクチャを定義し、合意することは、狙った社会を実現する確度をあげる上で重要である。

---

[1] 第 5 期科学技術基本計画より。

[2] System of System は複数の構成システムを組み合わせることで、構成システム単独では実現できない目的を達成するシステムであり、個々の構成システムの管理と運用が独立していることが特徴。

[3] 本書においては SoS を構成する各構成システム及びそのステークホルダ間における価値連鎖を指す。

Society5.0 の実現に向けて、ビジネスエコシステムの変革企図する者、SoS の設計を行う者から各構成システムの事業者等のステークホルダに向けて、ガイドラインや標準を示す場面も想定される。アーキテクチャを定義することで、構成システムへの要求、構成システム間のインターフェースへの要求が明確になることから、ガイドラインや標準の作成者は先ずアーキテクチャを定義することで、適切なガイドラインや標準化の範囲と内容を示すことができる。

1.3. DADC グリッドを使用する目的

Society5.0 を実現する為のプロジェクトは複数領域において同時並行で進むことが想定される。これらプロジェクトはそれぞれ Society5.0 の一部を設計している為、その設計結果は他プロジェクトにおいて利用されたり、また影響を与えたりするシステムとなる可能性がある。特に協調領域や共通的に使用されるシステムについては、他のプロジェクトでも活用可能な形で設計されるべきものである。つまり各プロジェクトの設計結果は、他のプロジェクトでも引用可能な形式で揃えておく必要がある。SoS を設計する者は、その設計した SoS や SoS の構成システムが広く社会で使われることを狙うのであれば、その設計結果の相互運用性を考慮しておくことが望ましいと考えられる。

そして SoS という多様なステークホルダが関係したシステムの開発と、そのシステムの長期的な運用、また運用中の構成システムの入れ替わりを考えた場合に、アーキテクチャの知見を引き継ぎ、改善していく為に統一的な名称や形式で設計結果を蓄積することが必要である。そのため、Society5.0 を目指すシステムを開発する場面において、DADC グリッドを利用し、設計結果の共通的な理解を促進することで、継続的にビジネスエコシステム、SoS 設計及び設計プロセスを改善していくことが有用と考えられる。

1.4. Soceity5.0 を実現するシステムのアーキテクチャを定義する大まかな流れ

前述したように DADC グリッドは成果物の定義であることから、設計順序について規定しているものではない。また各プロジェクトの局面によって、どの順番で着手するのかについては都度検討する必要がある。しかしながら、ある程度の大まかな流れとしては、Figure 1 のように考えることができる。

はじめに「As-Is の分析」では、アーキテクチャ設計プロジェクトが対象とする領域に影響する既存のステークホルダを洗い出し、そのビジネスエコシステム及び業務、業務を実行するシステムの観点から分析を行う。社会システムである Society5.0 の検討においては、多くの場合、そこに関わる既存業務・システム、ビジネスエコシステムが存在する。To-be を実現する際には、既存で存在する業務・システムを置換しビジネスエコシステムを変革すること、また既存システムと新たに導入するシステムの接続を

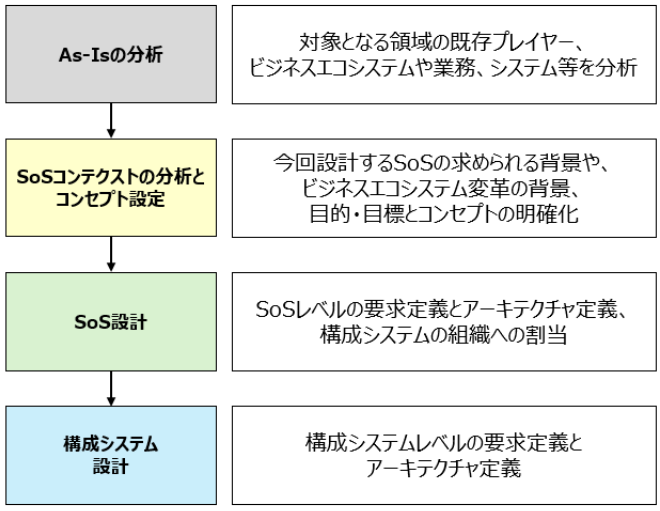

Figure 1 Society5.0 アーキテクチャ定義業務の概要フロー

行うことから、要求・制約を明確にするうえで As-Is の理解は必須となる。また、As-Is のシステムが「何故このようになっているのか」を意識することで、その背景に隠れた制約の有無を確認することも必要である。As-Is の分析が不足すると、想定外の手戻りが発生する。ここで述べる As-Is の範囲については現状稼働中のシステム・業務だけでなく、既に計画し開発中のシステム・業務を含むことに留意が必要である。

次に「SoS コンテクストの分析とコンセプト設計」では変革が求められる背景を整理した上で、目的・目標を定義し、ビジネスエコシステム及びシステムのコンセプト明確化を行う。ここでは As-Is 分析で明らかにした構造に対するステークホルダの評価や諸外国の状況、政策的な意向などを受けて、何故ビジネスエコシステムの変革や SoS 設計が必要とされているのかの背景、変化のドライバーについて整理する。それに対応し、設計対象のスコープや目指すアウトカム、そしてどの様なケーパビリティを持ってアウトカムを実現するのか、コンセプトを定める。To-Be のビジネスエコシステムは SoS によって実現されること、また SoS の実現にはビジネスエコシステムの成立が不可欠であることから、この 2 つのコンセプトは密接に関係する。

昨今の IT システムにおいて大手プラットフォーマの提供するサービスが浸透していることから分かるように、各社が事業を行うシステム環境において、各社の価値の源泉たるコア業務以外については、個々社で抱えずに複数社で共用・共通化することが可能な協調領域として切り出すことが検討できる。この協調領域の設定は、新規サービス導入時のスピードアップや、産業競争力や社会全体のコスト抑制等に影響することから、To-Be のビジネスエコシステムを検討する上でのポイントとなる。

また SoS はその運用中に技術や環境の変化に応じて構成システムの追加・改廃なども想定されることから、構成システムを入れ替え可能とする柔軟な設計としつつも、トラストを確保したアーキテクチャが求められる。

その為には自律分散的に独立した構成システムが、相互運用性とトラストを確保するレイヤ構造をもってデータスペースを介した連携を行うことをコンセプトに含めることを推奨する。また併せて構成システムの入れ替えはビジネスエコシステムの変化を伴うことから、デジタル化された契約管理、マーケットプレイスも、柔軟なアーキテクチャにおいて必要となる。SoS とは、管理と運用が独立した複数の構成システムを組み合わせることで価値を生み出すシステム形態であることから、特に、この構成システムを担う主な主体、ビジネスエコシステムのプレイヤ候補を早期に特定し、コンセプトについて合意形成を図ることが To-Be の実現において重要と考えられる。

次に「SoS 設計」について、このプロセスのゴールは、SoS がどのように要求を満たすのか、各構成システムに割り当てる機能・性能を明確にし、構成システム間のインターフェースを定義すること、そして各構成システムの担い手を割り当てることである。また同時に、このシステムに関するリスクの分析・評価を行い、各構成システムが期待する役割を適切に果たすためのガバナンスについても、このプロセスの中で検討することが必要となる。SoS 設計ではコンセプト段階には具体化されていなかったイネーブリングシステム[4]やガバナンスシステムも設計する事になるため、各役割の分担が持続可能な価値連鎖となっているか、ビジネスエコシステムが成立するのかの分析も行った上で、設計に反映する。

少し補足的な説明を加える。SoS 設計の上では、コンセプトをベースとしつつ、まず要求定義として SoS が何をどの程度、達成する必要があるのかを具体化し、定義・合意した上で、どの様なロジックと実現手段、役割分担で達成するのかをアーキテクチャ定義で定め合意する。要求定義、アーキテクチャ定義とは文字どおり「定義」することであり、意思決定を進めることである。

システム開発において最も多く発生する不具合は要求定義の不具合であるとされており、またこの不具合は工程が進む程に修正コストが高くなる特徴を持つ。要求に曖昧さを残すことは大きなリスクとなる。そのうえで SoS は前述したとおり、管理と運用の独立した数のシステムを組み合わせて実現されるものであるから、元来、不確実な要素を排除しきることが難しい。そのため、要求定義やアーキテクチャ定義の各時点での設計根拠、意思決定の根拠を明確にし、ステークホルダによる多角的な視点からの評価を受け合意しておくことで、少しでも不確実な要素を抑えると共に、想定外が発生した場合においてもそのリスクを分担できるように進めたい。

要求定義では、要求と同時に制約（制約も要求の一部に当たる）について洗い出すことが重要である。特に As-Is で存在し今後も残るシステム（つまり SoS の一部を構成することになるシステム）や、今後連携すべきシステム、国内だけでなく諸外国のシステ

[4] 目的を達成するシステムのライフサイクル各段階において支援するシステム

ム等による制約にも留意する必要がある。また Society5.0 の観点から、SoS や CPS のシステム特性（ilities[5]）に対処した設計が求められる。相互運用性とトラストついては既に触れたが、その他にもセーフティやセキュリティ等、システム特性への対応には専門性・知識が求められることから、設計に当たっては DADC や IPA で別途公開している各種ホワイトペーパー・ディスカッションペーパー等も参照し、必要に応じて要求に含めることを推奨する。またデータ連携基盤など既存で活用可能なシステム、準拠すべき標準についても考慮し、必要に応じて要求として定義する。要求とは必ずしも外部のステークホルダに起因するものではなく、プロジェクトの置かれた環境や組織のアセット等、様々なところから発生するものであることも意識し明示的に扱うことで、トレードスタディの際の根拠が明確となる。

最後に「構成システム設計」では SoS アーキテクチャ定義の結果を踏まえて、SoS を構成しているシステムの要求定義およびアーキテクチャ定義を行う。これは新たに開発する、もしくは改修する構成システム毎に行うこととなる。因みに、既存で存在する構成システムにとっては、SoS アーキテクチャ定義結果は SLA に繋がるインプットとなる。構成システムの設計にあたって、構成システムが SoS とは異なる目的、ライフサイクルを持つこと（例えば企画、設計、開発、運用、廃棄が SoS とは異なるタイミングで行われる）や、今回の設計対象である SoS 以外の SoS からも用いられること等も考慮し、構成システムとして改めて要求を定義する必要があることに留意する。

参考までに、構成システムが複数 SoS に用いられるイメージを Figure 2 に示す。この例では SoS I で用いる為に開発した構成システム C が、SoS II においても利用されている図としている。また **Figure 3** では同じシステム構成について、ふるまいとして表現したものを示す。SoS I に属する構成システム A、B、そして SoS II に属する構成システム D のそれぞれと構成システム C が接続している。

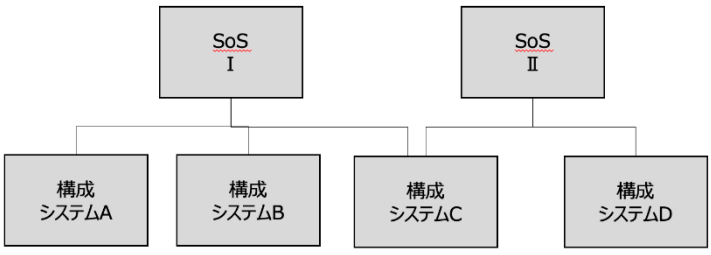

Figure 2 構成システムと SoS の関係

[5] システム特性の英単語語尾が”ility“となるものが多いことから、ilities と呼ばれている

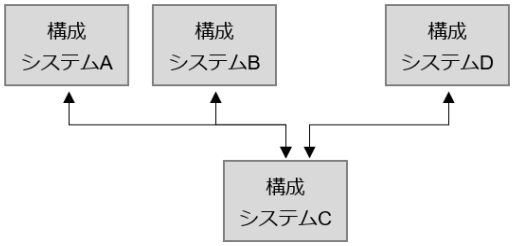

Figure 3 複数の SoS で使用される構成システムふるまい

これらの図は例として構成システム C が、２つの SoS において共通的に機能を提供しているイメージであり、両方の SoS からの要求を受けることになる。もちろん単一 SoS の中において共通機能を提供する構成システムもあるが、どのように機能及び構成システムの配置を設計するかを検討するうえで、前述した協調領域の考え方やレイヤ構成を意識する必要がある。

2. DADC グリッドの概要

2.1. DADC グリッドの基本構成

DADC グリッドでは、Society5.0 を実現するシステムのアーキテクチャ記述を行ううえで、設計結果として各ビューで表現すべき事項を示している。Table 1 に、グリッドの大まかな構成を示す。

行方向（表側）では、大きく As-Is と To-Be に分けしており、それぞれの内訳として、「SoS のコンテクスト・ビジョン」「SoS アーキテクチャ」「構成システムアーキテクチャ」としてシステムを 3 階層とし、定義すべきものを管理する。この 3 階層は基本的に「SoS コンテクスト・ビジョン」から「構成システムアーキテクチャ」に向かって段階的に詳細化する流れを取っている。As-Is では分析結果を記載し、To-Be では設計結果を記載する。SoS コンテクスト・ビジョンでは主に SoS と外部との関係を示し、SoS アーキテクチャでは SoS の内部の要素とその関係を定義する。構成システムアーキテクチャでは、SoS を構成するシステムのうち、新たに開発対象となるものの内部の要素とその関係性を定義する。基本的に合計 6 行で構成しているが、設計する構成システムが複数ある場合には、最終行を設計対象の構成システムの数に合わせて記述する。

列方向（表頭）では、大きく「モチベーション・要求」「ロジック」「実現手段」「ロジックの実現手段への割当」「組織」「価値連鎖」「リスク分析」に分け、それぞれに内訳となる項目を設定している。「モチベーション・要求」とは Why/What であり、どういった背景のもとに何を実現するのかを整理・定義する。

Table 1　DADC グリッドの構成概要

		モチベーション・要求	ロジック	実現手段	ロジックの実現手段への割当	組織	価値連鎖	リスク分析	設計根拠	ロードマップ
As-Is	SoS コンテクスト・ビジョン									
	SoS アーキテクチャ									
	構成システムアーキテクチャ									
To-Be	SoS コンテクスト・ビジョン									
	SoS アーキテクチャ									
	構成システムアーキテクチャ									

「ロジック」は What をどのように実現するかの How の論理・機能であり、「実現手段」はロジックを担う物理的な How としてハードウェア・ソフトウェアや人、ルール等の定義である。How をロジックと実現手段に分割することは機能と物理の分離とも呼ぶが、これを明確に分けて管理することは解空間を広げることや、検証性の確保、技術変化への対応の容易性を上げるうえでも重要である。「組織」は、その実現手段を担う具体的な組織の定義である。「価値連鎖」では、ビジネスエコシステムにおける価値の連鎖を表現し、「リスク分析」では設計対象の生み出すリスクについて分析する。また、その他に「設計根拠」と「ロードマップ」を配置している。

基本的には左から右に向かって具体化していく（設計解が絞られていく）構成としているが、各列に対する要求や制約も存在することから、単純に左から決めていけるものではなく、ウォーターフォール的に進めるものでもない。まず 1 度、手元の情報を元におおまかに記載してみることで、何の情報が不足しているのか、どこが論点になるのかを明らかにする使い方もある。重要なことは各セルの要素が相互に関係することを意識して、前提、目的から実現手段、組織までを一貫して齟齬なく繋げることである。これによって、どのように目的達成を図っているのかを示し、トレーサビリティを確保することができる。

分析、設計根拠については、各セルの設計の都度、必要に応じて使用する内容となっている。

ここでは構成概要を説明するためにセル内の記載を省いているが、グリッド自体ではこの行×列の関係で記述すべき項目を記載している。また、行列の設定の都合上、定義が不要となるセルも存在するので、詳細はグリッド本体も参照いただきたい

2.2.　グリッドを構成する要素の関係性

グリッドを構成する主なコンテンツの依存関係の概要について Figure4 に示す。ここでは SoS コンテクスト・ビジョンレベル及び SoS アーキテクチャレベルまでの成果物の主要な箇所を表現した。この図中の箱はグリッドを構成する要素であり、一つの箱の中にはグリッドのセルの一部分であったり、複数のセルに対応したりする要素を含んでいる。ここでは理解の容易性の観点から外観の把握を重視し、セルとの具体的な対応を示していない為、詳細はセルの説明も参照いただきたい。矢印で接続している箇所は矢印の発出元が根拠となるものであり、矢印の無い線で接続しているものは相互に影響する箇所である。矢印の根元にあるものが不明瞭である場合には、後々前提が崩れて手戻りが発生する可能性が高くなることに留意が必要である。

これら全ての箱の整合をとることで、システムを実現するロジックが明確とかなり、そのアーキテクチャについての議論が容易となる。また設計の結果、ピンクで塗りつぶしの箇所が構成システムに対する要求として、構成システムの担当者に対して提供する情報となる。

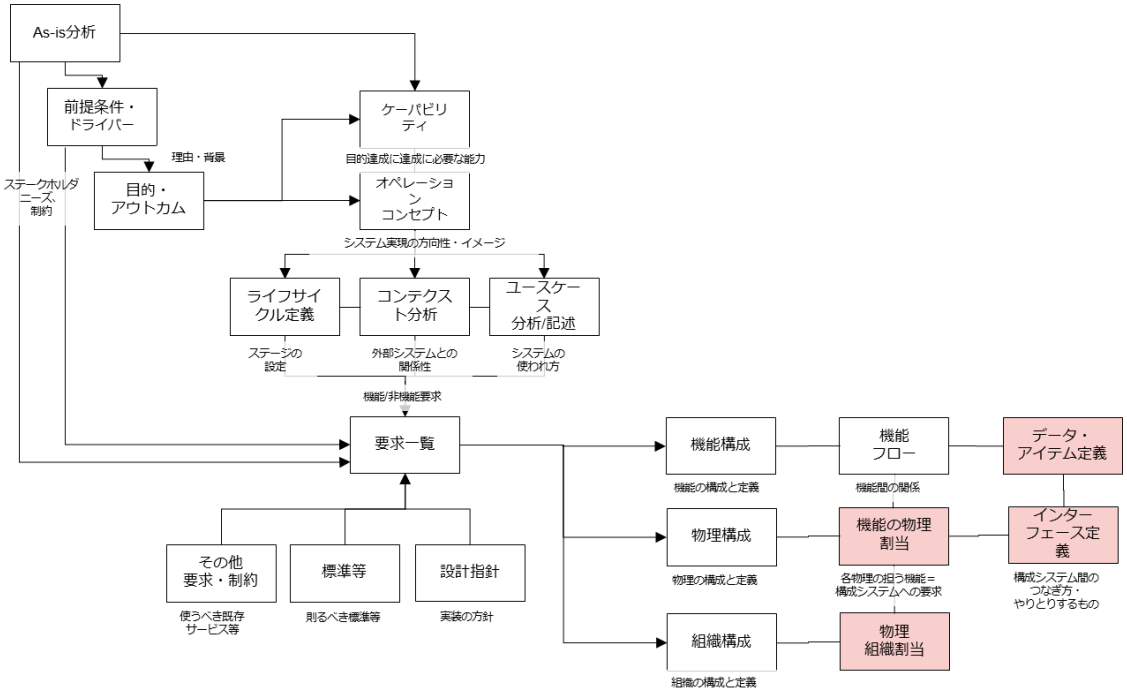

Figure 4　SoS レベルの成果物依存関係

2.3.　DADC グリッドの使用方法

DADC で整備しているメソッドは、システムズエンジニアリングの手法を参考にテーラーリングしたものとなっている。1.4 で示した大まかな流れを前提に、アーキテクチャ定義の進め方・プロセスについては ISO/IEC/IEEE15288 等を参考とし、それらのプロセスを進めていく中で、グリッドで示されたアーキテクチャ成果物を生成していくことを想定している。

なお検証および妥当性確認（V&V）はグリッドの列に明示的に示されていないが、それぞれのセルの成果物を生成する過程で V&V を実施することを想定している。グリッドで表現した成果物は設計の根拠や妥当性を示す上で必要なものであり、各セルで表現したものを用いて有識者やステークホルダとの議論を行い、設計の確からしさを評価、合意する。また、この成果物を元に、各構成システムに対する要求事項を生成することで、SoS 全体が整合をもって開発・維持することが可能となる。

セル内に記載している成果物の種別は最低限必要なものを列挙しているものであり、対象システムによってアーキテクチャ定義に必要な成果物の種類、量は異なる。

3. DADC グリッドを構成するセル

3.1. セルの特定

グリッドの各行・列には ID を振り、行 ID＋列 ID でセルを特定できるようにした。行 ID については Table 2 に、列 ID については Table 3 に示す。また併せて、2 章で説明した列項目に紐づく小項目の内訳もここで示す。

Table 2 行項目と行 ID

行項目	行小項目	ID
As-Is	SoS コンテクスト	ACn
	SoS アーキテクチャ	ASS
	構成システムアーキテクチャ	ACS
To-Be	SoS コンテクスト・ビジョン	TCn
	SoS アーキテクチャ	TSS
	構成システムアーキテクチャ	TCS

Table 3 列項目と列 ID

列項目	列小項目	ID
モチベーション・要求	前提条件	Pre
	対象範囲	Scp
	コンテクストとの関係	Cnt
	要求	Req
	コンセプト	Cnc
ロジック	ロジックの定義と構成	LT
	ロジックのフロー	LF
	データ・アイテム	Dt
実現手段	実現手段（SW/HW/人）	PE
	実現手段（ルール）	Rl
ロジックの実現手段の割当	アロケーションの構成	Alc
	実現手段間の関係性	SD
	実現手段間のインターフェース	IF
組織	実施組織	Org
分析	価値連鎖分析	VC

列項目	列小項目	ID
	リスク分析	RA
他	設計根拠（実際には各 View 毎に存在）	AR
	ロードマップ	RM

3.2. 各セルの記載内容

グリッドの各セルで記載する具体的な内容について説明する。前述のように、グリッドの行方向で As-Is と To-Be に大きく分けているが、それぞれで記述する事項は、分析結果か設計結果かの違いはあるものの、基本的には同様の記述となる。そのため、ここでは As-Is 及び To-Be で共通の内容として、SoS コンテクスト・ビジョン、SoS アーキテクチャ、構成システムアーキテクチャの 3 行と各列を掛け合わせたセルの内容について記載する。そのため、行 ID の T もしくは A に当たる箇所について、ここでは X として表記している。ただし、To-Be でしか記載しない（出来ない）項目については、X ではなく T で表記している。また分析と設計の違いによって生じる As-Is 記述における留意点については、次節で説明する。

グリッド及び本解説書内で各セルの表記方法についてもコメントしているが、表記方法は例示である。例えば MBSE 導入済みである場合等は別な表現の方が妥当であることもあり、現時点でグリッドにおいては表記方法やモデリング言語は指定していない。重視している点は明記すべき事項である。

3.2.1. SoS コンテクストレベルの前提条件（TCn-Pre）

このセルでは、SoS アーキテクチャ検討の前提となる情報、変化のドライバーについて取りまとめる。変化のドライバーについては、主には①現状に対するステークホルダの評価、②諸外国における当該領域の状況、③政策的な意向、④その他関連する要求などから記載する。それぞれについて説明する。

① As-Is 分析結果をもとにインタビューその他で収集した As-is に対するステークホルダの評価を整理、分析する。SoS や構成システムの観点だけでなく、ビジネスエコシステムの観点からも分析する。

② 諸外国での該当領域における動向を記述する。先行するシステムがある場合には、そのアーキテクチャの可視化・分析を行った上で特徴についてテキスト化する。先行するシステムが必ずしも今回目指す姿や日本のコンテクストに合致するとも限らないが、優れた点は参考にすべきである。複数の国のアーキテクチャを比較する必要がある場合には、【ASS-Org】,【ASS-VC】,【ASS-SD】,【ASS-LT】,【ASS-LF】相当の分析結果を並べることで統一的な観点でその特徴を分析することができる。

③ 政府や他組織において関連する取り組みがなされている場合には、動向・意向を収集する。
④ 関連要求・その他ステークホルダ等からのニーズを記述する。例えば、先行するプロジェクトで扱う他分野との連携などにより、当該の SoS に対する要求や制約がある場合がある。

変化のドライバーはテキスト、表などで取りまとめることを想定する。

3.2.2. SoS 対象範囲（XCn-Scp）

前提条件の分析を総合的に踏まえて、対象となる領域のビジネスエコシステムをどのように変革することを狙うかを念頭に、SoS が①何を目的に、②どういったアウトカムを③いつ頃出すものかを定義する。

① 設計対象となる SoS の目的やミッションを記載する。
② 目指すアウトカムとして、変化のドライバーに対応した形でアウトカムを記述する。目的・ミッションとの整合についても確認する。表もしくはダイアグラムでの記述を想定、アウトカムは可能な限り定量的に表現する。
③ ターゲットとなる SoS の実現時期を、年単位程度のイメージで記載する。段階的な実装の場合には、そのステップを併せて記載する。

3.2.3. SoS とコンテクストとの関係性（XCn- Cnt）

このセルでは SoS と外部システムとの関係について記載することで、対象とする SoS の置かれた環境と SoS の境界を示す。記載内容は、ライフサイクルステージ定義、コンテクスト分析、利用シーン、ステークホルダ分析とする。

外部システムとの関係性を分析するうえで、ライフサイクルステージ定義が重要となる。これはステージ毎に外部環境との関係が異なることで、各ステージから異なる要求が発生することに起因する（外部環境が異なる単位をステージとして考える）。そして 1 つのステージで抽出した要求は他のステージへも影響する為、システムを設計する上では、まずライフサイクルステージを定義した上で、各ステージにおける要求を分析する必要がある。SoS のシステムライフサイクルの特徴として、運用中に構成システムの追加・改廃などがあるので、留意する必要がある。ダイアグラムや表での表現を想定。

併せて、ライフサイクルステージ毎のコンテクスト分析を示す。SoS の範囲は目的達成に必要となるシステム群であり、その外側には SoS につながる外部システム（ステークホルダを含む）、外部環境がある。外部システムに何があり、SoS とどの様な関係を持つのかについて、ここでは概略を示す。表記としてはコンテクスト図や表形式での記述を想定する。

また利用シーンとして、SoS が適用される領域のイメージを共有するために、どう

いった場面でシステムが使われるのか示す。例えばドローンの効率的な管制を行う SoS を考えた時に、その利用シーンとして、物流での利用やインフラ点検における利用などを考える。どういった利用シーンを想定するのか、表やブロック図などでの記載を行う。

ステークホルダ分析はプロジェクトマネジメント的な要素でもあるが、ここではコンテクスト分析や利用シーンの分析を踏まえて、どういったステークホルダがどういった立場で関わるのか分析を行う。表や各種分析手法などを想定。

3.2.4. SoS 要求（XCn- Req）

SoS とコンテクストとの関係を踏まえ、SoS が目的を達成するために、どういった能力が求められるのか、目的を最上位としてライフサイクルを考慮したケーパビリティを示す。ツリー構造や表での記載を想定。

3.2.5. SoS ビジョン・コンセプト（XCn- Cnc）

SoS のコンセプトとして、どういったビジネスエコシステム、SoS を実現しようとしているのか、コンセプトイメージを示す。システム図や CVCA,イラスト等の記述を想定する。コンセプト設定にあたっては、1.4 に記載の推奨事項も参照する。

3.2.6. ロジックの定義と構成（XCn- LT）

ケーパビリティが発揮される際のオペレーションをユースケース程度の粒度に分解することで、求められる能力を具体化する。オペレーションの記載は実装を選ばない形で記載をする。

ケーパビリティをトップに置いたツリー構造で記載。

3.2.7. 組織定義（TCn- Org）

求められるケーパビリティを保有していることを期待する組織について検討し、表等で整理する。

3.2.8. SoS とコンテクストとの関係性詳細　（XSS- Cnt）

オペレーションを細分化し、システムの外部から見たふるまいを具体化する。ユースケース分析・ユースケース記述などの記載を想定する。記述にあたっては実現手段に捕らわれない表記を意識するものの、制約において使用が決まっている As-Is システム等がある場合には明示的に記載し、前提や範囲で記載した内容と整合を取る。またユースケース記述では人間中心の UX を意識する。3.2.3 で記載したコンテクストから更に分析を詳細化し、外部から見たふるまいと整合している必要がある。

3.2.9. SoS アーキテクチャレベルの要求（XSS- Req）

SoS に求められる要求を取りまとめる。コンテクスト分析、ユースケース記述、前提から抽出した要求を一覧化する。抽出する要求は機能だけでなく、対象の SoS の特性を元に考慮すべきシステム特性（ilities）についても踏まえ非機能要求についてもまとめる。Ilities の対応についてはセーフティ、セキュリティ、トラスト等、専門的知識が求められることから、有識者の見解の反映し、別途 IPA や DADC が発出しているドキュメント類も参考とする。
また各ライフサイクルステージにおいて則るべき標準などがある場合には要求として記載する。併せて、後述するリスク分析の結果から抽出された要求も記載する。

3.2.10. SoS アーキテクチャレベルのコンセプト（XSS- Cnc）

SoS の設計方針を定義する。設計方針は DADC が別途定義している「Society5.0 に資するシステム設計原則」なども参照し、人間中心の CPS を実現するためにアーキテクチャ設計において統制すべき点を明確にする。設計方針は、設計ステージに対する要求となる。

3.2.11. SoS 機能定義と構成（XSS- LT）

要求を元に SoS が持つべき機能の構成を定義する。実現手段に依存しない機能の形で記述する。

デジタル完結に十分な機能となっているのか、データドリブンを実現する機能となっているのか、SoS を適切に保つガバナンスの機能が考慮されているのか等の観点から確認する。ツリー図で階層化し記載する。

3.2.12. SoS 機能フロー（XSS- LF）

ロジックの構成で定義した機能について、機能間の関係性をフローで整理する。併せて、機能間を流れるデータやアイテムについて記載する。機能間の関係性を明確化することは、機能の実現手段への割り当て時の参考となり、また実現手段間の関係性を定義する上で必要となる情報である。Enhanced Functional Flow Block Diagram（EFFBD）やアクティビティ図で表現する。

3.2.13. SoS データ・アイテム（XSS- Dt）

ロジックのフローにおいて挙げたデータやアイテムについて、具体的な内容・構造等を定義する。表での記載を想定。

3.2.14. SoS 実現手段（HW/SW/人）（XSS- PE）

SoS の実現手段の定義と構成を示す。前述のとおり、As-Is のシステムが含まれる

場合が中心であり、要求として実現手段自体が規定されている場合も存在する。また、対象業務分野の As-Is システムではなく、他分野で先行して開発され、共通基盤としての利用が期待されているプラットフォームやモジュールを活用する場合もある。SoS レベルで検討する構成システムにおいては、その構成システムを実現する一部として人の活動を含む場合がある（ただし、この段階では構成システムの中にある下位のサブシステムについては基本的にブラックボックスである）。SoS をトップに置いたツリー図で表現する。

3.2.15. SoS 実現手段（ルール）（XSS- Rl）

SoS の実現手段のうち、ハードウェアやソフトウェア、人等ではなく、ルール、規則などにより機能を実現する場合が想定される。その場合には、ルールや規則を実現手段として定義する。

3.2.16. SoS アロケーションの構成（XSS- Alc）

ロジックの構成で定義した機能を実現手段に割り当てる。併せて要求として定義されている性能についても割当てを行う。適切にモジュール化され、疎結合かつ相互運用性を意識した構成となっているのか、非機能要求を満たせる構成になっているのか等の観点から確認する。ブロック図や表等で示す。

3.2.17. SoS 実現手段間の関係性（XSS- SD）

実現手段に割り当てられて機能と、ロジックのフローで定義した機能間の関係性を元に実現手段間の関係性を記載する。実現手段を箱で示し、その実現手段が持つ機能を箱の中に内包する形で記載し、構成システム間をつなぐ。システム図、ブロック図の様な表現を取る。

3.2.18. SoS 実現手段間のインターフェース（XSS- IF）

実現手段の関係性で定義した構成システム間の関係性において発生するデータ・アイテムのやり取りを実現する手段について、インターフェースの定義を記載する。

3.2.19. SoS 実施組織定義（XSS- Org）

各構成システムを担う主体について記載する。競争・協調の振り分けが産業競争力や持続可能性を考慮しているか、またシステム機能が維持されるためのガバナンスが効いているか等の観点で確認。協調部分は具体的にどの組織が行うのかなどもこの段階で示される。表やブロック図での記載。

3.2.20. SoS 価値連鎖分析（XSS- VC）

金銭やサービスの授受を中心とし組織間の価値連鎖、ビジネスエコシステムがどのように実現されるかの分析。一方的に支払うだけの組織が存在するなど持続可能でない設計なっていないか、価値連鎖におけるリスクや、産業として脆弱な箇所がないかの分析。懸念が発生した場合には機能追加や組織見直しを検討。分析手法として Customer Value Chain Analysis（CVCA）や SVN（Stakeholder Value Network）等を想定。

3.2.21. SoS リスク分析（XSS- RA）

コンテクスト分析・ロジック設計・実現手段設計等、各段階で行ったリスク分析について、想定したリスクと評価結果、対応方針の一覧を示す。対応するものは機能、構成システムやルールに反映済みとなっている。前述のとおりグリッド上のセルの配置は順番を示すものではないことに留意し、各設計の都度、段階的にリスクを分析する。社会実装された SoS が機能喪失した場合に社会へ与える影響もリスクとして想定する。SoS はその創発的な特性から不確実性が高いこと、また構成システムの管理・運用主体が複数存在する中で、設計した SoS をどのように維持するのか、リスクへの対応としてはガバナンスも併せて検討する必要がある。検討にあたっては別途 DADC が発出している「SoS-CPS 安全設計ディスカッションペーパー」なども参照する。

表現方法としては FTA、FMEA、STAMP/STPA などのリスク分析手法を用いた結果と、結果に対する評価・対応が示された表などを用いる。

3.2.22. SoS 設計根拠（XSS- AR）

各設計の段階では、複数の案、設計オプションが生成されることから、最終的な設計結果がなぜ採用されたのかの根拠を残す。アカウンタビリティ確保のためにも重要なものであり、生成された案と比較軸、比較結果を示す。

3.2.23. SoS ロードマップ（XSS- RM）

SoS 実装に向けた構成システム開発や改修などを含めたロードマップを示す。

3.2.24. 構成システムアーキテクチャ 前提（XCS- Pre）

SoS アーキテクチャで定義した事項に基づき、SoS から構成システムに対して求められている要求事項を表やブロック図などで記載する。

3.2.25. 構成システムアーキテクチャ スコープ（XCS-Scp）

構成システムは SoS とは異なる独自の目的をもつ。新たに開発する構成システム

も他の SoS において使われることも想定し、その目的、アウトカム、実現時期を明示する。文章での記載を想定。

3.2.26. 構成システムアーキテクチャと外部システムの関係性（XCS-Cnt）

前提条件を踏まえ、構成システムのライフサイクル定義、ライフサイクル毎のコンテクスト分析、ユースケース図・ユースケース記述を記載する。他の SoS からの使われ方も含めて、構成システムと外部システムの関係性を表現する。

3.2.27. 構成システムアーキテクチャ その他のセル

構成システムのその他セルで記述すべき内容は基本的に XSS の各セルで示したものと同等の内容について、構成システムを対象として記載する。

3.3. As-Is に関する留意点

SoS の As-Is システム・業務を分析する上では、自身の携わっていないシステムも含めた全体を対象とすることから、各システムに十分な知識がない状態で外部から分析する場面が想定される。つまり外部から見て分かる実施組織や実現手段などから遡って論理的な機能の分析に進むことが中心と考えられる。As-Is は可能な限り分析することで To-Be 設計のインプットに繋げるものの、最小限として、SoS コンテクスト・ビジョンの対象範囲（ACn-Scp）、コンテクストとの関係（ACn-Cnt）、SoS アーキテクチャの実現手段間の関係性（ASS-SD）、実現手段間のインターフェース（ASS-IF）、データ・アイテム（ASS-Dt）、実施組織（ASS-Org）、価値連鎖分析（ASS-VC）、を中心的に記載することで As-Is の把握を図ることを想定している。特に、ASS-VC はビジネスエコシステムの分析において、ASS-SD は既存構成システムの配置の分析において、重要なビューとなる。分析にあたっては As-Is ステークホルダへのヒアリング等により、分析の確からしさを向上させることも必要である。As-Is 分析では、既存の課題を発見するだけでなく、複数の事業体が活動している競争領域において各事業体が持つ構成システムの機能を分析することで、競争の必要がない業務（ノンコア業務）にあたる機能を抽出し,プラットフォームとして共通機能化を検討する材料にもなる。

4. おわりに

DADC では、Society5.0 実現にむけて産学官の幅広い方々と協同しアーキテクチャを定義するためのメソッド整備を進めている。DADC アーキテクチャ成果物グリッドでは、まずどういったアーキテクチャ成果物が必要かを示した。

今後、Society5.0 に資するシステム設計の実践を重ねる中でメソッドのアップデートを行う予定であり、Society5.0 を指向する皆様には、ぜひグリッド及び本解説書を活用しアーキテクティングを進めていただくと共に、フィードバックを頂きたい。

- 本書の作成に携わったメンバー（五十音順）

  大野嘉子
  黒飛岳志
  齋藤毅
  清水勝人