AI 時代のルール（法・標準）とソフトウェアエンジニアリングに関する共同調査研究報告書

2026 年 3 月

独立行政法人情報処理推進機構
Information-technology Promotion Agency, Japan

京都大学大学院法学研究科法政策共同研究センター

はじめに

本書は独立行政法人情報処理推進機構（IPA）と京都大学大学院法学研究科法政策共同研究センター（KILAP）が実施した共同研究「AI 時代のルール（法・標準）とソフトウェアエンジニアリングに関する共同調査研究（2025 年 5 月～2026 年 3 月）」の研究結果をまとめた報告書である。

本共同研究では、AI の利活用が急速に進展する中で、法令・ガイドライン・標準等の「ルール」と、ソフトウェアエンジニアリング（設計・開発・運用・保守）の実務をどのように接続し、説明責任を伴う形で実装・運用していくべきかを整理した。

本書は、政策・制度設計に関わる実務者、標準化やガバナンスに携わる担当者、ならびに AI を含むシステムの開発・運用責任者が、共通の前提と語彙をもって議論できることを目指している。なお、ルールには社内ルールも含んでおり、本書の内容は企業においても活用可能である。

報告書は、本書および別冊（付属資料）の計 4 点で構成される。本書は本体（第 1～3 章）で本共同研究の背景・目的・枠組み（LE4SDS）を概説し、まず本体のみで研究の概要を把握できる構成としている。付録 A・付録 B は、関連する最新動向および手法の概観を整理したものであり、本体で述べた枠組みの理解を補完するため、併せて通読されたい。具体的な分析・ケース適用・適合性評価の設計および関連資料は、別冊に詳述する。

① 共同研究報告書（本書）：共同研究の背景や目的、全体概要、関連動向を取りまとめたもの。
② 別冊 1　ルールと技術の統合的運用としての「法」：LE4SDS に関して国内外の状況を調査すると共に、技術的及び社会的なフィージビリティも踏まえながら、今後の我が国がとるべき戦略を提言。
③ 別冊 2　LE4SDS における構造可視化手法の検討：具体的なケースを対象に、本書で提唱するソフトウェアエンジニアリングの方法論に基づきモデル化を試行した検討結果。
④ 別冊 3　AI 分野における適合性評価について：LE4SDS における法・標準・適合性評価の関係を整理するための素材を提供。AI 分野における国際標準化と適合性評価の動向を概観するとともに、AISI 事業実証 WG／適合性評価 SWG の活動について概説した。

目次

第 1 章　本研究の背景

― AI 時代におけるルール（法・標準）とソフトウェアの再統合の必要性 ―

1.1 Software-Defined Society への構造転換

現代社会は、単なるデジタル化の段階を超え、社会機能そのものがソフトウェアによって定義される「**Software-Defined Society（ソフトウェア・ディファインド・ソサエティ）**」へと移行しつつある。行政手続、金融取引、交通制御、医療診断、教育支援など、社会の基幹的機能の多くがソフトウェアによって実装され、データとアルゴリズムを通じて継続的に更新される構造が一般化している。特に人工知能（AI）の急速な普及は、意思決定の自動化と高度化を加速させ、社会の構造そのものを再定義しつつある。

この変化の本質は、「**更新可能性**」にある。従来の制度や製品は、設計・施行・出荷の時点で固定されることを前提としていた。しかし現在のソフトウェアおよび AI は、継続的なアップデートを前提とし、稼働後も機能や特性が変化する。社会が「常に更新される存在」となった以上、それを支えるルール（法律・標準・規制）もまた、更新可能性を前提とした構造へ転換しなければならない。

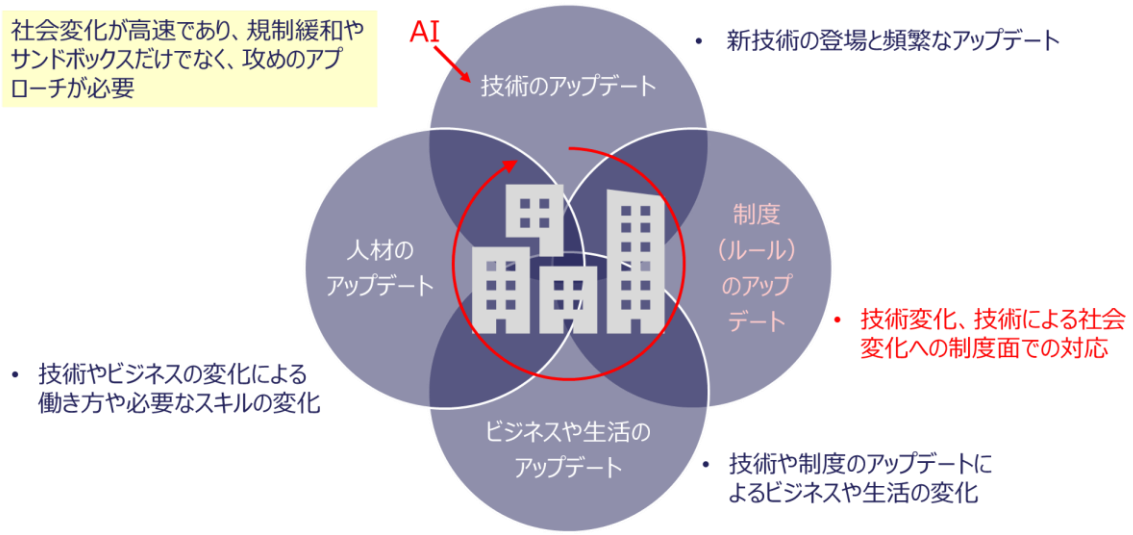


図 1 アップデートし続ける時代におけるルールの重要性

ルールは社会の制約条件であると同時に、社会の OS（Operating System）である。技術革新を阻害せず、かつ社会的価値を守るためには、ルールが技術進化と同期可能であることが不可欠である。しかし現実には、技術と制度の間に深刻な構造的乖離が生じている。

この変化は三つの意味で従来と質的に異なる。

1. **更新頻度の高速化**：システムは継続的にアップデートされる。
2. **複雑性の増大**：複数システムが相互作用することで予測困難性が増す。
3. **意思決定の自動化**：法的・倫理的影響を伴う判断が機械的に実行される。

社会が「常にアップデートされる存在」になったにもかかわらず、それを支えるルール体系は、依然として固定的・静的構造に依拠している。この非対称性が、AI 時代の制度的根本課題である。

さらに、Software-Defined Society の特徴は、あらゆるものがモデル化されるという特徴がある。建物や設備、自動車などは図面でモデリングされるだけでなく、その属性データもモデリングされている。そして、センサーデータなども用いたリアルタイム・デジタルツインの実現、さらには AI との組み合わせが近づいてきている。

こうした中で社会の OS ともいえる法律や制度のモデル化、ソフトウェア化も避けて通れなくなってきている。

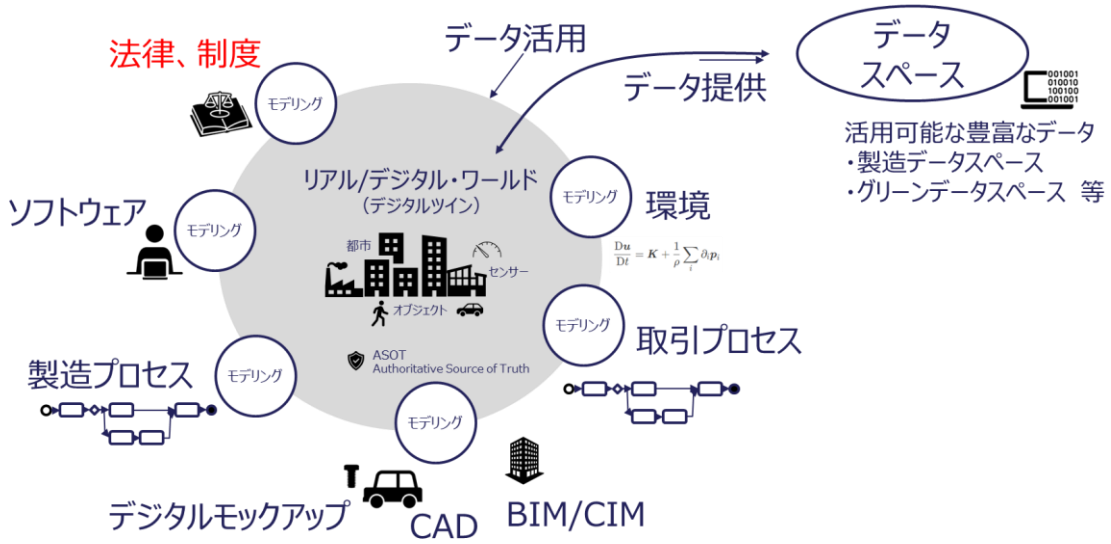


図 2 あらゆるものがモデル化される

1.2 「法」と「技術」のクロックサイクルの乖離とその影響

現在の最大の課題は、技術進化と法制度更新のクロックサイクルの圧倒的な差異である。ソフトウェア開発はアジャイル開発や CI/CD（継続的インテグレーション／継続的デリバリー：変更を頻繁に統合・テストし、継続的にリリースへ反映する開発・運用手法）の普及により、週単位あるいは日単位で機能改善が行われる。一方、法律の制定・改正には通常年単位の検討が必要であり、国際標準の策定にも複数年を要する。

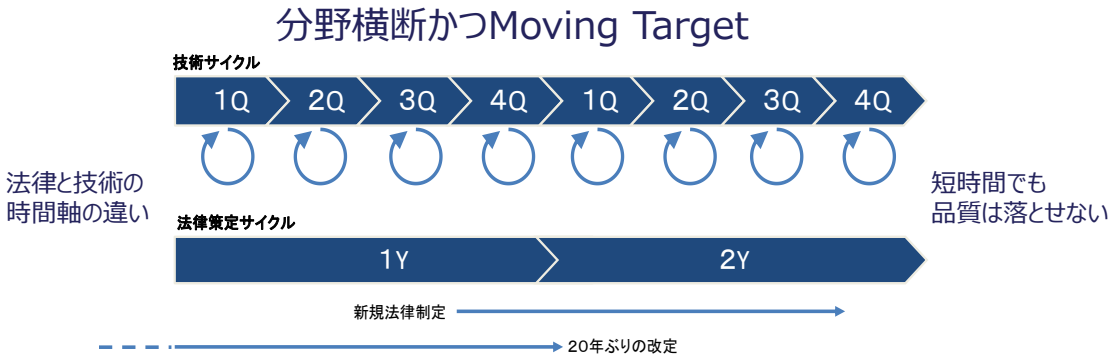


図 3 クロックサイクルの乖離

この速度差は偶発的な問題ではない。制度設計の前提そのものが、工業社会型の安定環境を想定しているためである。

この結果、社会実装の現場では次の三つの現象が発生している。

- ルールが存在しない「空白地帯」
- 想定外技術に対する過剰規制または無規制
- 解釈に依存するグレーゾーンの拡大

日本においては、この傾向が「萎縮効果（Chilling Effect）」として顕在化している。一方で、欧米諸国では、その空白地帯をブルーオーシャンととらえ、実証を通じて規範形成を先行させるアプローチが採用されている。技術競争力のみならず、「ルールを設計し、活用する能力」が国際競争力を左右する局面に入っている。

日本においてもアジャイルガバナンス（運用データや状況変化を踏まえ、制度・運用・統制を継続的に見直すガバナンス）の理念は提示されている。しかし、その理念を具体的な制度設計手法や実装モデルへ落とし込む体系は確立されていない。

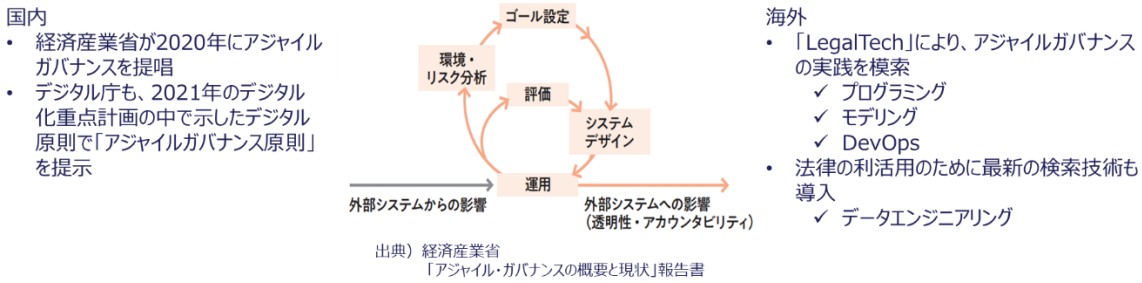


図 4 アジャイルガバナンスの動向

したがって、国内外の **LegalTech**（法務・規制対応などを IT で支援する技術・サービスの総称）や **Rules as Code**（法令・制度のルールを機械可読に記述し、実装・検証・提

供に活用する取り組み）、**標準化動向**などを体系的に整理し、現状の到達点と限界を明らかにする必要がある。

さらに、競争力の源泉が、技術そのものではなく、**技術とルールを統合的に設計する能力**へと移行しつつある現状に早急に対応していく必要がある。

1.3 パッチワーク化した法体系と法益ベース再設計の必要性

現行の法体系の多くは、長年にわたる部分改正の積み重ねにより構築されてきた。その結果、条文構造は複雑化し、技術前提が更新されないまま維持されている場合も少なくない。いわば制度の「スパゲッティ化」が進行している。

AI 時代においては、条文の表層的修正では不十分である。必要なのは、各制度が守ろうとしている本質的価値、すなわち法益に立ち返ることである。安全性、公平性、透明性、プライバシー、説明可能性といった法益を明確化し、その達成水準をベンチマークとして定義する。そして、その達成方法については、技術革新を許容する柔軟な構造とする。これは、条文中心主義から目的志向型設計への転換を意味する。

これは単なる立法技術の問題ではない。**社会の進化速度に耐えうる制度設計哲学への転換**である。

1.4 法律とソフトウェアの構造的な類似性とリーガルエンジニアリング

法律とソフトウェアは、ともに目的達成のための論理体系である。両者は、定義、条件分岐、手続き、モジュール性、改訂履歴管理といった構造的特徴を共有する。この類似性に着目すれば、ソフトウェアエンジニアリングの方法論を法制度設計に適用することが理論的に可能となる。

この構造的類似性は偶然ではない。両者は「**目的達成のための規範体系**」である。

表 1 ルールとソフトウェアの類似性

	制度（ルール）	ソフトウェア
表現形式	文書	コードやモデル
内容	構造化（章、条文、項目化）	構造化（機能設計）
可視化、理解容易性	必要（専門性が必要） コンプライアンスを実現	必要（専門性が必要なものもある） 障害などの検証が可能
単語定義	曖昧性がある場合がある	一意性を求める

辞書性	正当な辞書になりえる	必要だが、独自策定することが多い
モジュール性	手続きなどで類似例を参照することがある	モジュール化は一般的
雛形の活用	類似例を参照することがある	パッケージや外部サービスの業務雛形を活用
シミュレーション	仮説に基づく計算	パラメータなどを使って可能
実装までのリードタイム	長い	短い
運用からのフィードバック	報告などに基づく	DevOps

モデリング、要求工学、テスト駆動設計、シミュレーション、DevOps（開発と運用を一体で改善し、継続的なリリースと安定運用を両立する考え方）、トレーサビリティ設計といった技術は、法律の可視化、事前検証、事後評価、継続的改善を可能にする。法律のデジタルツイン化や、ベンチマークに基づく評価設計も視野に入る。

しかし、法をコードで表現する単純な「Law as Code」の推進は十分ではない。抽象概念のコード化の困難性、未知リスクの網羅不可能性、価値判断の固定化リスク、そして法の硬直化の問題が存在する。これらの限界を踏まえたうえで、より高度な枠組みを構築する必要がある。

この応用を体系化する概念が、**リーガルエンジニアリング**である。

1.5 AI 時代における適合性評価の構造的課題

ルールを考える上では、法律だけでなく技術や運用の標準も考える必要がある。

AI やソフトウェアは出荷後も更新される。つまりは、AI やソフトウェアを組み込んだ製品が更新を継続的に行うということである。従来のプロダクト認証は「点」での評価を前提としており、動的更新を前提とするシステムには適合しない。一方、マネジメント認証のみでは製品の安全性やリスク管理を十分に保証できない。

このため、組織の継続改善能力と製品の動的安全性を統合的に評価する新たな認証モデルが必要となる。これが**ジョイントサーティフィケーション**（組織のマネジメント認証とプロダクト認証を一体で行い、更新を前提に評価・証跡を継続的に管理する考え方）の問題意識である。**適合性評価**（規格・基準への適合を、試験・監査等により確認・評価する枠組み）は、単なる技術仕様や組織のマネジメントシステムの確認ではなく、「何を証拠として収集し、どのように評価し、どのように更新するか」を設計するプロセスへと進化しなければならない。

これは、「何を作ったか」ではなく「どう作り、どう更新し続けるか」を認証する仕組みである。

第 2 章　本研究の目的

― LE4SDS を通じた法・標準・実装の統合的設計モデルの提示 ―

本研究は、AI 時代におけるルール（法・標準）とソフトウェアエンジニアリングを統合する体系的枠組みを提示することを目的とする。その核心は、理論の提示にとどまらず、実装可能性と評価可能性を備えた方法論を構築することにある。

本研究の成果は、本体（第 1〜3 章）／付録（付録 A・付録 B）／別冊（別冊 1〜3）に分けて整理する。本体は研究の背景・狙い・用語定義・枠組み（LE4SDS）を提示し、付録は関連動向および手法の概観を参照情報としてまとめる。具体的な分析、ケース適用、評価設計等の詳細は別冊に詳述する。以下、各目的と成果物の対応を示す。

- 第一の目的は、LegalTech の現状を構造的に整理することである。国内外における最新動向を体系化し、従来型の LegalTech （LegalTech1.0）の到達点と限界を明確化する。これにより、本研究の問題設定の妥当性を実証的に裏付ける（本体：要点整理、付録 A：動向詳細）。
- 第二の目的は、法益を基軸とした目的志向型法設計思想を提示することである。条文中心主義から脱却し、ベンチマークとリスク評価を中核とする設計モデルを提示する（本体：枠組み提示、別冊 1：枠組み詳細、戦略提言）。
- 第三の目的は、ソフトウェアエンジニアリングの手法を用いた検証である。モデリング、要求定義、シミュレーション、証拠設計を実際に適用し、理論の実装可能性を確認する（本体：位置づけ・論点整理、付録 B：手法概観、別冊 2：ケーススタディ）。
- 第四の目的は、ジョイントサーティフィケーションの制度設計の方向性を提示することである。システム、組織、ガバナンス（証拠、継続改善等）を統合し国際的にも通用する適合性評価モデルを構想し、標準化への道筋を示す（本体：全体像、別冊 3：概説）。

本研究は、LegalTech1.0 から LegalTech2.0 への進化を理論化するのみならず、その橋渡しを行う方法論として「**Legal Engineering for Software-Defined Society（LE4SDS）**」を提示する。LE4SDS は、「目的 → ベンチマーク → 実装 → 証拠 → 評価 → 更新」の循環構造を確立することで、法と標準と技術を再統合する枠組みである。法と標準の二層構造を前提に、要求工学を中核とする体系を構築する。

本調査研究の最終的な意義は、日本が AI 時代において制度設計能力を強化し、技術革新と社会的信頼を両立させる基盤を構築することにある。

第 3 章　本研究の概要

3.1 本研究の位置づけ

本調査研究は、「Software-Defined Society（SDS）」を実現するために、法・標準・技術を再統合する枠組みを構築することを目的とする。

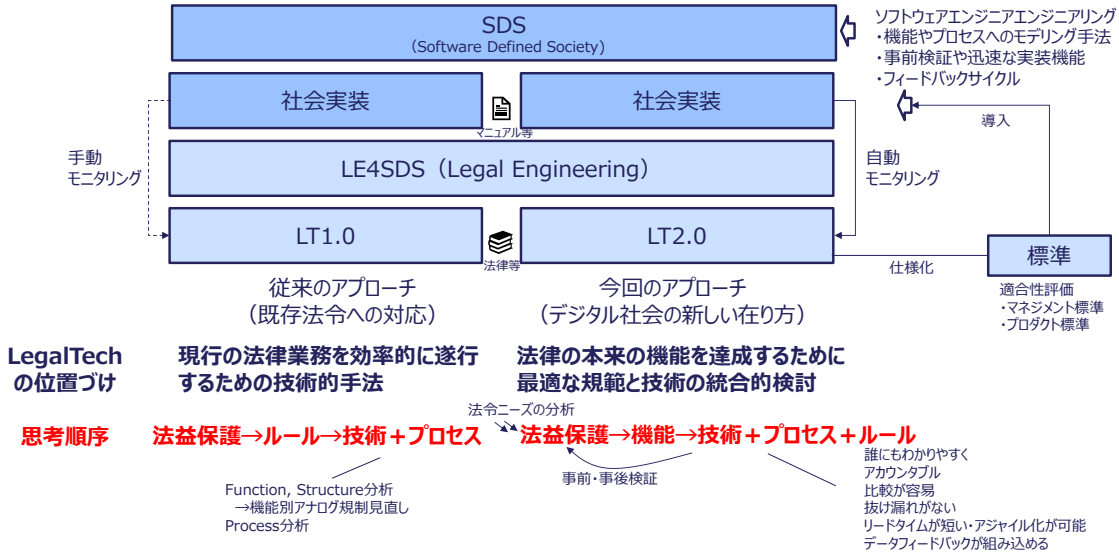


図 5 本調査研究の全体像

SDS とは、社会機能がソフトウェアによって定義・実装され、継続的に更新される社会構造を指す。この社会においては、ルール（法・標準・規制）は静的な外在的制約ではなく、ソフトウェアと融合した実装可能な構造として設計される必要がある。

本研究では、従来型の LegalTech（以下、LT1.0）と、本研究で提案する LegalTech（以下、LT2.0）を明確に区別し、その差異を理論化した上で、LE4SDS（Legal Engineering for Software-Defined Society）という方法論として確立する。そして、その有効性を検証し、さらに社会実装のための制度的担保として標準（ジョイントサーティフィケーション）を位置づける。

ここで、本研究における定義を整理する。

- **LegalTech（リーガルテック）**
  - ソフトウェアの考え方や技術を法律や制度といった**“ルール”**の設計・運用に取り入れ、より柔軟かつ迅速なルールのアップデートを可能とする技術の総称。
- **Legal Engineering（リーガルエンジニアリング）**

- 対象となるルールの機能を特定した上で、その機能を補完・代替しうる技術を用いることによって、ルールと技術を統合してルール本来の目的を実現する手法。単なる技術の適用ではなく、エンジニアリングとして、一意に理解でき、再現でき、検証でき、説明でき、追跡でき、改善できる性質を持つ。

- **Legal Engineering for Software-Defined Society（LE4SDS）**
  - SDS（Software-Defined Society）とは、社会制度や公共サービス等のルールが、ソフトウェアによって実装・運用されることを前提に、制度改正・運用変更・社会環境変化に対して継続的に適応する社会像を指す。
  - LE4SDS（Legal Engineering for SDS）とは、SDS の実現に向けて、法・標準等の「ルール」をソフトウェアエンジニアリングの方法論と接続し、企画から運用・更新までを一貫して扱うリーガルエンジニアリングの方法論である。

さらに、LegalTech を 2 つに分ける。

- **現在の LegalTech（LT1.0）**
  - 主として「現行の法律業務を効率的に遂行するための技術的手法」

- **本研究における LegalTech（LT2.0）**
  - 「法律の本来の機能」を達成するために最適な規範と技術との統合的運用の手法
  - ある法の法益保護機能を達成するために、技術及び規範が最適な形で統合的に運用され、それらは最先端の技術動向を踏まえてタイムリーに更新される

なお、「Legal」の定義であるが、社会は、交通ルールや手続き、社会保障など様々なルールで形成されており、そのルールはソフトウェアで提供されるものも多い。「Legal」という名前が付いてはいるが、法律や制度だけでなく、民間の社内規則、自治体の条例、標準などでも適用可能であり、適用対象は幅広いと考えている。

3.2 LegalTech1.0（LT1.0）の構造と限界

LT1.0 は、「既存法令への対応」を中心とするアプローチである。LT1.0 は主として現行の法律業務を効率的に遂行するための技術的手法であり、法令検索、契約レビュー、電子署名などが典型例である。

その思考順序は、「**法益保護 → ルール → 技術＋プロセス**」という順序である。すなわち、既存のルールを前提とし、それをいかに効率的に運用するかという観点に立脚している。分析の対象も主として Structure 分析（条文構造の整理）や Process 分析（既存手続の見直し）に留まる。

このアプローチは一定の合理化を実現したが、以下の限界を抱えている。

- 法制度そのものの再設計には踏み込まない
- モニタリングは手動であり、事後的である
- データフィードバックが制度設計に組み込まれていない
- 古い法律に内在する目的不明確性や情報不足に対処できない

つまり、LT1.0 は制度の「効率化」であり、制度の「再設計」ではない。

一方、LegalTech を制度の「再設計」に適用するためには解決すべき課題もあり、以降に代表的な 4 つの課題と LT1.0 のアプローチの限界について述べる。

⑴ 法の「曖昧性」と「解釈」の壁

法規範の多くは、意図的に抽象的・一般的な言葉（「公序良俗」「信義誠実の原則」「合理的」など）を用いて記述される。これは、予測困難な個別の事案（コンテキスト）に応じて、裁判所や行政が柔軟に解釈し、妥当な結論を導くための「余白」である。

LT1.0 の限界

- プログラミングコードは、原則として明確（Explicit）かつ一義的（Unambiguous）でなければ実行できない。
- AI によって高速化する社会は、従来の人間社会よりも一層多様かつ複雑である。無限に近い「コンテキスト」を事前にすべて想定し、「合理的」とは何か、「公序良俗」に反するとは何かをコードで定義し尽くすことは事実上不可能と考えられる。

⑵ リスクシナリオの爆発的増大と事前記述の限界

AI・デジタル技術は、社会システムや人間関係の相互作用を飛躍的に複雑化させる。これにより、従来は想定し得なかった新たなリスクシナリオが爆発的に増大する。

LT1.0 の限界

- 「Law as Code」は、基本的に既知のリスクや予測可能な違反行為を前提としてルールを設計（コーディング）するアプローチである。
- AI が他の AI や社会インフラと自律的に相互作用する場合など、その組み合わせによって生じる潜在的な危害や抜け穴（ループホール）は無数に存在し得る。これら無数のリスクシナリオと、それに対する妥当な軽減策（例外処理）を、人間が事前にすべて網羅してコードに記述することは、現実的に不可能である。結果として、コード化された法は、未知のリスクに対して脆弱となり得る。

⑶ 「価値判断」のコード化の困難性

AI 社会、特に自律型 AI（自動運転車、AI による診断など）は、人間の生命や権利に関わる倫理的なジレンマに直面する（例：自動運転における「トロッコ問題」）。

LT1.0 の限界

- 多くの論点は、相反する価値（例：個人のプライバシー vs 公共の安全）の衡量（バランス）を求めるものであり、法律によって唯一絶対の「正解」を提供することは困難。そのため、法はこれらの問題に対する「判断の枠組み」や「手続き（デュー・プロセス）」を定められるにとどまる。
- したがって、これらの高度な価値判断や倫理的衡量を、事前にプログラマーがコードに「正解」として埋め込むことは、社会的な合意形成の観点からも、技術的な観点からも極めて困難であるし、適切でもない（特定の価値観を固定化・強制するリスクを伴う）のではないか。

(4) **「コードの支配」による法の硬直化**

「Law as Code」が徹底され、「Code is Law」（コードが法そのもの）の状態になると、法システムが本来持つ柔軟性や自己修正機能が失われる危険がある。

LT1.0 の限界

- 法は本来、社会の変化や個別の事情に応じて解釈が見直され、必要であれば立法（改正）や判例の変更を通じて更新されていく動的なシステムである。不当な法適用に対しては「異議申し立て（裁判など）」が保障されている。
- AI が高速・大規模に「コード化された法」を自動執行する場合、そのコードにバグがあったり、設計が不公正だったりした場合の被害は甚大となり得る。また、コードのロジック（アルゴリズム）の妥当性を市民が検証し、異議を申し立てるプロセス（アルゴリズムに対するデュー・プロセス）を確立することは非常に困難となる可能性がある。

3.3 LegalTech2.0（LT2.0）の構造転換

これに対し、LT2.0 は「デジタル社会の新しい在り方」を前提とする。LT2.0 では、法律の本来の機能（法益保護）を起点とし、それを達成するために最適な規範と技術を統合的に検討する。

思考順序は次のように転換する。

「**法益保護 → 機能 → 技術＋プロセス＋ルール**」

ここではルールは出発点ではなく、設計対象となる。

LT2.0 の特徴は以下の通りである。

- 法益を明示し、ベンチマーク化する
- 機能単位で制度を再設計する
- ソフトウェアエンジニアリング手法を導入する
- 事前検証（シミュレーション）を行う
- 実装後に自動モニタリングを行う
- データフィードバックを制度更新に組み込む

LT2.0 ではモニタリングは「自動化」される。法律がソフトウェア的に埋め込まれることで、実行ログ、監査証跡、リスク指標などを自動収集し、改善ループを回すことが可能となる。これは単なる効率化ではなく、制度の動的化である。

3.4 LE4SDS という方法論

「スパゲッティ前提」の社会は持続不能であり、保護法益（目的）とリスクを基軸に、ベンチマーク（達成基準）を明示し、到達方法は多様な技術・運用の組合せで達成可能にする設計へ転換する必要がある。

本研究は、LT1.0 と LT2.0 を単なる概念区分に留めず、それを橋渡しする方法論として LE4SDS を確立する。

まずは、LE4SDS の視点を示す。まずは、社会変化に対して、スピードとアジャイル性を持って対応する必要がある。ただし単にそれだけでは社会に適用できない。様々な分野やグローバルにつながるインタオペラビリティが必要であり、またそのためのセマンティックスの定義も重要である。特に法令や規則では、様々な言葉の定義が行われており、社会の辞書としての機能も重要となる。そして適用に当たってはパーソナライズされる必要があり、コンプライアンス対応のための可視化、事前検証のための可視化が重要となる。

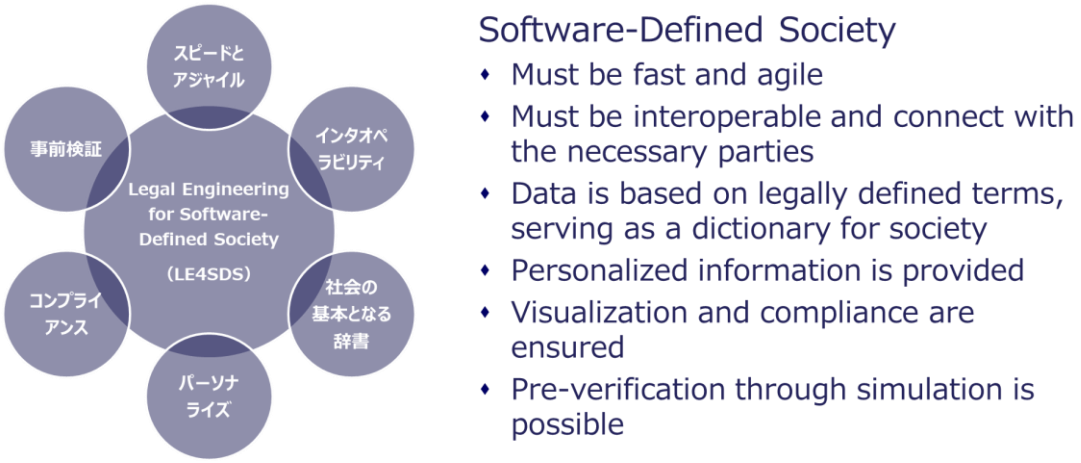


図 6 LE4SDS の視点

LE4SDS は、次の循環構造を中核とする。

「目的（法益）→ ベンチマーク→ 実装（技術＋プロセス）→ 証拠（Evidence）→ 評価→ 更新→（目的（法益））」

この循環は、ソフトウェアエンジニアリングにおける DevOps と同型である。

LE4SDS の意義は以下三点にある。

- 第一に、法制度をモデル化し、可視化し、アカウンタブルにすること。
- 第二に、比較可能性と抜け漏れ防止を確保すること。
- 第三に、制度設計のリードタイムを短縮し、アジャイル化を可能にすること。

LE4SDS は、法と標準の二層構造を前提とする。法は目的とベンチマークを規定し、標準は具体的な実装要件・試験方法・評価手順を定義する。

重要なのは、法のレベルでは「何を守るべきか」「何を達成すべきか（ベンチマークの骨格）」を定め、標準のレベルでは「どうやって守るか」「その達成基準を満たすための具体的な事項・方法」を整理するという、法と標準の「二層構造」でルールを設計することである。そのため、LT1.0 と LT2.0 をつなぐ方法論としての LE4SDS（法＋標準の橋渡し）が必要になる。

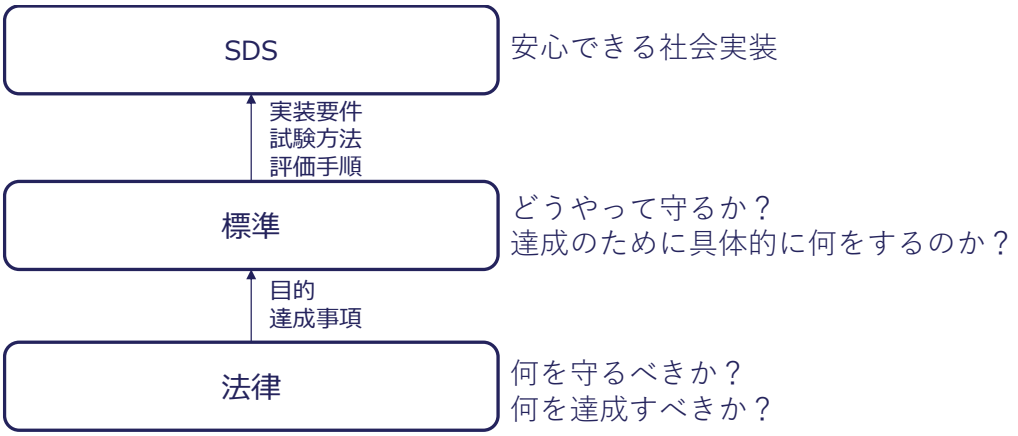


図 7　法律と標準と社会実装の関係

LE4SDS は、「法の目的・リスク」から出発し、標準に書くべき要求・試験・評価の形にまで落とし込むプロセスとして位置づく。

また、第一の意義である法律の可視化については、Human-Readable であるとともに Machine-Readable であることが重要になり、そのためにはソフトウェアエンジニアリングの要件管理手法が重要になる。

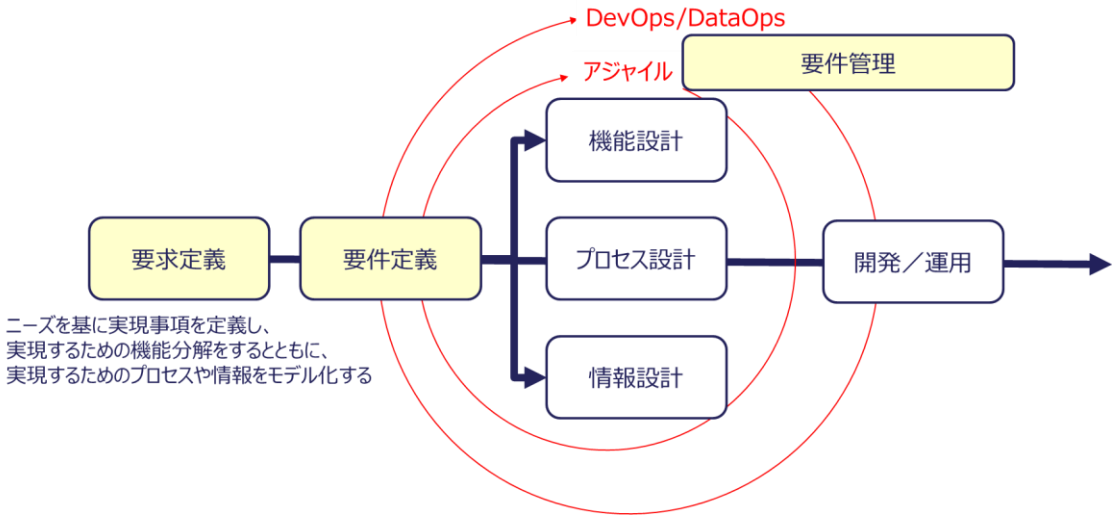


図 8 要件管理手法の適用

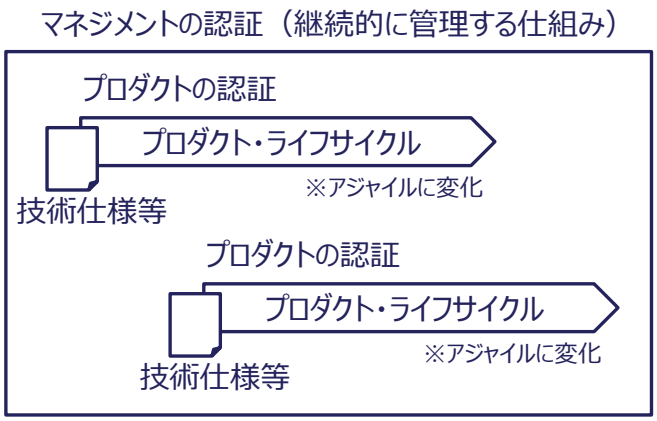
関係者のニーズである要求を可視化し、その中から実現すべきことを要件として定義し、その要件を実現し、運用の中で評価を行う要件管理を行っていく。これらは、モデリングや要件管理ツールで管理される必要がある。

3.5 標準（ジョイントサーティフィケーション）による補完アプローチ

LT2.0 を社会に確実に埋め込むためには、単なる設計思想では不十分である。制度を実効的に運用するための担保メカニズムが必要となる。重要なのは、設計された法・標準・システムが本当に目的達成とリスク抑制に貢献しているかをどう担保するかである。

AI は継続的にアップデートするとともに部分的ブラックボックスであり、システムだけを静的に見る評価は限界である。ここでの「標準」は、単なる技術仕様ではなく、「何を証拠として集めるか（Evidence）」「どう審査するか（Conformity Assessment のプロセス）」を規定する器として機能する。LE4SDS で定義したベンチマークとエビデンスを材料に、「システム × 組織（人・プロセス・ガバナンス）」を一体で評価する新タイプの第三者適合性評価を設計し、その観点・手順も標準化（標準・技術仕様・ガイドライン）することで、多様な事業者・システム間で一貫した評価を可能にする。評価は一度きりではなく、標準に基づく継続モニタリング／改善（アジャイル適合性）として回していく。

特にジョイントサーティフィケーションは、組織マネジメント認証とプロダクト認証を統合し、継続的改善能力と実装安全性を同時に評価する新しい枠組みである。AI のように更新を前提とする技術に対しては、「点」の認証ではなく「プロセスと証拠」の認証が必要となる。ジョイントサーティフィケーションは、その制度的実装形態である。



- 何を証拠として集めるか（Evidence）
- どう審査するか（Conformity Assessmentのプロセス）

図 9 ジョイントサーティフィケーションの概要

3.6 結論

本研究の核心は、LegalTech の効率化段階（LT1.0）から、目的志向型統合設計段階（LT2.0）への転換を理論化し、それを LE4SDS として体系化し、さらに標準（ジョイントサーティフィケーション）によって社会に埋め込む枠組みを提示することである。

これは、単なる LegalTech 研究ではない。この全体構造は、SDS を実現するための制度設計アーキテクチャであり、**AI 時代における「社会 OS」の再設計**である。

付録 A　LegalTech の最新動向

A.1 日本の LegalTech の視野の狭さと限界

日本における現在の LegalTech は主として効率化技術である。

- 契約書レビュー
- 法令検索
- 電子署名
- 文書管理

など

しかしこれは「既存制度の作業効率化」に過ぎない。制度設計の構造そのものを変革していない。

A.2 プロセスベースでの LegalTech の動向整理

法律は、突然できあがるものではない。社会の課題を把握し、ルールを設計し、制度として導入し、運用し、改善していく一連のプロセスを経て社会に定着する。

本節では、その流れを法案策定の企画・法案作成・導入・運用の 4 つの段階と、実際に法案を活用する利活用の段階に分け、それぞれの段階で LegalTech がどのような役割を果たしているのかを説明する。

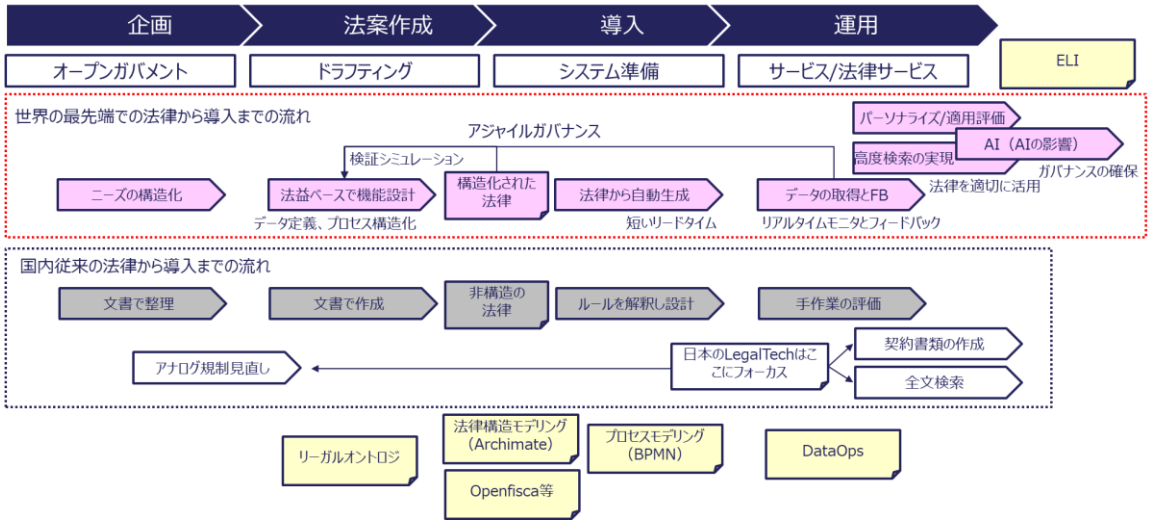


図 10 法律策定プロセス

A.2.1 企画（Policy Design）

企画段階は、「なぜ法律が必要なのか」を考える最初の段階である。社会で何が問題になっているのか、誰が困っているのか、どのような価値（安全・公平・プライバシー・市場の健全性・環境保護など）を守る必要があるのかを整理する工程である。

従来は、専門家の経験や一部の統計データ、有識者会議に基づいて政策が検討されてきた。しかしデジタル社会においては、社会変化の速度が格段に上がっている。AI、プラットフォーム経済、越境データ流通などの領域では、数年単位ではなく月単位で環境が変化する。このため、政策企画もより客観的・継続的・動的なアプローチが求められる。

現在は、データ分析や AI を活用することで、より客観的に社会のニーズやリスクを把握できるようになっている。しかし、単にデータを集めるだけでは不十分であり、「何を守るための制度か」という法益との接続、将来リスクの予測、国際整合性の検討、実装可能性の検証まで含めた総合的設計が必要となる。

また、政策設計（Policy Design）における要求工学の適用も検討できる。

- **概念：**法律が守るべき基本的な「法益」（安全性、プライバシー、公平性など）を、ソフトウェアの**要求仕様**として定義し直す。
- **活動：**政策目的や保護すべき価値を明確化し、データ分析・シミュレーションにより制度導入後の影響を予測・検証する。たとえば、法律を施行した場合の財政負担、利用者行動の変化、社会的副作用などをデジタルツイン（仮想社会）上でシミュレーションする。
- **最新動向（2025–2026）：** OECD などが提唱する EBPM（Evidence-Based Policy Making：証拠に基づく政策立案）に AI シミュレーションを組み合わせ、施行前に制度のバグ（想定外の副作用）を早期発見する手法が普及し始めている。データドリブンな政策評価や目標志向ガバナンスの考え方が各国で導入されつつある。[1]

以下に、企画段階における主な LegalTech の取り組みを整理する。

① EBPM（Evidence-Based Policy Making）

目的

- 思い込みや経験則ではなく、データに基づいて政策を設計すること。

1（注）政府における Rules as Code（ルールの機械可読化）等の実験と学びの整理として、OECD OPSI によるカナダ事例の取りまとめを参照。 OECD OPSI, “Rules as Code in Canada: Summary of Experiments and Lessons Learned” (Feb 2024), https://oecd-opsi.org/wp-content/uploads/2024/04/Rules-as-Code-in-Canada.pdf （参照日：2026-03-03）

説明

- オープンデータや各種統計データ、行政データ、民間データを活用し、政策課題の実態を把握する。例えば、「若者の雇用が不安定だ」という仮説に対して、地域別・年代別のデータを分析し、実態を確認する。その上で、どの層にどのような制度が必要かを設計する。
- さらに近年では、単なる過去データの分析にとどまらず、将来予測モデルを用いたシミュレーションも行われている。制度導入後の財政影響や行動変容を事前に推計することが可能になりつつある。

不足点

- データはあっても、「どの法益を守るための制度なのか」という目的との結びつきが弱い場合がある。また、定量化できない価値（信頼、公平感、倫理性）をどう評価するかという課題も残る。

サービス例

- 各国のオープンデータポータル
- データ可視化ツール（Tableau、Power BI 等）
- 政府の政策評価システム

② **国民の声の分析（テキスト分析）**

目的

- 国民や企業の声を体系的に把握すること。

説明

- 有識者委員会による代表的意見集約に加え、アンケート、パブリックコメント、SNS 投稿などを分析し、「どのような不満が多いか」「どの制度に改善要求が集中しているか」を整理する。
- これにより、従来は見えにくかった潜在的課題や少数意見を可視化できる可能性がある。

不足点

- 声は集められても、それを制度設計にどう組み込むかの明確なプロセスが不足している。また、パブリックコメントについては、募集自体が認知されていない、回答が形式的に見えるといった課題も指摘されている。さらに、SNS 分析ではノイズや偏りの影響を受けやすい。

サービス例

- テキスト分析サービス
- 行政向け意見分析ツール

③ 対話型サイトなどオープンガバメントの仕組み

目的

- 国民や企業が政策形成に主体的に関与できる仕組みを構築すること。

説明

- 対話型プラットフォームを通じて、市民が政策案に対して意見を述べたり、代替案を提示したり、請願を行ったりできる仕組みである。場合によっては、クラウドファンディング等を通じて行政に依存しない社会実装も可能となる。
- これにより、政策形成が「説明されるもの」から「共創されるもの」へと変化する。

不足点

- 議論を建設的に進めるためには高度なファシリテーション能力が必要であり、実施できる人材が限られている。また、集まった議論を法制度にどう反映するかの設計が未成熟である。

サービス例

- デジタル請願サービス
- 対話型サイト（Decidim[2]等）
- GitHub を用いた政策草案公開
- クラウドファンディング

④ アナログ規制の見直し（Regulatory Reform by Digitalization）

目的

- 技術進展に適合しない旧来の規制を再検討すること。

説明

- 対面義務、紙媒体保存義務、常駐要件など、デジタル化により代替可能な規制を見直す取り組みである。これは単なる効率化ではなく、規制目的（法益）を再確認し、より合理的な達成手段へ転換することを意味する。

不足点

- 個別規制の置き換えにとどまり、制度全体の構造改革や目的志向設計にまでは至らない場合が多い。

サービス例

2 Decidim, “Decidim: A participatory democracy platform”, https://decidim.org/ （参照日：2026-03-03）

- デジタル庁アナログ規制の見直し[3]

⑤ リスク分析・影響評価の高度化

目的

- 制度導入による副作用や新たなリスクを事前に把握すること。

説明

- AI やデジタル技術を用いて、制度変更が市場や市民行動に与える影響をモデル化する。規制影響評価（Regulatory Impact Assessment：RIA）を定量的に高度化し、複数シナリオを比較検討する。

不足点

- リスクは動的に変化するため、事前評価だけでは不十分であり、継続的モニタリングとの接続が必要である。

⑥ 国際整合性・リーガルインタオペラビリティの確認

目的

- 国際的な法制度との整合性を確保すること。

説明

- 新制度が他国法令や国際標準と整合しているかを確認する。特にデータ、AI、サイバー分野では、越境取引を前提に設計する必要がある。

不足点

- 各国法令の構造が異なるため、横断比較が困難であり、体系的な方法論が未成熟である。

サービス例

- AISI 日米クロスウォーク[4]

⑦ 実装可能性の事前検証（Implementation Feasibility）

目的

- 制度が現実に運用可能かを事前に確認すること。

説明

---

[3] デジタル庁,アナログ規制見直しの取組,https://www.digital.go.jp/policies/digital-extraordinary-administrative-research-committee（参照日：2026-03-03）

4 AISI, 「AI 事業者ガイドラインと米国 NIST AI RMF のクロスウォーク（1/2）」, https://aisi.go.jp/output/output_information/240918_1/ （参照日：2026-03-03）

- 法案段階で、システム実装や業務プロセスに落とし込んだ際の負荷やコストを検証する。モデル化や試験実装を通じて、実務との乖離を減らす。

不足点

- 立法と実装が分断されている場合、後工程で大幅修正が発生する。

サービス例

- OpenFisca[5]

⑧ 法益ベース設計

目的

- 「何を守るための法律か」を明確にし、そのための因数分解を行うこと。

説明

- 例えば自動運転車の制度では、「交通の安全」「責任の明確化」「イノベーション促進」などの法益を明確にし、それを達成するための具体的要件へ分解する。

不足点

- 法益を技術要件へ体系的に変換する方法論が確立途上であり、ここに LE4SDS の意義がある。

A.2.2 法案作成（Drafting）

法案作成段階は、企画段階で整理された政策目的・法益・リスク・要求を、実際の条文として具体化する工程である。
この段階では、

- わかりやすさ（明確性）
- 矛盾のなさ（一貫性）
- 抜け漏れのなさ（完全性）
- 実装可能性（運用可能性）
- 他法令との整合性

が重要となる。

従来の法案作成は、専門的な法技術に基づく高度な文章作成作業であり、自然言語による条文化が中心であった。しかし、Software-Defined Society においては、法律は単なる文章ではなく、システム実装や自動判断に直接影響を与える「仕様」としての側面を持つ。

---

[5] OpenFisca, “OpenFisca”, https://openfisca.org/en/ （参照日：2026-03-03）

そのため、条文を自然言語で整えるだけではなく、構造的・論理的に整理し、検証可能な形で記述する必要がある。LegalTech は、この段階において、法律を「見える化」し、「検証可能」にするための技術基盤を提供する。

起草（Drafting)における構造化とは

- **概念：** 法律を従来通りの自然言語で記述する一方で、コンピュータが処理できる**データ形式**（XML など）でも表現する。
- **活動：** OASIS の LegalDocML（Akoma Ntoso）等を用い、条文を章・条・項など階層構造でマークアップする。法令の改正履歴や条文間の参照関係もタグで明示し、Git 等のバージョン管理システムで共同編集・差分管理する。[6]
- **最新動向：** 生成 AI（LLM）を用いて、過去法令や関連判例との整合性をチェックする「リーガル・リンティング（Legal Linting）」機能が研究・実装されている。これにより起案段階で矛盾や曖昧さを自動検出できるようになり、立案効率と品質が向上している。文書構造のモデル化により、専門家以外にも法律の設計意図が伝わりやすくなる。

以下に、法案作成段階における主な LegalTech の取り組みと課題を整理する。

① 法律構造モデリング

目的

- 法律の構造を図やモデルで可視化し、法律間や条文間の関係性や論理構造を明確にすること。

説明

- 法律は通常、章・節・条・項・号といった階層構造を持つ。また、定義規定、要件規定、効果規定、罰則規定などが複雑に相互参照している。
- 自然文のままでは、次のような課題が生じる。
  - どの条文がどの定義に依拠しているかが分かりにくい
  - 例外規定がどこまで適用されるかが曖昧になる
  - 改正時に影響範囲が把握しづらい
- 法律構造モデリングでは、これらを図や構造モデルとして整理する。以下のような取り組みが行われている。
  - 「定義 → 要件 → 効果 → 罰則」の関係を図示

6 OASIS, LegalDocML Technical Committee (Akoma Ntoso / LegalDocML), https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=legaldocml （参照日：2026-03-03）

- 参照関係をネットワークとして可視化
- 条文ごとの役割を分類

これにより、立法者・審査担当者・実装担当者が共通理解を持ちやすくなる。

不足点

- 構造の可視化は可能であるが、「合理的」「相当」「必要な措置」といった抽象概念の解釈そのものまでは自動的に解決できない。意味論的解釈は依然として人間の判断を要する。

提供サービス例

- ArchiMate（制度構造をアーキテクチャとして整理）
- LegalDocML / Akoma Ntoso（法令構造の機械可読化）

② **プロセスモデリング（BPMN 等）**

目的

- 法律が想定する行政手続や事業者の行動プロセスを図式化すること。

説明

- 多くの法律は、単なる禁止・義務規定にとどまらず、「申請」「審査」「認可」「報告」「是正命令」などの手続きを規定している。しかし、これらを文章で読むだけでは、実際の運用フローが見えにくい。
- プロセスモデリングでは、以下の項目をフローチャート形式で表現する。
  - 誰が
  - いつ
  - どの条件で
  - どのような手続きを行い
  - どのような判断をするのか
- これにより、以下が可視化される。
  - 手続きの重複
  - 不必要な待機時間
  - 権限の曖昧さ
  - 例外処理の不足

これらは、デジタル化前提の行政手続を設計する際にも不可欠である。

不足点

- 裁量判断や個別事情を考慮する規定は、単純なフロー図では表現が難しい。また、法的解釈と運用実務が乖離する場合、図が実態を反映しない可能性がある。

提供サービス例

- Camunda

- Bizagi
- BPMN（Business Process Model and Notation）[7]対応ツール

③ Rules as Code：ルールのコード化（LegalDocML、DMN 等）

目的

- 法律の条件と効果を計算可能な形に変換すること。

説明

- 法律には、「もし A ならば B とする」という条件分岐構造が多く存在する。以下に例示する。
  - 所得が一定額以下であれば給付対象
  - 一定基準を満たせば許可
  - 違反があれば罰則

  これらをプログラムとして記述することで、判定を自動化できる。
- ルールのコード化により、次のことが可能になる。
  - 給付額の自動計算
  - 適合判定の自動化
  - 条文の論理矛盾の検出
  - 政策効果の事前シミュレーション

  これは、法案作成段階で論理構造を事前検証する手段ともなる。

不足点

- 抽象概念や倫理的判断（合理的、公正、適切など）はコード化が難しい。また、全てをコードに落とし込もうとすると、法の柔軟性を損なう可能性がある。
- また、Rules as Code の試みも、次の限界に直面する。
  1. 抽象概念の完全コード化は不可能
  2. 未知リスクの事前網羅は困難
  3. 倫理的価値判断の固定化リスク
  4. コード化による硬直化
- したがって、「Rule as Code」の単純推進では不十分である。必要なのは、目的・ベンチマーク・実装・評価・更新を回す構造的枠組みである。

提供サービス例

- OpenFisca

---

[7] Object Management Group (OMG), “Business Process Model and Notation (BPMN)”, https://www.omg.org/bpmn/ （参照日：2026-03-03）

- DMN（Decision Model and Notation）[8]
- ルールエンジン

④ 用語定義と辞書化（Semantic Structuring）

目的

- 法律用語の意味を一意に定義し、誤解を防ぐこと。

説明

- 多くの法令では用語が定義されているが、それが横断的な辞書として整備されていない場合が多い。
- 用語を辞書化し、構造化することで、以下が確保される。
  - 解釈の一貫性
  - 他法令との整合性
  - データ項目との対応

  これは Machine-Readable 化の基盤でもある。

不足点

- 既存法令との整合を取るには大規模な整理が必要である。

⑤ 条文間整合性チェックと影響分析

目的

- 条文の矛盾や参照漏れを検出すること。

説明

- 構造化された法令は、参照関係や定義依存関係を機械的に解析できる。
- 改正時には、以下を迅速に確認できる。
  - 影響を受ける条文
  - 他法令への波及
  - 標準やガイドラインとの整合

不足点

- 完全な自動検証は困難であり、専門家による確認が必要である。

⑥ 実装連携（Compliance by Design への接続）

目的

---

[8] Object Management Group (OMG), “Decision Model and Notation (DMN)”, https://www.omg.org/dmn/ （参照日：2026-03-03）

- 条文化段階から実装を見据えた設計を行うこと。

説明

- 従来は、法案成立後に実装検討が始まることが多かった。しかし、LE4SDS の視点では、法案作成段階で、以下を考慮する。
  - データ取得可能性
  - システム実装可能性
  - 証拠収集可能性

  これにより、法とシステムの乖離を最小化できる。

法案作成段階における課題

法案作成段階の最大の課題は、「自然言語による柔軟性」と「構造化による厳密性」のバランスである。過度に厳密化すれば硬直化し、過度に抽象化すれば実装不能となる。
したがって、以下の三点を統合することが重要である。

- Human-Readable な条文化
- Machine-Readable な構造化
- 要求工学によるトレーサビリティ確保

LE4SDS との接続

法案作成は、LE4SDS の循環において「目的 → リスク → 要求 → 条文」へと至る工程である。この段階で構造化と検証を行わなければ、その後の「実装 → 証拠 → 評価 → 更新」が不安定になる。

したがって、法案作成段階は単なる文章化作業ではなく、「**制度の論理構造を設計するエンジニアリング工程**」として再定義されるべきである。

A.2.3 導入（Implementation）

導入段階は、成立した法律を現実の行政運用・事業活動・情報システムへと具体的に組み込む工程である。ここで初めて、法律は社会の中で実際に機能し始める。

従来、導入は「制度の周知」「様式の整備」「業務マニュアル作成」「システム改修」といった作業として扱われてきた。しかし、Software-Defined Society においては、法律は単なる遵守対象ではなく、**システムの振る舞いを規定する仕様**となる。そのため、導入段階は「**法を実装するエンジニアリング工程**」として再定義される必要がある。

この段階では、次の観点が重要となる。

- 法律の要件がシステムに正しく反映されているか
- 必要なデータが取得・保存・検証可能か
- 判断過程が説明可能か（トレーサビリティ）
- 将来の改正や更新に対応できる構造か

LegalTech は、この導入工程において法と技術の接合を支える役割を果たす。

実装（Implementation）における Rules as Code（RaC）

- **概念：**法律の論理（例えば「もし A ならば B」）をプログラムコード（API）として公開し、外部システムが直接利用できるようにする。
- **活動：**法的判断ロジックを DMN（Decision Model and Notation）などの計算可能モデルに落とし込み、「条件→結果」の規則をソフトウェアエンジニアリング手法で実装する。税額計算、給付可否判定、ライセンス認可などを自動化し、Web API で提供する。
- **最新動向：**世界各国の政府が税制や補助金、社会保障制度などの計算ロジックを Open API として公開し、誰でも利用できるようにする「Computable Law」（計算可能な法律）が標準になりつつある。たとえばカナダやフランスでは、法律テキストと並行して機械読取可能なルール実装を整備する Rules as Code が進展している。[9]

以下に、導入作成段階における主な LegalTech の取り組みと課題を整理する。

① Compliance by Design（設計段階からの法組込み）

目的

- 法律を後付けの制約ではなく、システム設計の前提条件として組み込むこと。

説明

- 従来は、システムが完成した後に「法的に問題がないか」を確認することが多かった。しかしこの方法では、後からの修正コストが大きく、コンプライアンスリスクも高まる。
- Compliance by Design では、以下の取り組みを行う。
  - 法律の要件をシステム仕様書に直接反映
  - アクセス制御やログ保存要件を初期設計に組み込み
  - データ最小化や保存期間制限を構造として実装

  これにより、法遵守は「意識」に依存するのではなく、「構造」として保証される。

不足点

9（注）各国での試行・整備の進展を示す根拠として、Rules as Code の政府実験に関する OECD OPSI の整理を参照。 OECD OPSI, “Rules as Code in Canada: Summary of Experiments and Lessons Learned” (Feb 2024), https://oecd-opsi.org/wp-content/uploads/2024/04/Rules-as-Code-in-Canada.pdf （参照日：2026-03-03）

- 法律の抽象的規定を具体的技術仕様へ変換する能力が必要であり、法務とエンジニアの協働体制が不可欠である。

⑤ データ設計とエビデンス基盤の整備（Evidence Infrastructure）

目的

- 制度運用の証拠を自動的に収集・保存できる基盤を構築すること。

説明

- LE4SDS では、「証拠（Evidence）」が極めて重要である。制度が目的を達成しているかを評価するには、運用データが不可欠である。
- 導入段階では、以下の項目を設計する。
  - 必要なデータ項目の定義
  - ログ設計
  - 証拠の保存方法
  - データ改ざん防止

  これにより、「法 → 要求 → 実装 → 証拠 → 評価」の循環が成立する。

不足点

- 証拠収集が過剰になるとプライバシー侵害やコスト増大のリスクがあるため、バランス設計が必要である。

⑥ ルールエンジン・判定エンジンの実装

目的

- 法律の条件判断を自動化すること。

説明

- 税制、給付制度、許認可制度などでは、条件分岐が明確な部分については判定エンジンを実装できる。以下はその例であり、これらを入力として、自動判定を行うことができる。
  - 申請者の属性データ
  - 所得情報
  - 行為履歴
- これにより、以下の効果が期待できる。
  - 処理速度の向上
  - 人的ミスの削減
  - 公平性の確保

不足点

- 裁量判断や倫理的評価を完全に自動化することは困難であり、Human-in-the-loop 設計（重要判断や例外処理に人が介在する運用設計）が必要である。

⑦ API 化・相互運用性の確保（Interoperability）

目的

- 制度を他システムと接続可能にすること。

説明

- 導入段階では、法律に基づく判定機能を API として公開することで、民間サービスや他行政機関との連携が可能になる。以下が API の例である。
  - 補助金対象判定 API
  - 認可状況照会 API
  - コンプライアンスチェック API

  これにより、制度は「静的な文書」から「動的なサービス」へと変化する。

不足点

- 標準化やセマンティクスの統一がなければ、相互運用は困難となる。

⑧ 運用前シミュレーションとテスト（Pre-Deployment Testing）

目的

- 制度実装前に影響を検証すること。

説明

- デジタルツインやテストデータを用いて、以下のケースを検証する。
  - 想定ケース
  - 極端ケース
  - 境界条件

  これにより、制度バグや想定外の副作用を事前に把握できる。

不足点

- 未知のリスクを完全に排除することはできないため、継続的モニタリングとの接続が必要である。

⑨ 組織体制とガバナンスの整備

目的

- 制度実装を支える組織的能力を確保すること。

説明

- 導入段階では、法務、IT、データ管理、セキュリティ、リスク管理の各部門が連携する体制が必要となる。これはジョイントサーティフィケーションにおいて評価対象となる「組織能力」にも直結する。

⑩ **モニタリングの実施**

目的

- 法令遵守を継続的に監視すること。

説明

- 規制遵守状況を自動的にモニタリングする。以下の情報などを通じて、リアルタイムでコンプライアンスを確認する。
  - 異常検知
  - リスクスコアリング
  - 自動レポート生成

不足点

- 過度な自動化は、誤検知や過剰規制のリスクを伴う。

導入段階の構造的課題

導入段階では、次の課題が存在する。

- 条文と実装の乖離
- データ不足による評価不能
- 責任所在の曖昧さ
- 改正時のシステム改修コスト

これらを解消するためには、法案作成段階からの構造化と要求定義が不可欠である。

LE4SDS との接続

導入段階は、LE4SDS の循環において

　要求 → 実装 → 証拠

を担う工程である。

　ここで適切な実装と証拠収集が行われなければ、評価・更新の工程が機能しない。

したがって、導入段階は単なる運用準備ではなく、「**制度の信頼性を構造として確立する工程**」である。

A.2.4 運用（Operation）

　運用段階は、制度が実際に社会の中で機能し、その効果とリスクが顕在化する工程である。

従来、運用は「制度の執行」や「監査」に重点が置かれてきた。しかし LE4SDS では、運用は以下の循環を中核にする必要がある。

- 制度の効果を測定し
- リスクを監視し
- 改善の根拠を蓄積し
- 制度更新へと還流させる

運用は制度の「テストフェーズ」であり、同時に「学習フェーズ」である。

運用・監視 における Operation & Compliance as Code

- **概念：** 施行された法律が設計通りに機能しているかをリアルタイムでモニタリングし、運用データから証跡（エビデンス）を自動生成する。
- **活動：**システムのログをもとに、どの条文に基づきどのような判断が行われたかを追跡可能にする。運用中に得られた統計データをもとに、目標達成度やリスク状況を定期的に評価する。違反や事故が起きた場合にはログ解析により原因を迅速に特定し、フィードバックをルール改正に活かす。
- **最新動向：**AI を含むシステムはアップデートで特性が変わり得るため、AI モデルの学習状況や性能変化も監視する仕組みが注目されている。特に EU の AI 規制では、高リスク AI に対しポストマーケット監視を要求する（Article 72）、AI の品質マネジメントシステムを義務づける（Article 17）など、動的ガバナンスの法制度化が進んでいる。[10]

以下に、運用段階における主な LegalTech の取り組みを整理する。

① リアルタイムモニタリング

目的

- 制度遵守状況やリスクを継続的に監視すること。

説明

- 運用ログや統計データを用いて、「違反率」「異常値」「苦情件数」「不正兆候」等を自動検知する。これにより、問題を早期に発見できる。

10（注）EU AI Act における高リスク AI の運用上の義務（例：ポストマーケット監視、品質マネジメントシステム等）は条文に基づき整理されている。 European Commission, AI Act Service Desk, “Article 72” , https://ai-act-service-desk.ec.europa.eu/en/ai-act/article-72 （参照日：2026-03-03）; European Commission, AI Act Service Desk, “Article 17: Quality management system”, https://ai-act-service-desk.ec.europa.eu/en/ai-act/article-17（参照日：2026-03-03）

② **効果測定（Impact Assessment）**

目的

- 制度が法益を達成しているかを評価すること。

説明

- 制度の目的が安全確保であれば事故率を、公平性であれば格差指標を測定する。
- 定量指標だけでなく、「市民満足度」「公平感」「透明性評価」などの定性指標も組み合わせる。これにより、制度の実効性を多角的に評価する。

③ **フィードバック・ループ（Agile Governance）**

目的

- 運用結果を制度見直しへ迅速に反映すること。

説明

- 運用データに基づき、「想定外リスクの特定」「要件の修正」「条文改正」「技術仕様更新」を行う。これにより制度は静的なものから動的なものへ変わる。

④ **事故・不具合の検証**

目的

- トラブル発生時に原因を特定し再発防止を図ること。

説明

- ログ・判定履歴・意思決定記録を用い、以下を検証する。
  - どの条文が適用されたか
  - どのデータが判断に影響したか
  - どの工程で誤りが生じたか

  トレーサビリティ構造があることで迅速な検証が可能となる。

⑤ **国際比較と整合性確認**

目的

- 他国制度との比較を通じて制度改善を行うこと。

説明

- Machine-Readable な法制度であれば、国際的なクロスウォークが可能となり、国際競争力を強化できる。

運用段階の本質

運用は、「実装 → 証拠 → 評価 → 更新」を担う工程である。

ここでデータが収集されなければ、制度は改善できない。

導入と運用の関係

導入は構造設計、運用は学習と改善である。両者が分離していては、LE4SDS は成立しない。**制度は、設計された時点で完成するのではなく、運用によって成熟**する。

A.2.5 利活用（Utilization）

利活用段階は、成立・施行され、運用されている「ルール（法・標準）」を、国民・企業・行政の意思決定と業務の現場で「実際に使える形」に転換し、価値を継続的に生み出す段階である。従来、法律は Human-Readable（人が読む文書）として提供され、利用者は条文を読み解き、必要に応じて専門家に確認することで行動してきた。

しかし、AI 時代・Software-Defined Society では、ルールは現場の判断やサービス提供に直接影響するため、Machine-Readable（機械が処理できる）な形でも提供され、検索・照会・推薦・自動判定・監査可能性といった機能を備える必要がある。

この段階の LegalTech は、単なる「検索の高度化」に留まらない。ルールをナレッジ（構造化された知識）として扱い、個々の利用者の状況に合わせて提示し、サービスを止めずに更新し、さらに複数制度・複数国の差分を効率的に扱える状態にすることが中核になる。その結果、利用者にとっては「守るべきことが分かる」「迷わず行動できる」、提供者にとっては「コンプライアンスとイノベーションの同時達成」が可能になる。

以下に、利活用段階における主な LegalTech の取り組みを整理する。

① 高度な検索（ナレッジグラフを含む）

目的

- 法令・判例・ガイダンス・標準・Q&A 等を横断し、必要な根拠に最短で到達させる。
- 条文間の参照、改正履歴、関連制度の関係を機械的に辿れるようにする。
- AI 活用（要約・質問応答・根拠提示）の前提となる信頼できる「法の知識基盤」を作る。

説明

- 普通の検索は「キーワードが一致する文書」を探す。しかし法律は、同じ概念でも別の言葉で書かれたり、定義条文に答えがあったり、他法令の参照を辿らないと結論に行き着けないことが多い。
  そこで、法律を「章・条・項・号」「定義」「参照関係」「改正」「効力」など

の形で構造化し、知識のネットワーク（ナレッジグラフ）として扱うと、単語一致ではなく「意味と関係」で探せる。

- EU の CELLAR[11]は、法令等のメタデータをナレッジグラフ（トリプルストア）で管理し、SPARQL（RDF 等の Linked Data を問い合わせるための検索言語）で機械照会できることを明確にしている。
  同様に、英国 legislation.gov.uk でも、Linked Data と SPARQL を用いて、法令メタデータや改正（effects）等を照会できる仕組みを提供している。[12]

不足点

- 全文・条文本文の意味論（「合理的」「相当」など）まで自動で確定できない。
- データ供給側（官公庁・出版社）のメタデータ品質、ID の統一、参照表記の揺れがボトルネックである。
- 国際比較・多言語対応は、法体系の差（概念のズレ）を埋める追加設計が必要である。

サービス例

- EU：CELLAR（EUR-Lex 等を支える機械可読の提供基盤、SPARQL/API）
- UK：legislation.gov.uk の Linked Data / SPARQL（ベータ）
- 参照自動リンク：Ref2Link[13]（EU 法令参照を検出しリンク化、ELI/ECLI[14]対応）
- 生成 AI＋根拠リンク型の法情報サービス（例：Lexis+ AI[15] など、回答に引用根拠をリンク）

② パーソナライズしたリコメンデーション（制度・義務・手続の「あなた向け化」）

目的

---

11 Publications Office of the European Union, “CELLAR Data: Knowledge graph / SPARQL”, https://op.europa.eu/en/web/cellar/cellar-data/metadata/knowledge-graph （参照日：2026-03-03）

12 Legislation.gov.uk Data Documentation, “Linked Data and the SPARQL endpoint”, https://legislation.github.io/data-documentation/api/linked-data.html （参照日：2026-03-03）

13 Interoperable Europe Portal, “Ref2Link”, https://interoperable-europe.ec.europa.eu/collection/justice-law-and-security/solution/ref2link （参照日：2026-03-03）

14 EUR-Lex Help, “ELI – European Legislation Identifier”, https://eur-lex.europa.eu/content/help/eurlex-content/eli.html （参照日：2026-03-03）; European e-Justice Portal, “European Case Law Identifier (ECLI)”, https://e-justice.europa.eu/topics/legislation-and-case-law/european-case-law-identifier-ecli_en （参照日：2026-03-03）

15（注）以下は特定製品を推奨するものではなく、回答に引用根拠（参照リンク等）を付す支援機能を備えたサービス例として挙げる。 LexisNexis, “Lexis+ AI”, https://www.lexisnexis.com/ja-jp/products/global-solutions/lexis-plus-ai （参照日：2026-03-03）

- 利用者（国民・企業）が自分の状況を入力すると、適用される義務・利用できる支援制度・必要手続を提示する。
- 「読んで理解する」負担を減らし、予見可能性を高める。
- 企業の海外展開では、国・地域・業種・製品属性に応じた事前リサーチを効率化する。

説明

- 法律は「一般的に書かれる」ため、自分に当てはまるかどうかを読み解く必要がある。これを、(a) 条文構造の機械可読化（どこに何が書いてあるか）と、(b) ルール・判断の形式化（条件→結果）を組み合わせることで、利用者が入力した条件に応じて、該当する条文・義務・給付を推薦できる。
- 民間の法務領域では、会話型検索や文書分析、組織内文書（DMS）と連携したパーソナライズを特徴とするサービスが登場している。（例：Lexis+ AI のパーソナライズ／自社文書連携）。

不足点

- 「推薦」は便利だが、説明責任（なぜそう言えるか）が弱いと逆にリスクになる。
- 誤推薦時の救済（異議申立て・再審査）や、推薦ロジックの監査が必要になる。
- 国際法務では、翻訳だけでなく概念差を吸収するクロスウォーク設計が要る。

サービス例

- 法務向け生成 AI＋根拠提示：Lexis+ AI（会話型検索、要約、起案、文書分析、根拠リンク）
- （行政側での方向性）制度ナビ、受給資格判定、申請ガイドの高度化（将来的にはルールの機械可読化が鍵）

③ ノンストップサービス（Non-stop：止めない行政・止めないコンプライアンス）

目的

- 申請・審査・交付・監査・更新を、できる限り中断なく提供する。
- ルールの改正・解釈更新があっても、サービスを止めずに追随する（更新耐性）。
- 利用者の「不安・手戻り・窓口往復」を減らし、制度利用の実効性を高める。

説明

- 法律や手続が複雑だと、「書類不備で差戻し」「窓口で再説明」「制度が変わり再申請」などが起きる。

- ノンストップ化は、単にオンライン化するだけでなく、(1) ルールと手続のモデル化、(2) データ連携（必要情報を再入力させない）、(3) 変更の追跡（改正・運用変更の影響範囲を把握）、(4) 監査証跡を揃えることで実現する。
- ここで重要なのは、利活用段階のサービスが「運用・監査」と結びつき、モニタリング→改善の循環に接続される点である（LT2.0 の中核）。

不足点

- 行政手続の例外処理・裁量の扱いが難しい（自動化は段階的に）。
- データ共有の法的根拠・同意・目的外利用制約など、制度設計と一体でないと進まない。
- 「止めない更新」には、変更管理と検証（テスト）が不可欠である。

提供サービス例（方向性）

- ワークフロー基盤＋監査ログ（BPM/Case management）
- ルール判定基盤（DMN/ルールエンジン）
- 法令データの安定識別子・参照解決（ELI/ECLI、Ref2Link 等）

④ 法律のビルディングブロック化（再利用可能な「部品」としてのルール）

目的

- 法律・標準・ガイドラインを、再利用可能な単位（定義、要件、手続、判断、証拠）に分割し、組合せで実装できるようにする。
- 改正時に「全部作り直し」ではなく、影響箇所を局所化して更新できるようにする。
- 国際展開では、共通部品＋国別差分で、比較と実装を高速化する。

説明

- ソフトウェアは部品（モジュール）を組み合わせて作るため、変更に強い。同じ考え方を法律にも持ち込むと、例えば「用語定義」「適用範囲」「義務」「例外」「手続」「罰則」「証拠（ログ・監査）」を部品化し、制度ごとに組み合わせられる。
- このとき、条文そのものの構造化（どこが何か）を担う仕組みと、判断ロジック（条件→結論）を担う仕組みを分けることが重要である。たとえば Ref2Link は参照検出・リンク生成を「ビルディングブロックとして」利用できることを明示している。

不足点

- 立法技術（文章）と部品設計（機械可読）のギャップを埋める方法論が必要である。

- 部品の粒度（細かすぎると運用困難、粗すぎると再利用できない）の設計が難しい。
- 部品の正当性（法的効力・解釈の根拠）と、実装の変更管理を結びつける必要がある。

サービス例

- 構造化・識別子：ELI/ECLI、LegalDocML 系、参照抽出（Ref2Link）
- ナレッジ基盤：CELLAR のような機械アクセス前提の提供基盤（SPARQL/API）

プロセス全体にかかわるもの（利活用段階の「横串」）

利活用の段階は、単独で成立しない。特に LT2.0／LE4SDS を志向する場合、次の横断要件が効いてくる。

1. **根拠へのトレーサビリティ（説明可能性）**
   推薦・自動判定・要約が高度化するほど、「どの条文・どの版・どの解釈」に基づくかの追跡が不可欠である。機械可読の識別子、参照、改正履歴の管理が土台になる（例：EU の知識基盤、参照リンク化）。
2. **変更管理（改正・運用変更・標準更新）と継続的検証**
   「止めずに更新する」には、影響範囲の把握、テスト、段階的リリースが必要である。これは、後段の「運用→制度見直し」や「ジョイントサーティフィケーション」と接続する。
3. **相互運用性（国境・組織・制度間）**
   輸出や海外拠点の事前リサーチ効率化は、単に翻訳するだけでなく、参照解決や制度間マッピングが鍵になる。EU の Ref2Link や、機械アクセス可能な法令基盤は、その前提を提供する。
4. **安全・信頼（品質、プライバシー、セキュリティ、責任分界）**
   利活用が「意思決定の自動化」に近づくほど、誤りの影響が大きくなる。したがって、法令データの品質、AI 出力の根拠提示、監査可能性、責任分界（誰が何を保証するか）を、制度設計とセットで確立する必要がある。

A.3 諸外国における「リーガルエンジニアリング」への転換

欧米では、AI や DX がもたらす社会変化に即応するため、法律を「社会の要求仕様」と捉え、ソフトウェアエンジニアリングの手法を積極的に導入している。

① 社会変化への適応力を高める技術スタック

法律の立案から施行までのタイムラグをゼロに近づけるため、以下のエンジニアリング技術が活用されている。

- **ソーシャルネットワーク分析：** 政策立案前に国民の意見や社会の動向を多角的に把握する。
- **モデリング＆シミュレーション：** 法案を導入する前に、デジタルツイン上でその影響をシミュレーションし、意図せぬ副作用を検証する。
- **リーガル DevOps（Legal DevOps）：** 法律の実装後に得られるデータを継続的にモニタリングし、バグ（不具合）や時代遅れになった規定をアジャイルに修正・改善する。

② EU におけるアーキテクチャ思考

EU では、行政サービスと法制度を「4 つのレイヤー」で捉える体系的なアーキテクチャが採用されている。

1. **リーガル・レイヤー：** 法律・規制そのもの（要求事項）。
2. **業務・レイヤー：** 行政手続やビジネスプロセス（ワークフロー）。
3. **データ・レイヤー：** 交換される情報の定義と正確性の確保。
4. **技術・レイヤー：** 相互運用性を担保するインフラ。これにより、国境を越えたデータ連携や、複雑なコンプライアンスの自動化を実現している。

③ モデルベース・リーガル・デザインと要求工学

あらゆる事象がデジタルモデルで定義される現代において、法律もまた「デジタルツイン」としての更新性が求められている。ここで重要となるのが「要求工学（Requirements Engineering）」である。社会が求める法益を「要求」として整理し、それをシステムの「仕様」へと変換し、変更を管理し続けるエンジニアリングの規律こそが、次世代の法制度設計の核心となる。

付録 B　リーガルエンジニアリングの実装手法と体系化

本章では、Software-Defined Society における新たなルールの形である「リーガルエンジニアリング（Legal Engineering: LE）」について、その具体的なプロセスと技術体系を詳述する。AI 時代にあっては、法律はもはや「一度作られた静的な文書」ではなく、技術革新や社会変化に合わせて**継続的に更新・改善されるシステム**でなければならない。

ここでは、ソフトウェア開発ライフサイクル（Software Development Life Cycle：SDLC）の考え方を法制度に導入し、要求定義から実装、モニタリングまでを一体的に回す方法論を示す。

B.1 リーガルエンジニアリングのライフサイクル：法を「動くシステム」へ

従来、法律は国会で成立すれば改正までに数年・数十年を要する**静的な存在**であった。しかしデジタル社会では、制度の硬直化が競争力の阻害要因となり得る。そこで、リーガルエンジニアリングでは、ルールをソフトウェアのように**継続的に更新・改善可能なシステム**と捉える。

リーガルエンジニアリングでは「目的→要求→実装→証拠→評価→更新」のサイクルを回すことで、制度が常に最新の要件に適合し続ける仕組みを設計する。これにより、法律は不変の規範ではなく、ソフトウェアのように改善・進化し続ける社会の OS となる。

B.2 リーガルエンジニアリングの 9 つの専門領域

リーガルエンジニアリングは、単なる IT ツールの利用ではなく、法制度とソフトウェアの融合を実践する総合的な専門分野である。その構成要素は、以下の 9 つの技術領域からなる。本節ではそれぞれの概要を簡潔に整理する。

B.2.1 法情報に関する工学（Legal Information Engineering）

ミッション：「法の透明性を高め、誰もが迷わないナビゲーションを実現する」

単に法律をデジタル化するのではなく、法を「活用可能なデータ資産」へと変換することで、法的情報のアクセシビリティ（到達しやすさ）を根本から変革することをミッションとする。

概要：法を「静的な文書」から「動的なデータ」へ

法情報に関する工学の中核は、自然言語（日本語などの日常語）で書かれた法令、判例、ガイドラインを、コンピュータが意味を理解できる形式で構造化することにある。

- **構造化と標準化（Structural Semantics）：** 単なる PDF や Word ファイルとしての保存ではなく、XML（特に国際標準の Akoma Ntoso や LegalDocML）などの形式を用いて、条・項・号といった構造を厳密にタグ付けする。これにより、条文の「どの部分が定義で、どの部分が罰則か」をプログラムが即座に判別可能になる。また、法で定める様式もデータとして定義することが求められる。
- **網羅的な関連性マッピング（Citation Networks）：** 法律は単体で存在するのではなく、無数の参照関係（「第〇条の規定にかかわらず〜」「〇法第〇条を準用する」など）で繋がっている。これらをナレッジグラフ（知識グラフ）化することで、ある一条文の変更が、他のどの法律や規則に影響を及ぼすかを瞬時に可視化する。
- **高度な検索とバージョン管理（Consolidation & Time-travel）：**「現時点で有効な条文」だけでなく、過去の特定の時点での条文（改正前の状態）を即座に復元・比較できるバージョン管理機能を実装する。また、キーワード一致だけでなく、AI を用いたセマンティック検索（意味検索）により、専門用語を知らなくても意図した規制情報に辿り着ける環境を整備する。

活動内容

1. **法令データの構造化（XML 化）：**官報や法令全書を、機械読取可能な LegalDocML 形式へ変換・整備。
2. **判例データベースの高度化：**判決文の要旨、適用条文、論点をメタデータとして付与し、類似判例を AI で抽出。
3. **コンソリデーション（合体法令）の自動生成：**改正の度にバラバラに出る「新旧対照表」から、常に最新の「一気通貫した条文」をリアルタイムに合成・提供。
4. **法情報のポータル化（Law-as-Data）：**民間エンジニアが API を通じて法令情報を自社システムに取り込めるプラットフォームの構築。

解説：法律の「デジタル地図」を作る技術

これまでの法律探しは、いわば「索引しかない巨大な紙の地図」から目的地を探すようなものであり、どこに何が書いてあるかを知るには、専門的な訓練と膨大な時間が必要であった。

法情報に関する工学は、この紙の地図を「最新の GPS ナビ（Google マップのようなもの）」へと作り変える技術である。

- **検索：**行き先を入れれば即座に場所を示す。
- **ルート案内（関連性）：**その法律に辿り着くために、関連する他の法律や過去のルールもセットで表示。

- **更新（バージョン）：**道路が工事中（法改正）になれば、即座に地図が書き換わり、過去の古い道（旧法）と比較することも可能である。

つまり、専門家しか立ち入れなかった「法律の迷宮」を、誰もが迷わずに歩ける「デジタルな街」へと再構築する、リーガルエンジニアリングの土台となる技術である。

AI 時代における重要性

特に生成 AI（LLM）を活用する際、AI に誤った情報（ハルシネーション）を答えさせないためには、根拠となる正しい法令データを AI が読み取れる形で与える（RAG：検索拡張生成）必要がある。法情報に関する工学によって整備された高精度なデータは、「AI が嘘をつかないための教科書」としての役割を果たす。

B.2.2 法知識のモデリング（Legal Knowledge Modeling）

ミッション：「法律の『行間』を構造化し、コンピュータに正しい解釈を授ける」

法律の条文に含まれる抽象的な概念や言葉の定義、そしてそれらの相互関係を、コンピュータが誤解なく処理できる「知識の体系（モデル）」として再構築することをミッションとする。

概要：言葉の曖昧さを排し、論理的な意味を定義する

自然言語で書かれた法律には、人間には理解できてもコンピュータには判別できない「含み」や「文脈」が多く存在する。法知識のモデリングは、これらを工学的手法で整理する。

- **オントロジー（概念体系）の構築**: 「人」「組織」「権利」「義務」「禁止事項」といった法的要素を単なる単語としてではなく、階層構造や属性を持つ「オブジェクト」として定義する。例えば、「『所有者』とは『物』に対して『使用・収益・処分』の権利を持つ主体である」といった定義を、コンピュータが処理可能な形で記述する。
- **セマンティック・インターオペラビリティ（意味的相互運用性）の確保**: 同じ「住所」という言葉でも、税法上、民法上、住民基本台帳法上で微妙に意味が異なる場合がある。これらをシステム間で連携させる際、意味の食い違い（セマンティック・ギャップ）によるエラーを防ぐため、各法令間の意味的なマッピング（翻訳）を行う。
- **義務論理（Deontic Logic）の実装**: 法律の本質である「〜しなければならない（義務）」「〜してもよい（許容）」「〜してはならない（禁止）」という論理構造をモデル化する。これにより、ある行動が法的制約に対して「違反」なのか「適法」なのかを、システムが論理的に推論できるようになる。

活動内容

1. **法的概念の抽出と定義（ナレッジ・エンジニアリング）**：専門家とエンジニアが協力し、法令からキーとなる概念を抽出し、その属性（プロパティ）を厳密に定義。
2. **法規等情報間の整合性チェック**: 複数の法律にまたがる定義を比較し、論理的な矛盾や重複を特定。また、判例などとの整合性も確認。
3. **ナレッジグラフの構築**: 概念同士を「A は B の例外である」「A は B の一部である」といった関係性で結び、巨大な「意味のネットワーク」を構築。
4. **ビジネスルールへの変換**: モデリングされた法的知識を、企業の業務システムや AI がそのまま利用できる「ビジネスルール（実行可能なロジック）」の設計図へと変換。

解説：システムの『共通辞書』と『翻訳機』を作る技術

前項の「法情報に関する工学」が目的地を探すための「地図（GPS）」だとすれば、この「法知識のモデリング」は、その街で話される「言葉の定義を定めた共通の辞書」であり、かつ「高度な翻訳機」である。

人間同士でも「それはどういう意味で言ったのか？」と確認が必要なように、コンピュータも法律用語をそのまま渡されただけでは正しく動けない。

- **共通の物差し：**例えば「18 歳未満」という言葉を、「誕生日を迎えた瞬間」と判定するシステムと「その年の 4 月 1 日」と判定するシステムが混在するとトラブルが起きる。モデリングは、この判定基準を統一する。
- **言葉の裏側を教える：**「契約を結ぶ」という言葉の裏には、「申し込み」と「承諾」というステップが必要であることをコンピュータに教え込む。

これにより、異なる会社の AI や政府のシステムが連携しても、「法律の解釈」においてお互いに「話が通じない」という事態を避けることができる。

AI 時代における重要性

現在の生成 AI は、非常に滑らかな文章を作るが、時として法的概念を「それっぽく間違える（ハルシネーション）」という致命的な弱点がある。法知識のモデリングによって構築された厳密な「知識モデル」を AI に参照させることで、AI は単なる確率的な言葉の羅列ではなく、**「法的に正しいロジック」に基づいた回答**を出せるようになる。これは「信頼できる AI」を実現するための不可欠なガードレールとなる。

B.2.3 ルール・意思決定に関する工学（Rule & Decision Engineering）

ミッション：「法のロジックを『動く知能』へと変換し、公平・迅速な執行を自動化する」

法律や規制の中に含まれる「条件」と「結論」の論理構造を抽出し、コンピュータが判断可能なアルゴリズムとして実装することで、人手による判断の遅れやばらつきを排除し、社会全体のコンプライアンスを自動化することをミッションとする。

概要：曖昧な解釈から、厳密な実行コードへ

本領域は「Rules as Code (RaC：コードとしてのルール)」という世界的な潮流の中核を成す技術である。自然言語で書かれた「もし〜ならば」という規定を、エラーのない実行プログラムへと昇華させる。

- **論理構造のフォーマライゼーション（形式化）：**法律の条文を「入力データ（要件）」「判定ロジック（解釈）」「出力結果（効果）」に分解する。例えば、補助金の受給資格判定において、「所得制限」「家族構成」「居住地」などの変数を定義し、それらを論理式（If-Then ルール）として再構築する。
- **決定モデル表記法（DMN）の活用：**ビジネスの世界で標準的な DMN (Decision Model and Notation) などの視覚的な表記法を用いる。これにより、弁護士（法務）とエンジニアが同じ図表を見ながら、「このロジックは法解釈として正しいか」を確認し、そのまま実行コードへと変換できる環境を構築する。
- **形式検証（Formal Verification）による安全性担保：**実装されたルールが、「どのような入力に対しても矛盾した答えを出さないか」「法の抜け穴（デッドロック）がないか」を数学的な手法で検証する。これにより、大規模な制度変更時でも、予期せぬシステムの不具合を防ぐ。

活動内容

1. **判定ロジックの抽出（ロジック・マイニング）：**法令や業務マニュアルから、判断の分岐点となる条件（閾値や期間など）を漏れなく抽出。
2. **実行可能モデルの構築：**LegalRuleML[16]（法的ルール記述言語）や、OpenFisca のようなルールエンジンを用いて、計算可能なロジックを記述。
3. **適合性テストとデバッグ：**仮想のケーススタディを多数走らせ、人間が行う法解釈とシステムが出す結論が一致するかを確認。シミュレーションを行うとともに運用からのフィードバックを活用する。

16 OASIS, “LegalRuleML Core Specification Version 1.0 – OASIS Standard”, https://www.oasis-open.org/standard/legalruleml-core-specification-version-1-0-oasis-standard/ （参照日：2026-03-03）

4. **RaC API の提供：**実装されたルールを API として公開。これにより、民間企業は自社のアプリから「この取引は適法か？」を政府提供の公式ルールエンジンにリアルタイムで問い合せることが可能になる。

解説：法律を『スマートな自動判定機』にする技術

前項の「法知識のモデリング」が言葉の意味を教える「辞書」だとすれば、この「ルール・意思決定に関する工学」は、その辞書を使って実際に判断を下す「賢い自動判定機」を作る技術である。

例えば、行政の窓具とサービスをイメージすると、

- **これまでの法律：**「添付の証明書の期限が切れておらず、手数料が支払われれば手続きが可能」というルールが本に書いてあり、窓口職員が一人ひとりの申請書を見て判断している状態。
- **ルール・意思決定に関する工学：**そのルールを手続自動受付機のプログラムに書き込むこと。
    - **瞬時の判断：**誰が来ても、0.1 秒で「OK か NG か」を正確に判定する。
    - **不平等の解消：**「職員の気分」で申請の受付における判断のブレがなくなる。

つまり、複雑な法律の条件を「はい／いいえ」のスイッチの組み合わせに変え、誰でもボタンを押すだけで「法的に正しい答え」が即座に得られる仕組みを整える技術である。

AI 時代における重要性

現在の AI（生成 AI など）は「おそらくこうだろう」という確率的な推論は得意であるが、法律のような「1 円でも 1 日でも違えばアウト」という厳格な数値判断のためにあるわけではない。 ルール・意思決定に関する工学は、AI という「柔軟だが曖昧な頭脳」の横に、「絶対に間違えない論理の骨組み」を置く役割を果たす。AI ルールエンジンのルールを定義し、最終的な法的適合性をプログラムがチェックすることで、法的な正確性を100%保証することが可能になる。欧州の LDES（Linked Data Event Stream）は、そこを補完する仕組みである。

B.2.4 法プロセス工学（Legal Process Engineering）

ミッション：「法の執行を淀みのない『サービス』へと昇華させ、適正手続を自動で保証する」

法律が定める行政手続や権利行使のステップ（申請、審査、認可、不服申し立て等）を、ソフトウェア工学のワークフロー管理手法を用いて最適化し、誰もが迷わず、かつ不正や遅滞のない「透明なプロセスの自動走行」を実現することをミッションとする。

概要：静的な規定から、動的なワークフローへ

法律は「何をすべきか」を規定するが、「どう効率的に、漏れなく処理するか」という実装レベルの設計図は示さないことが一般的である。法プロセス工学は、条文の裏側にある「手続き」を工学的に再定義する。

- **プロセスのモデル化と標準化（BPMN の適用）：**ビジネスプロセス・モデリング表記法（BPMN）などの国際標準を用い、法的手続きを可視化する。「誰が」「いつ」「何のデータをもとに」判断し、次に「誰に」渡すのかという流れを、曖昧さを排して定義する。
- **ステート（状態）管理と例外処理の厳密化：**「審査中」「差し戻し」「許可」といった申請の状態をデジタルで厳密に管理する。特に、期限切れ（タイムアウト）への自動警告や、不服申し立て（エスカレーション）が発生した際のルートをあらかじめプログラムに組み込み、属人的な放置やミスを物理的に防ぐ。
- **エビデンス・バイ・デザイン（証拠の自動生成）：**プロセスが実行されるたびに、それが法に則って行われたことを証明するログ（監査証跡）を自動で生成・保存する。これにより、事後の監査や情報公開請求に対するコストを劇的に下げ、行政の透明性を高める。

活動内容

1. **手続きの再設計（リエンジニアリング）：**単に今の紙のフローをデジタル化するのではなく、デジタル技術を前提に、不要な添付書類の削減や並列審査を導入し、プロセスを最短化する。
2. **ケースマネジメント・システムの実装：**複雑で判断が分かれる個別の事案（ケース）に対し、過去の判断事例や関連法令をシステムがサジェストし、審査官の判断を強力に支援する環境を構築。
3. **API 連携による自動取得：**申請者に証明書を出させるのではなく、システムが他の行政データベースから自動で必要なデータ（住民票や納税証明等）を取得し、ユーザーの手間をゼロにする設計。
4. **不服申し立て・救済プロセスの実装：**決定に対して納得がいかない場合の「異議申し立て」ルートを最初からシステム内にボタン一つで組み込み、権利救済を容易にする。

解説：役所の窓口を『全自動のスマートな工場』に変える技術

前項の「ルール・意思決定に関する工学」が個別の「はい／いいえ」を判定する「センサー」だとすれば、この「法プロセス工学」は、部品が運ばれ、組み立てられ、検査を通って製品が出るまでの「工場全体のスマートなライン」を作る技術である。

例えば、申請をイメージすると、

- **これまでの手続き：**自分で書類を集め、窓口に行き、担当者がハンコを押し、別の担当者が確認し、数週間後にまた取りに行く。どこで作業が止まっているのか外からは見えない。
- **法プロセス工学：**手続き全体が「透明なパイプライン」になる。
    - **見える化：**今、自分の申請が「どこで」「誰が」確認中なのかがスマホでリアルタイムに分かる。
    - **渋滞の解消：**特定の部署で作業が溜まると、システムが自動で検知して助けを呼んだり、優先順位を上げたりする。
    - **証拠の保存：**「誰がこの認可を出したか」という記録が自動で残るため、後で「不当だ」と言われた時も、瞬時に正しい手続きだったことを証明できる。

つまり、複雑でブラックボックスになりがちな「行政の裏側」を、滞りなく、ミスなく、そして誰にでも開かれた「スムーズな自動ライン」へ変える技術である。

AI 時代における重要性

AI による自動決定（自動認可など）が増える中で、「なぜその結果になったのか」というプロセスが不透明になることへの懸念が高まっている。 法プロセス工学は、AI が判断を下すまでの「道のり」をしっかりと管理し、必要に応じて人間が介入（Human-in-the-loop）できるポイントを設計する。これにより、スピード感のある AI の活用と、人間社会が求める「適正な手続き（正当性）」を両立させることが可能になる。

B.2.5 法システムのアーキテクチャ（Legal System Architecture）

ミッション：「ガバナンスをシステムの『骨組み』に組み込み、適法性を構造的に保証する」

個別のルールや手続きをデジタル化する段階を超え、システム全体の設計思想（アーキテクチャ）そのものに法的規律を融合させることをミッションとする。これにより、後付けの対策ではなく、システムの動作原理そのものが法に準拠している「コンプライアンス・バイ・デザイン（Compliance by Design）」を実現する。

解説：点から面へ、ルールから構造へ

本領域は、法務の知見をエンジニアリングの最上流工程である「システム設計」に同期させる、リーガルエンジニアリングの司令塔とも言える分野である。

- **コンプライアンス・バイ・デザインの具現化：**プライバシー保護、公平性、説明責任といった抽象的な法的・倫理的要件を、システムのデータ保持ポリシー、認証認可の仕組み、処理ログの構造といった具体的な「技術的制約」へと落とし込む。こ

れにより、開発者が意識せずとも、システムが法を逸脱した動きをできないように制御する。

- **ポリシーと実行の分離（Policy-Execution Separation）：**「何が許されるか（ポリシー）」と「どう動くか（実行ロジック）」を分離したアーキテクチャを設計する。これにより、法改正があった際にシステム全体を作り直すのではなく、ポリシー層の設定を書き換えるだけで、システム全体の挙動を適法な状態へと同期させることが可能になる。
- **エコシステム全体の整合性管理（Interoperability Architecture）：**自社システムだけでなく、外部の API や政府の共通プラットフォームと連携する際の「法的責任の分界点」をアーキテクチャ図として定義する。データが組織をまたぐ際の同意管理や、責任の所在を技術的に追跡可能な構造を設計する。

活動内容

1. **法的要件の技術要件への翻訳（Requirement Mapping）：**複雑な規制要件（例：GDPR や AI 法）を、システムの非機能要件やデータ定義書へ直接マッピング。法令などの条文を ID で関連付けたりすることもある。
2. **リファレンス・アーキテクチャの策定：**特定の業界（金融、医療、公的サービス等）において、法を遵守するために最低限必要な機能ブロックとデータフローの標準モデルを設計。
3. **アクセス制御とデータガバナンスの設計：**「誰がどのデータに、どの法令根拠に基づいてアクセスできるか」をシステム基盤のレベルで制御する仕組みの実装。
4. **変更管理と検証戦略の立案：**法改正がシステムに与える影響範囲を瞬時に特定できるよう、法令コンポーネントとシステムモジュールの依存関係を管理。

解説：『後付けの鍵』ではなく『安全な建物の構造』を作る技術

前項の「法プロセス工学」がスムーズな「工場のライン」を作る技術だとすれば、この「法システムのアーキテクチャ」は、その工場や街全体の「安全で揺るぎない設計図」を作る技術である。

例えば、セキュリティ対策をイメージすると、

- **これまでのやり方：**ソフトウェアを作った後、公開直前に「法的に大丈夫か？」をチェックし、問題があれば慌てて修正したり、後から「防犯カメラ（監視ツール）」を取り付けたりする。これはコストが高く、抜け漏れも生じやすい方法である。
- **法システムのアーキテクチャ：**最初から「泥棒が物理的に入れない構造」を設計する。

- **構造的な安全：**そもそも個人情報を一箇所に集めない構造にする、あるいは特定の条件を満たさない限りデータがロック解除されない仕組みを、土台（アーキテクチャ）として組み込む。
- **後からの変更が楽：**部屋のレイアウト（ルール）が変わっても、建物の骨組みがしっかりしていれば、壁を動かす（設定変更）だけで対応できる。

つまり、法律を「守るべき面倒な外付けのルール」ではなく、「正しく安全に動くための設計の一部」として最初から組み込んでしまう技術である。

AI 時代における重要性

AI モデルはブラックボックス化しやすく、運用中に予期せぬ挙動を示すことがある。法システムのアーキテクチャは、AI という「予測困難なエンジン」を、法的に許容される「安全な檻（ガードレール・アーキテクチャ）」の中に閉じ込める役割を果たす。万が一 AI が誤った判断をしようとしても、周囲のシステム構造がそれを遮断し、法的リスクを最小化する。この「構造による信頼の確保」こそが、AI を大規模に社会実装するための大前提となる。

B.2.6 契約・トランザクションにおける工学（Contract & Transaction Engineering）

ミッション：「契約を『静的な文書』から『動的な実行ユニット』へと進化させ、合意と履行を直結させる」

契約という法的合意を、単なる文字の羅列ではなく、コンピュータが監視・実行可能な「データとロジックの集合体」として設計することをミッションとする。これにより、契約の締結から義務の履行、決済に至るまでをシームレスにつなぎ、取引コストの削減と信頼の自動化を実現する。

概要：合意（プロミス）を自動的な履行（アクション）へ

本領域は、契約法と分散型台帳技術（ブロックチェーン）や IoT、決済システムを融合させる分野である。

- **契約のデジタル・モデリング：**契約を「いつ（期限）」「誰が（主体）」「何を（義務）」「いくらで（対価）」といった属性を持つ構造化データとして定義する。条文をモジュール化（部品化）し、特定のイベント（例：商品の着荷、市場価格の変動）をトリガーにして、次のアクション（例：支払い、通知）が自動的に起動する仕組みを設計する。
- **スマート・コントラクトと法理の統合：**コードによって自動実行されるスマート・コントラクトを導入する際、それが民法や商法の原則（錯誤、解除、公序良俗な

ど）にどう適合するかを工学的に整理する。完全に自動化する部分と、人間の判断や法的介入が必要な部分を「ハイブリッド契約」としてアーキテクチャ化する。

- **ライフサイクル・トランザクション管理：**締結して終わりの契約ではなく、履行状況をリアルタイムで追跡する。例えば、サプライチェーンにおいて温度管理が契約条件にある場合、IoT センサーのデータが閾値を超えた瞬間に、自動的に賠償手続きや警告が発動するような「リアルタイムな義務管理」を可能にする。

活動内容

1. **契約条項のテンプレート・ライブラリ化：**標準的な契約条項をコード化し、開発者が「法的に検証済みの部品」を組み合わせて契約システムを構築できる環境を整備。
2. **イベント駆動型ロジックの実装：**外部データ（オラクル）を取り込み、条件合致を自動判定して決済や権利移転を実行するプロトコルの設計。
3. **紛争解決プロセスの組み込み：**自動実行に不服がある場合の「停止スイッチ」や、デジタル上での調停・仲裁プロセスへの自動連携機能の実装。
4. **トランザクションの監査証跡確保：**いつ、どのような条件で契約が実行されたかを改ざん不可能な形で記録し、法的証拠能力を持たせる設計。

解説：契約書を『賢い自動実行マシン』に変える技術

前項の「法システムのアーキテクチャ」が安全な「建物の構造」を作る技術だとすれば、この「契約・トランザクションにおける工学」は、その中で行われる約束を「条件が揃えば勝手に動く自動マシン」にする技術である。

例えば、荷物の配送契約をイメージすると、

- **これまでの契約：**紙の契約書を交わし、荷物が届いたら受領印をもらい、後日請求書を送り、銀行振り込みを確認する。それぞれの段階で人間がチェックし、ミスや遅れが発生する。
- **契約・トランザクションにおける工学：**契約そのものが「プログラム」になる。
  - **自動判定：**荷物が指定の場所に届いたことが GPS やセンサーで確認されると、システムが「履行完了」と即座に判断する。
  - **自動支払い：**「完了」と同時に、銀行や決済システムへ指示が飛び、その日のうちに支払いが完了する。
  - **トラブル対応：**もし配送中に温度が上がりすぎて中身がダメになったら、センサーがそれを検知し、即座に「お詫び」や「返金」の手続きが自動で始まる。

つまり、契約書を金庫にしまっておく「死んだ文書」から、24 時間 365 日、約束が守られているかを監視し、実行まで代行してくれる「生きているパートナー」に変える技術である。

AI 時代における重要性

AI がエージェントとして自律的に取引（株の売買、広告枠の入札、電力の融通など）を行う時代において、人間が全ての契約を目視で確認することは不可能である。契約・トランザクションにおける工学は、AI エージェントが活動する際の「法的枠組み（檻）」をプログラムとして定義する。AI が勝手に法外な契約を結んだり、義務を無視したりできないように、取引の「ルール」をシステム自体に埋め込むことで、超高速・大量の取引が行われる SDS 社会の安全性を担保する。

B.2.7 コンプライアンス・リスクに関する工学（Compliance & Risk Engineering）

ミッション：「法令遵守を『事後確認』から『リアルタイム管理』へ転換し、リスクを未然に防ぐ防護網を構築する」

絶え間なく変化する規制環境と、複雑化する企業のビジネス活動をデジタル技術で常に照らし合わせ、コンプライアンス（法令遵守）の状況をリアルタイムで可視化・制御することをミッションとする。これにより、「問題が起きてから対処する」のではなく、「リスクが顕在化する前に予兆を捉えて制御する」攻めのガバナンスを実現する。

概要：静的な監査から、動的なモニタリングへ

本領域は、LegalTech の核心であり、ソフトウェア開発における「オブザーバビリティ（観測可能性）」の概念を法務の世界に導入するものである。

- **規制情報の自動マッピング（Regulatory Mapping）：**日々更新される世界中の法令やガイドラインを「法情報に関する工学（領域 1）」から取得し、自社のビジネスルールやシステム設定（領域 5）と自動的に紐付ける。どの条文が自社のどの業務に対応しているかを常に最新状態に保つ。
- **継続的コンプライアンス・モニタリング（Continuous Compliance）：**一過性の監査ではなく、システムの稼働データや取引ログをリアルタイムで解析する。例えば、AI の判断が特定の属性に対してバイアス（偏り）を生じさせていないか、あるいは個人データの処理が同意の範囲を逸脱していないかを、システムが常時監視する。
- **リスク・スコアリングと自動アラート：**蓄積されたデータに基づき、コンプライアンス違反が発生する可能性を「リスクスコア」として算出する。異常値が検知され

た場合には、即座に担当者へ通知する、あるいはリスクの高い処理をシステムが自動で一時停止（サーキット・ブレーカー）する仕組みを構築する。

活動内容

1. **コントロール・フレームワークの設計：**法令要件を、システムで測定可能な「指標（KPI/KRI）」に変換し、監視項目を設定する。
2. **証跡（エビデンス）の自動収集：**監査時に必要な証拠資料を、システムが日々の運用の中で自動的に収集・整理し、いつでも提示できる状態（Audit-ready）にする。
3. **法改正の影響範囲分析（Impact Analysis）：**新しい規制が導入された際、自社のどのシステムコンポーネントに改修が必要かを、アーキテクチャ図（領域 5）と連携して即座に特定。
4. **AI ガバナンスの実装：**AI モデルの精度低下や、入力データの変化に伴う法的リスクの変動をモニタリングし、再学習やモデル停止の判断を支援。

解説：一年に一度の健康診断から、『スマートウォッチ』による常時監視へ

前節の「契約・トランザクションにおける工学」が約束を自動実行する「マシン」だとしたら、この「コンプライアンス・リスクに関する工学」は、企業やシステムの健康状態を常にチェックする「高機能なスマートウォッチ」のような技術である。

例えば、健康管理をイメージすると、

- **これまでのやり方：**一年に一度、人間ドック（年次監査）を受けて「異常なし」と判明しても、その翌日に病気になるリスクは残る。問題が見つかった時には、すでに手遅れ（大きな不祥事）になっていることも少なくない。
- **コンプライアンス・リスクに関する工学：**24 時間 365 日、脈拍や血圧（ビジネスデータ）を測り続ける。
    - **予兆を捉える：**「少し血圧が上がってきた（リスクが高まった）」という段階でアラートを出し、大きな病気になる前に生活習慣（業務プロセス）を改善させる。
    - **常に最新の医療知識：**医療ガイドライン（法改正）が新しくなれば、スマートウォッチの判定基準も自動でアップデートされ、新しい基準で健康かどうかを判定し続ける。

つまり、コンプライアンスを「年に一度の苦労する行事」から、システムが自動で守ってくれる「社会的な予防システム」へと変える技術である。

AI 時代における重要性

AI は、従来のソフトウェアと異なり、入力されるデータによって挙動が刻々と変化（ドリフト）する。そのため、開発時に「適法」だった AI が、半年後には「差別的」な判断を下すリスクがある。コンプライアンス・リスクに関する工学は、この AI の「揺らぎ」をリアルタイムで監視し、法的な許容範囲（ガードレール）を外れそうになった瞬間にブレーキをかける。この「動的なリスク管理」があって初めて、企業は予測不能な AI 技術を、自信を持ってビジネスの核に据えることが可能になる。

B.2.8 AI 支援型リーガルエンジニアリング（AI-Assisted Legal Engineering）

ミッション：「AI と人間の協調により、法務の生産性を極限まで高め、複雑すぎる規制を人間が制御可能なサイズに解体する」

生成 AI を中心とした先端技術を、単なる「文章作成の代行」としてではなく、法的な論理構築、複雑な規制の構造分析、そして将来のリスク予測を支援する「エンジニアリング・ツール」として活用することをミッションとする。

概要：AI を「執筆者」から「思考の増幅器」へ

本領域は、従来のリーガルテックが苦手としていた「非構造化データ（自由な文章）」の処理に、最新の大型言語モデル（LLM）を適用する分野である。

- **根拠に基づいた論理起草（Augmented Drafting）：**AI が単にそれらしい文章を作るのではなく、指定された法令データ（領域 1）や知識モデル（領域 2）を「参照（RAG：検索拡張生成）」しながら、法的根拠（ソース）を明示したドラフトを作成する。これにより、AI の弱点である「ハルシネーション（もっともらしい嘘）」を徹底的に排除した、信頼性の高いドキュメント作成が可能になる。
- **多層的な規制影響の自動解析（Regulatory Impact Analysis）：**数千ページに及ぶ国内外の規制文書から、自社の特定の事業領域に影響を与える箇所を自動で抽出・要約し、既存の業務ルール（領域 3）やプロセス（領域 4）との「差分」を特定する。
- **シナリオ・シミュレーション（Legal Simulation）：**「もしこの法改正が行われた場合、自社のどの取引がどの程度のリスクにさらされるか」といった仮定の質問に対し、過去のデータと法理を組み合わせて、多数のシナリオをシミュレーションし、最適な対策をサジェストする。

活動内容

1. **リーガル・エージェントの構築：**特定の法域や業務に特化し、法務担当者と対話しながら「リサーチ、起草、レビュー」を完結させる専用 AI エージェントの設計。

2. **プロンプト・エンジニアリングの法務最適化：**法的な論理展開（三段論法など）を AI に正確に実行させるための、高度な指示文（プロンプト）の体系化。
3. **マルチモーダルな法務分析：**テキストだけでなく、契約書の署名、企業の組織図、複雑な取引スキーム図などを画像として読み取り、矛盾やリスクを分析。
4. **ヒューマン・イン・ザ・ループ（Human-in-the-loop）の設計：**AI の判断を人間が最終確認・修正し、その修正内容を AI が学習して精度を高める、人間と AI の共同作業フローの構築。

解説：膨大な法律の迷宮を飛び越える『専属のスーパー秘書』

前項の「コンプライアンス・リスクに関する工学」が 24 時間健康を監視する「スマートウォッチ」だとしたら、この「AI 支援型リーガルエンジニアリング」は、あらゆる法律に精通し、一瞬で数万ページの資料を読んで答えを出す「超人的な記憶力と分析力を持つスーパー秘書」である。

例えば、新しい海外ビジネスを始める場面をイメージすると、

- **これまでのやり方：**弁護士や法務担当者が、現地の法律を数週間かけて調べ、何百枚もの資料を読み、一つずつリスクを洗い出す。時間もコストも膨大にかかる。
- **AI 支援型リーガルエンジニアリング**:
  - **一瞬でリサーチ：**AI が世界中の最新規制を数秒で読み込み、「このビジネスをやるなら、この 3 つの法律に注意が必要である」と根拠付きで提示する。
  - **下書きの作成：**「その法律を守るための社内規程を書いて」と頼めば、過去の自社のルールと矛盾しない、完璧な下書きを数分で用意してくれる。
  - **相談相手：**「このプランで行くと、後でどんな揉め事が起きそう？」と聞けば、過去の裁判例から起こりうるリスクを予測して教えてくれる。

つまり、人間が「探す・書く・比べる」という単純作業から解放され、**「判断する・決断する」という最も高度な創造的業務に集中できる**ようにする技術である。

AI 時代における重要性

現代の規制はあまりに複雑になり、もはや人間一人の脳で全てを把握することは不可能である。 AI 支援型リーガルエンジニアリングは、この「人間の認知限界」を突破するための武器である。ただし、AI を盲信するのではなく、リーガルエンジニアリングの他の領域（領域 3 の論理チェックや領域 9 の信頼性確保）と組み合わせることで、**「スピード」と「正確性」の両立**が可能になる。これにより、日本企業はグローバルな規制の荒波を、誰よりも速く、かつ安全に航行できるようになる。

B.2.9 ガバナンス・信頼に関する工学 （Governance & Trust Engineering）

ミッション： 「システムの意思決定に『法の支配』と『正当性』を宿し、社会が安心してテクノロジーに身を委ねられる信頼の基盤を構築する」

自動化されたシステムや AI による判断が、憲法・法律・倫理的原則に照らして正当であることを工学的に証明し、万が一の誤りに対しては救済や修正の手段を保証することで、社会的な信頼（Trust）を確立することをミッションとする。

解説：ブラックボックスから、説明可能なガバナンスへ

本領域は、技術的な「正しさ」を社会的な「正当性」へと変換する、リーガルエンジニアリングの最上位レイヤーである。

- **トレーサビリティの確保（Law-to-Decision Traceability）：**「法律（条文）」→「論理モデル（ルール）」→「実行コード（アルゴリズム）」→「個別の判断結果（アウトプット）」という一連の流れを数珠つなぎに記録する。これにより、ある判断がどの法律のどの条文に基づいて行われたのかを、後から誰でも（あるいは監査システムが）検証可能にする。
- **説明可能な AI（XAI）と法的根拠の統合：**AI の複雑な計算結果に対し、「なぜその結論に至ったか」を人間が理解できる言葉（法的ロジック）で再構成する。単に統計的な確率を示すのではなく、「この要件を満たさなかったため、不許可となった」という法的な理由付けを自動生成する。
- **異議申し立てとオーバーライド（救済措置）の設計：**システムによる自動決定に対して、人間が異議を唱える権利を技術的に保障する。人間が介在して判断を覆す（オーバーライド）プロセスを最初から組み込み、その修正履歴もまた「正しい手続き」の一部として記録する。

活動内容

1. **説明責任（Accountability）のフレームワーク設計：**システムが社会に対して負うべき説明の範囲とレベルを定義し、それを実現するためのデータログ構造を設計する。
2. **アルゴリズム監査の自動化：**第三者機関や規制当局が、システムの内部ロジックを安全かつ客観的にテスト・検証できるインターフェースの構築。
3. **バイアス・公平性の定量的評価：**特定の属性（人種、性別、年齢等）に対して不当な差別が生じていないかを、統計学的・法学的な指標を用いて常時測定する。
4. **トラスト・アーキテクチャの構築：**ゼロトラスト（何も信じない）の考え方をベースに、改ざん不可能なログ（ブロックチェーン等）を用いて、プロセスの正当性を客観的に証明する仕組みの実装。

解説：AI の判断に『納得感の判明』を与える技術

これまでの「ルール・意思決定に関する工学」が自動で答えを出す「計算機」だとすれば、この「ガバナンス・信頼に関する工学 」は、その計算機が「本当に正しいルールで動いているかを見張り、納得のいく説明をさせる」ための技術である。

例えば、銀行のローン審査をイメージすると、

- **これまでの AI：**「あなたのローンは不可である」とだけ言われ、理由はブラックボックス。「AI がそう判断したから」では、誰も納得できないし、間違いがあっても直せない。
- **ガバナンス・信頼に関する工学 ：**AI が判断を下す横で、常に証拠を揃える。
  - **理由の明示：**「あなたの年収が規定の〇〇円に届かなかったため、第〇条の基準に基づき否決されました」と、人間にも分かる理由を教えてくれる。
  - **間違いの修正：**もし入力データが間違っていたら、「ここが違う」と指摘して再審査させるルートが最初から用意されている。
  - **公平性の証明：**「この AI は、特定の地域や性別で差別をしていない」という証拠を常に公開し、社会全体の信頼を得る。

つまり、テクノロジーを「得体の知れない怖いもの」から、「ルール通りに動き、いつでも理由を説明してくれる誠実な代理人」に変える技術である。

AI 時代における重要性

AI が社会の至る所で意思決定を下す「Software-Defined Society（SDS）」では、一度のアルゴリズムのバグやバイアスが、数万人、数百万人に対して一瞬で不利益を及ぼすリスクがある。ガバナンス・信頼に関する工学 は、この巨大な力を持つテクノロジーに「ブレーキ」と「説明書」を取り付ける役割を果たす。この領域が確立されて初めて、私たちは AI に社会の重要な機能を委ねることができ、ひいては「技術を信じ、安心して新しいサービスを利用できる社会」を実現することが可能になる。

以上の 9 つの領域は独立しているのではなく、相互に連携することで「社会の OS」を構築する。「法」という社会の意思を、「工学」という実行の力で具現化する。このリーガルエンジニアリング（LE）の体系こそが、日本が再び世界の技術競争において主導権を握り、かつ人間中心の豊かなデジタル社会を築くための羅針盤となるのではないか。