

文部科学省委託

## 専修学校による地域産業中核的人材養成事業

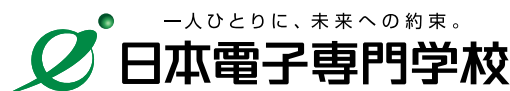
人口減少地域の職業人材を確保するための専修学校振興プログラム

人口減少地域を支える

## AI補完型ITエンジニア養成プログラムの開発

令和7年度

# 事業報告書



一人ひとりに、未来への約束。

日本電子専門学校

```
def develop_training_program(  
    role="AI-Supported IT Engineer",  
    *,  
    context="Population-Declining Regions",  
):  
    return role
```



## まえがき

生成AIは今日、文章の生成や要約の域を超え、情報システム開発における設計から運用まで広範の工程に浸透しつつある。

他方で、人口の減少と人材の都市部への集中は、地方企業におけるIT人材不足を構造的に深刻化させている。技術の進展が速くなるほど、外部委託に依存したままではその更新が追い付かず、事業継続の基盤になる情報システムや業務プロセスの改善は停滞し易い。

生成AIは、このような状況を緩和し得る一方で、誤情報の混入、品質の低下、セキュリティ上の欠陥等、新たなリスクも同時に顕在化させる。

そのため、いま求められるのは、AIを“代替者”としてでなく、あくまで“補完的な道具”として制御し、その生成物の妥当性を人が評価した上で、要件・品質・運用を統合して前進させられる専門人材の体系的な養成である。

本書は、学校法人電子学園 日本電子専門学校（以下「当校」）が本年度（令和7年度）に実施した文部科学省委託専修学校による地域産業中核的人材養成事業（人口減少地域の職業人材を確保するための専修学校振興プログラム）「人口減少地域を支えるAI補完型ITエンジニア養成プログラムの開発」における活動を取り纏め、その成果と今後の展開可能性を提示するものである。

本事業で養成を目指す「AI補完型ITエンジニア」は、生成AIの出力を鵜呑みにせず、要件定義から実装、運用自動化までを自律的に遂行し、速度と品質のトレードオフを実務として統制できる高度専門職である。

地方IT企業における開発規模の小ささや担当者の少なさ、外部依存の制約等を踏まえれば、その少数精鋭の開発体制にAIを組み込み、短サイクルで改善を積み重ねられる能力こそ、実務として現場に生きる競争力になる。

この人材像を具体化するべく、本事業では〈教育→評価→地域振興〉を一体の仕組（教育モデル）として設計した。その中核に据えたのが「AI駆動ITスキル指標」（BASIS：Benchmark for AI-Supported IT Skills）である。

BASISは、AIの操作技能に留まらず、AI補完型ITエンジニアの職務における各種の判断や統制を体系的なスキルとして整理する。そして、教育プラットフォーム上の「GPT-TA評価システム」（生成AIを活用した指導・評価支援）と連動し、学修成果の可視化と教育の質保証・継続的改善に繋げる。

人口減少地域における人材の養成と配置を阻害する最大の制約は、学びの機会と教育資源の偏在である。

本事業では、この制約を前提に、講義配信と演習指導を一体的に提供する“クラウド教育基盤”を構想し、地域差を縮減しながら同質の教育を提供し得る諸条件を整備した。開発する教育プログラムも、講義のみならず演習・実習・PBLを段階的に組み込み、〈理解→実装→改善〉の循環が回るように設計した。そして、教育成果が地域企業の課題解決と人材定着に連鎖するよう、各種支援との接続も視野に入れた。

本年度は、その3年事業の初年度として、教育プログラムの基礎設計と初期教材の開発を主軸に据えた。具体的には、地域・企業・学習者の実態を調査し、その結果を踏まえてBASISや評価システムの設計、教育プラットフォームの要件定義、カリキュラムとシラバスの整備、教育コンテンツの制作を進めた。そして、次年度に実施する短期実証に接続できるよう、運用を見据えた初期セットまで具体化した。

本事業は、単に“AIを使える人”を増やすことを目的にするのでない。生成AIが普及するほど、出力の事実性・一貫性・再現性を点検し、根拠に照合し、必要に応じて判断を留保・調整する能力が、組織の信頼性を左右する。そして、AI活用は開発速度を上げ得るが、同時に欠陥や脆弱性の混入リスクも高め得るため、AI補完型ITエンジニアは、品質と説明責任を担保する統制の作法を“技能”として体得する必要がある。

BASISが立脚する「AIは補完ツール、評価と最適化は人」という役割意識は、技術の進歩に過度に依存せず、“責任ある実装”と説明可能性を担保するための設計思想である。これを教育プログラムに落とし込むためには、拙速にツール導入やコンテンツ整備に走るのではなく、教育の内容と評価の枠組を先に整え、データに基づく改善ループを組み込む必要がある。そうすることで教育プログラムは、技術変化やツール更新に翻弄されず、到達目標と評価基準（何を教え、何を以て“できた”とするか）という軸を保ったまま、必要な変化だけを選別して取り込み、継続的に質を高められる。それこそ、AI時代の職業教育に求められる在り方であろう。

本事業は、専修学校の振興という点でも重要である。専修学校は、地域企業と近接し、実務に即した教育を短い更新周期で設計し得る機動力を有する。そこに、遠隔配信とクラウド環境を組み合わせることで、教育資源の集中と地域間格差という構造問題に対し、現実的な解を提示できる。そして、BASISとGPT-TA評価システムによって学修成果を可視化し、教育改善と人材マッチングに接続する仕組は、専修学校が“学びの提供者”に留まらず“人材の品質保証と供給を担う中核機関”として機能する可能性を拓ける。本事業で蓄積される設計知・運用知と教材群が参画機関で共有・展開されれば、専修学校がAI時代の職業教育を牽引する共通基盤になり得る。

人口減少は不可逆の趨勢であり、地方の現場は今後さらに少人数で多様な業務を担う。その現実を直視すれば、技術を地域に持ち込み、地域で使い熟し、改善を継続する力を養成することこそ、最も確実な将来投資になる。AI補完型ITエンジニアは、その投資を具体的な実装に変換する媒介である。

学習者が地域に居ながら先進の方法論を学び、地域企業の課題を解決し、その成果が次の雇用と次の学びを生む循環が立ち上がるのであれば、人口減少地域においても“技術の進展が地域の活力を支える”状態を実現できる。本事業は、その転換を微力ながら確実に始動させる試みである。

最後に、本事業にご参画いただいた教育機関・企業・団体、そして調査・開発等にご協力いただいた関係者各位に謝意を表す。

今後も本年度の成果を踏まえ、教育モデルと教育プログラムの成熟・洗練、実証の深化、全国展開に向けた統合・普及を着実に進めてゆく所存である。

令和8年2月

「人口減少地域を支えるAI補完型ITエンジニア養成プログラムの開発」事業推進委員会  
(代表機関：学校法人電子学園 日本電子専門学校)

# 目 次

まえがき	i
目 次	iii
<b>1 事業概要</b>	
1.1 事業趣旨	1
1.2 事業背景	2
1.3 事業計画	6
1.3.1 全体スキーム	6
1.3.2 年次計画	8
1.4 事業推進	9
1.4.1 実施内容	9
1.4.2 推進体制	10
1.4.3 事業推進委員会	12
<b>2 調査報告</b>	
2.1 調査概要	13
2.2 調査① 地方企業生成AI活用実態調査	14
2.2.1 調査仕様	14
2.2.2 調査内容	14
2.2.3 調査結果	18
2.2.4 調査考察	26
2.3 調査② IT “学び直し” ニーズ実態調査	28
2.3.1 調査仕様	28
2.3.2 調査内容	28
2.3.3 調査結果	31
2.3.4 調査考察	37

2.4 調査③ 生成AIコーディングアシスタント事例調査	39
2.4.1 調査仕様	39
2.4.2 調査結果	39
2.4.3 調査考察	60
2.5 調査④ AI活用エンジニアリング教育事例調査	62
2.5.1 調査仕様	62
2.5.2 調査結果	62
2.5.3 調査考察	87

### 3 開発報告

3.1 開発概要	91
3.2 教育モデルの構造	93
教育モデルの目標—AI駆動ITスキル指標 / 教育モデルの内部機構—教育システムと教育プログラム /	
教育プログラムの体系—カリキュラムとシラバス / 教育プログラムの内容—教育5形態と教育コンテンツ /	
教育プログラムの基盤—教育プラットフォームとGPT-TA評価システム	
3.3 教育目標—AI駆動ITスキル指標 (BASIS)	96
3.3.1 BASIS設計過程	96
3.3.2 BASIS枠組	98
3.3.3 BASISと既存標準の対照評価	101
3.4 教育体系—教育プログラム	109
3.4.1 設計方針—BASISからカリキュラムへ	109
3.4.2 カリキュラム—AI補完型ITエンジニア養成プログラム	111
3.4.3 シラバス—各コマ指導計画	115
3.5 教育素材—教育コンテンツ	124
3.5.1 開発方針	124
3.5.2 講義コンテンツ—講義動画・講義資料	125
3.5.3 演習コンテンツ—演習課題セット	128
3.5.4 自習コンテンツ—理解度確認CBT問題	130
3.6 教育基盤—教育プログラムの運用	132
3.6.1 設計方針と取組範囲	133
3.6.2 教育プラットフォーム—提供・管理・記録の統合基盤	133
3.6.3 GPT-TA評価システム—生成AIを活用した教育支援機構	138

## 4 事業総括

4.1 総括の型——CIPP+OR .....	143
4.2 Context——論点と課題 .....	144
4.3 I n p u t——施策と投入 .....	144
4.4 Process——調査・設計・開発の連結 .....	145
4.5 Product——成果と評価 .....	146
4.6 Outreach——普及・共有に向けた外部発信 .....	147
4.7 Roadmap——次年度以降の実装と成熟化 .....	148

## 5 別掲資料

5.1 教育コンテンツ——講義資料 .....	151
5.2 教育コンテンツ——演習課題 .....	175
5.3 教育コンテンツ——自習問題 .....	177
5.4 教育プラットフォーム——設計書 .....	180
5.5 GPT-TA評価システム——設計書 .....	189



# 事業概要

## 1.1 事業趣旨

少子高齢化に伴う人口減少でIT人材が不足・流出し、地方企業の情報基盤は更新が滞っている。そのため、地域産業を支えるサプライチェーンも技術的停滞の連鎖に陥っている。そして、生成AIの急速な商用化が開発の速度・品質の新基準を生んだことで、地方企業は対応余力を失いつつある。

かような状況下で求められるのは、生成AIの力を実務に応用し、情報システムの要件定義から運用改善まで継続的に遂行・主導できる**次世代型IT人材（AI補完型ITエンジニア）**である。

本事業では、専修学校をその人材養成拠点にし、教育・評価・地域振興を一体化した**教育モデル**を構築する。そこでは、遠隔講義等を展開するクラウド環境を教育基盤にし、生成AI評価システムとAI駆動ITスキル指標で学修の到達度を可視化する。企業のDX課題を収集してプロジェクト型学習に取り込み、その成果物で企業課題を解決する。修了後メンタリングで人材の地域定着を支援し、多層的な人材養成を実現する。これにより、若者が地方に居ながら“いま学びたい最新技術”にアクセスできる環境を整備し、地元人材の域外流出を抑えつつUターン人材の流入を促す。

そして、この教育モデルのコアとして開発する**教育プログラム**では、生成AIリテラシー・プロンプト設計・AI補完型アジャイル開発・AIOps等、従来のIT教育にない領域を体系化し、〈講義→演習→実習→PBL〉を段階的に配置することで、学習者が“学びながら創り、創りながら学ぶ”**実践的教育**を展開する。

かくして、本事業は、**地方企業の内製力を底上げするDXの中核を担う人材**を循環的に供給し、地域産業が技術変革を自律的に継続できる基盤を構築する。それは延いては、人口減少地域において産業競争力と質の高い学習機会・雇用機会を持続的に創出し、地域社会のレジリエンスと共創的イノベーションを促進するものである。

## 1.2 事業背景

### 人口減少×AI急伸という重圧の下で地方が選ぶべき道

日本の多くの自治体は現在、急速な少子高齢化の渦中にある。2050年には市区町村の7割超で生産年齢（15～64歳）人口比率が50%を下回り、“働き手半減社会”が常態化すると見通される<sup>★01</sup>。若者が都市に流出すれば、地方の教育インフラは縮小し、その空洞化が人口流出を加速する。この負のスパイラルを断ち切る鍵は、**地域に残るインセンティブを教育と雇用の両面で同時に創出**することにある。

他方で、生成AI技術やクラウド協働基盤が急伸する今日、“距離の壁”は大きな制約でなくなりつつある。そこで、地方企業が自社データを駆動源にしてビジネスを変革する場合、**AIを有効に活用しながら情報システムを高速かつ高品質に内製できる人材**が今後ますます必要とされる。

そして、そのような**次世代型IT人材を地方圏で養成できるような教育モデルの構築**も求められる。

★01 国立社会保障・人口問題研究所『日本の地域別将来推計人口：令和5（2023）年推計 概要』（2023年12月）

### 生成AI市場の爆発的成長——8000億円の“青い海”

日本の生成AI市場は、2024年に1016億円に達し、2028年に約8倍の8028億円規模に膨むと予測される<sup>★02</sup>。そして、年平均成長率84.4%——これは“半導体バブル”期を凌駕する勢いである。参入障壁が急速に下がり、地方企業でもスマート農業・観光DX・多言語接客ボット等の多様な案件が顕在化している。市場のブルーオーシャンは都市に限定されず、地方にも波及している。しかし、これをキャッシュフローに換えるためには、**AIが出力した成果物を精査して業務案件に落とし込めるエンジニア**が不可欠である。

★02 IDC Japan株式会社『IDC Worldwide AI and Generative AI Spending Guide』（2024年8月）

### 量・質共に拡大する人材ギャップ——45～79万人の“壁”

日本のIT人材は2030年に45～79万人不足すると予測され、その不足分の4割超がクラウド・AI・データ利活用分野である<sup>★03</sup>。都市部の大企業さえ確保に苦心する高度スキル人材を、人口減少地域の中小企業が調達・雇用するのはほぼ不可能である。そこで求められるのが、**地方で育てて地方で働く次世代型IT人材**の供給である。

★03 みずほ情報総研株式会社（経済産業省委託事業）『IT人材需給に関する調査 調査報告書』（2019年3月）

### 生成AIの諸刃性——生産性55%向上 vs. 品質リスク

生成AIを組み込んだコーディングアシスタント（GitHub Copilot等）は、開発者の生産性をどの程度高めるか、

他方でその支援下で書かれたコードの品質はどの程度のものか——その実証データが相次いで報告されている。生産性について、Copilot利用者はタスク完了時間を平均55%短縮し、自己申告ベースの心理的負荷（精神的疲労）も有意に軽減したことが確認された<sup>★04</sup>。品質について、AI支援下で書かれたコードはバグやセキュリティ欠陥の混入率が高いとの副作用が提示された<sup>★05</sup>。つまり、AI支援コーディングでは、速度・開発者体験の向上と品質低下リスクが同時に顕在化するため、速度と品質のトレードオフを制御する運用設計が不可欠である。したがって、「プロンプト設計・選択眼」「テスト自動化・静的解析による品質査定」「業務KPIを考慮した実装判断」——このようなスキルを複合的に備えた人材がいなければ、AI導入は“生産性ブースター”どころか、逆に“リスク増幅装置”になり得る。そのため、企業や業界は、**AIコーディングアシスタントを駆使する仕組みと人材養成をワンセットで整備**する必要がある。

★04 Peng・Kalliamvakou・Cihon・Demirer 「The Impact of AI on Developer Productivity: Evidence from GitHub Copilot」『arXiv』（2023年2月）

★05 Perry・Srivastava・Kumar・Boneh 「Do Users Write More Insecure Code with AI Assistants?」『arXiv』（2022年11月）

## 政策の後押し——地域定着型デジタル人材の育成という国家課題

デジタル庁は、地方でデジタル人材を育成し、その活躍機会を地域で創出する仕組みを政府横断で推進すると言う<sup>★06</sup>。その際に参照されるIPA「デジタルスキル標準」（DSS）はDX推進に必要なリテラシーと専門職能を体系化した国家レベルのデジタル人材指針<sup>★07</sup>であるが、ここまでの考察で導かれる本事業の狙いは、DSSで提示された職能要件の全類型<sup>★08</sup>を一律に充すことでない。むしろ従来型IT人材のアップスキル（既存職能の高度化）を基軸にし、**生成AIを駆使して情報システム開発の速度と質を飛躍的に向上させた次世代型IT人材を創出**する点に特徴がある。ここに、各種政策が求めるデジタル化の裾野拡大と、地域企業が切実に欲する実装力の高い開発者という双方のニーズを橋渡しする形で、政府・産業界・教育機関の三者の目標が合流する。

★06 デジタル庁「デジタル社会の実現に向けた重点計画 本文」（2024年6月21日 閣議決定）

★07 独立行政法人情報処理推進機構「デジタルスキル標準（DSS）——デジタル社会を支える人材育成の指針」『情報処理推進機構』（2022年12月）

★08 「ビジネスアーキテクト」「プロダクトマネージャー」「UX/UIデザイナー」「データサイエンティスト」「AIエンジニア」「データエンジニア」「ソフトウェアエンジニア」「クラウドプラットフォームエンジニア」「サイバーセキュリティエンジニア」「ITアーキテクト（DXアーキテクト）」の10類型

## 専修学校という“機動力あるフロンティア”

大学教育は研究志向、企業OJTは現場志向——その狭間で専修学校は、最新技術の授業化を短期間で実施できる制度的柔軟性と、地域企業と直結したプロジェクト学習を設計・実現できる機動力を持つ。

具体的には、遠隔スタジオから都市部の専門家講義をライブ配信し、現役エンジニアを非常勤教員として招聘できる。また、カリキュラムに地方企業の課題解決を扱うPBLを組み込み、その成果物を業務システムとして納品する枠組等も整え易い。

かように、専修学校は、その制度的柔軟性と地域企業との密接な連携を活し、技術革新の速度とカリキュラム更新の速度を同期できる数少ない教育機関の一つである。

## 目指す人材像と地方ニーズ——AI補完型ITエンジニア

本事業で目指すのは、既存のIT基礎を備えた人材が**生成AIを駆使して情報システム開発のプロセスを高速化・高品質化できる「AI補完型ITエンジニア」<sup>★09</sup>を養成する教育モデルの構築**である。この人材の養成により、東京の専修学校が**地方企業の開発力向上と地方のデジタル化促進**に寄与する。

この開発プロセスにおいて——

- 要件定義フェーズでは、自然言語で整理したビジネス要件を高精度プロンプトに変換し、AIが生成する設計案を速やかに評価・採択する。
- 実装フェーズでは、AI補完型コーディングと自動テストを組み合わせ、リリースサイクルを従来比50%以下に短縮する。
- 運用フェーズでは、ログデータからAIが示す改善策案を活用し、業務KPIを継続的に更新する。

このようなスキルセットは、開発規模が小さく人的リソースに限られる地方企業にこそ効果が高い。例えば、10人規模の製造業が在庫管理システムを内製する場合、従来は外注コストと3～6箇月の開発期間を要したが、AI補完型ITエンジニアがいれば、1～2箇月でプロトタイプを実装し、保守も社内で完結できる。つまり、人口減少地域でこそ、この“少数精鋭+AI”の戦略的優位性<sup>★10</sup>が顕在化するのである。

★09 本事業では、そこで養成を目指す「生成AIを駆使して情報システム開発の速度と質を飛躍的に向上させた次世代型IT人材」を「AI補完型ITエンジニア」と名付けることにした。この人材の“次世代型IT人材”という位置付けは、昨今の「DX人材/デジタル人材」が組織変革・価値創出に軸を移した“ビジネス牽引者”という意味で巷間語られがちであるのに対し、「AI補完型ITエンジニア」はむしろ従来型IT人材の軸で技術を深耕・先鋭化した“高度専門職”である点を強調したものである。

★10 システム開発にAIを導入している企業はまだ少数派である。「試行中・検討中」も含めて漸く、AI導入を意識している企業は、ベンダー企業で約4割、ユーザー企業で約2割である。つまり、それ以外はAI導入について「未対応」「わからない」であった。  
(独立行政法人情報処理推進機構「2024年度ソフトウェア動向調査 簡易分析レポート」2025年4月)

## 多重の波及効果——企業・学修者・自治体を繋ぐ価値連鎖

都市部の専修学校がこのAI補完型ITエンジニアを養成できれば、地方企業は、外注依存を低減し、保守コストとリードタイムを縮減できる。学修者は、AI運用スキルによる賃金プレミアム（10～20%程度）<sup>★11</sup>を地元で享受できる可能性が増し、Uターン/Iターン志向を強める。そして、自治体は、産業誘致と若年定住策を一体で進められ、地域経済と財政の両面を安定化できる。

こうして**学び・働き・暮しが相互に価値を生む地域エコシステム**として機能し、人口減少下でも地域が持続可能性を保ち得る。

★11 OECD（経済協力開発機構）「OECD Employment Outlook 2023 : Artificial Intelligence and the Labour Market」『OECD Employment Outlook 2023』（2023年7月）、PwC（PricewaterhouseCoopers）「2024 AI Jobs Barometer」『PwC Global Artificial Intelligence Practice』（2024年2月）等

## 専修学校×生成AIで切り拓く地方創生サイクル——学び・実装・定着の循環モデル

生成AI市場の拡大、45～79万人の人材不足、そして政府が掲げるデジタル人材の養成・定着——この歴史的とも言える三潮流が交錯する今日、地方は“待つ側”でなく“創る側”に回るべき時を迎えている。

人口減少と都市部への人材集中により、地方はAI・クラウド人材を確保できず、産業の存続さえ危ぶまれる。そのような中、都市部の専修学校が生成AI・DevOpsを組み込んだ教育プログラムを設計し、**遠隔配信で全国の学習者を結ぶ教育モデル**を構築することには、大きな意義がある。

そこで、本事業では、専修学校の機動力を以て、**生成AIを武器にした次世代型IT人材が地方から継続的に輩出する教育モデルと社会システム**の設計図を提示する。具体的には――

- 地方の学生・社会人が移住せずに最先端スキルを習得でき、流出を防ぐ。
- 現役エンジニアを非常勤講師として招き、教育内容を産業の最前線に同期する。
- 学習成果が地方企業の実装プロジェクトに直結し、学び・実装・定着の循環を生む。

この仕組は、人材不足・産業DX・一極集中という三重の課題を同時に解決し得る戦略的投資になり、**人口減少社会における地域産業の“第二の地場革命”**<sup>★12</sup>を実現するインフラ投資にもなる。

かくして、技術転換の臨界点を逃さず、学びを残し、価値を創り、人を根付かせる――その戦略的行動こそ、地方が未来を選び取る最短経路になると言えよう。

★12 本稿では、昭和後期～平成初期の“産地活性化・高附加価値化”運動を「第一次地場革命」と捉えた上で、令和時代の“デジタル・AIを梃子にした地域産業刷新”を「第二の地場革命」と言うことにした。

## 1.3 事業計画

### 1.3.1 全体スキーム

本事業は、生成AIを駆使して情報システム開発の速度と質を飛躍的に向上させた次世代型IT人材（AI補完型ITエンジニア）を人口減少地域において持続的に養成し、地域産業のDXを自走させる基盤を構築することを目的とするものである。

そこでは、専修学校が中核になってAI駆動ITスキル指標（BASIS：Benchmark for AI-Supported IT Skills）＊を策定し、これを“礎”にした教育プログラムを開発する（「AI駆動ITスキル指標」は3.4、教育プログラムは3.5で詳説する）。

＊BASIS（AI駆動ITスキル指標）は、本事業で養成を目指すAI補完型ITエンジニアのスキル要件を定義し、その修得度を評価するための指標である。この人材が0.5～1年で“要件定義→実装→運用自動化”を自走できるよう、汎用ITスキルをAIで高速・高精度化する観点で再編する。そして、「AIはツール、評価と最適化は人」という役割意識を前提にしてスキルと判断力を同時に測定する。さらに、開発した教育プログラムのカリキュラムの到達目標を可視化し、GPT-TA自動採点結果をドメイン別バッジとして発行する。

そして、その実装・運用を支える教育環境・評価枠組・普及体制等を束ねることを狙い、実務即応・地域定着・全国展開を同時に可能にする教育モデルを構築する。それは具体的に、次の3点を実現することを目指すものである。

#### ① 「AI補完型ITエンジニア養成プログラム」の開発

BASISに準拠した60コマ程度の生成AI×DevOpsカリキュラムとGPT-TA評価システムを運用し、0.5～1年間で“要件定義→実装→運用自動化”を自走できるエンジニアを養成する。

#### ② 地域制約を超える“ハイブリッド教育基盤”の構築

遠隔講義・クラウド統合開発環境・非常勤エンジニア教員養成ブートキャンプを組み込むことで、地理的制約や教員不足を克服し、人口減少地域でも同質の教育を実現する。

#### ③ 教育-評価-産学連携の“循環エコシステム”の整備

BASISバッジによるスキルレベルの可視化、企業DXクリニックと教育プログラムPBLの即時マッチング、修了後1年メンタリングを組み合せ、修了者のキャリア定着率85%、企業の内製DX案件化率70%を目標にした支援環境を構築する。

この教育モデルの全体スキーム図を次頁に示す。



### 1.3.2 年次計画

本事業は、令和7～9年度の3箇年度で推進するものである。

各年度の事業は、会議・調査・開発・実証・報告という共通の枠組で推進する。

会議： 事業の推進および各活動の実作業に関する協議

調査： 教育プログラム開発に資する基礎資料の作成のための4テーマに関する調査

開発： スキル指標や教育プラットフォームの構築、教育プログラムおよび各種教育コンテンツの開発

実証： 教育プログラムの有効性を検証するための実証講座の実施（令和8年度以降）

報告： 事業活動の取り纏めとしての成果報告会の開催と事業報告書（本書）の作成

そして、各年度の取組は、1.3.1で見た本事業の全体スキームに基づき、次のような展開で推移する。

令和7年度： 教育プログラムの基礎設計と初期教材開発

令和8年度： 教育プログラムの中核教材拡充と教育基盤構築

令和9年度： 教育プログラムの応用教材完成・PBL実証とモデル統合

その上で、各年度の具体的な取組内容を次のように設定する。

#### ▼ 各年度の取組内容

	令和7年度	令和8年度	令和9年度
	教育プログラムの基礎設計と 初期教材開発	教育プログラムの中核教材拡充と 教育基盤構築	教育プログラムの応用教材完成・ PBL実証とモデル統合
会議	○ 事業推進委員会 3回開催 ○ 分科会 3回開催	○ 事業推進委員会 3回開催 ○ 分科会 3回開催	○ 事業推進委員会 3回開催 ○ 分科会 3回開催
調査	① IT企業生成AI活用実態調査 ② IT“学び直し”ニーズ実態調査 ③ 生成AIコーディングアシスタント 事例調査 ④ AI活用エンジニアリング教育事例調査	⑤ 地域AI/IT人材需給実態調査 ⑥ 地方企業DevOps実装実態調査 ⑦ DevOps/AIOps教育事例調査 ⑧ プロンプトエンジニアリング教材 事例調査	⑨ 生成AI活用企業運用課題実態調査 ⑩ 地域IT人材キャリア意識調査 ⑪ DevOps人材定着施策事例調査 ⑫ AIセキュリティ教育事例調査
開発	01 AI駆動ITスキル指標設計 02 GPT-TA評価システム設計 03 教育プログラム全体カリキュラム開発 04 教育プログラムシラバス作成 05 教育プログラム講義動画制作 06 教育プログラム講義資料作成 07 教育プログラム演習課題作成 08 教育プログラム自習問題作成 ※ 対象科目：S1～S2前（12コマ） 09 教育プログラム基盤 要件定義	10 AI駆動ITスキル指標開発 11 GPT-TA評価システムβ実装 12 教育プログラムシラバス作成 13 教育プログラム講義動画制作 14 教育プログラム講義資料作成 15 教育プログラム演習課題作成 16 教育プログラム自習問題作成 ※ 対象科目：S2後～S4（23コマ） 17 教育プログラムPBL設計書作成 18 教育プログラム基盤β構築運用	19 BASIS×GPT-TA構築運用 20 教育プログラムシラバス作成 21 教育プログラム講義動画制作 22 教育プログラム講義資料作成 23 教育プログラム演習課題作成 24 教育プログラム自習問題作成 ※ 対象科目：S5～S6（13コマ） 25 教育プログラムPBL教材作成 26 教育プログラム基盤構築運用 27 企業DX診断課題バンク整備 28 修了後メンタリング体制整備
実証	—	◎ 先行実証講座（2週間）実施 ※ 対象科目：S1～S2前	◎ 統合実証講座（6週間）実施 ※ 対象科目：S1～S4＋ミニPBL
報告	○ 成果報告会 1回開催 ○ 事業報告書 100部作成	○ 成果報告会 1回開催 ○ 事業報告書 100部作成	○ 成果報告会 1回開催 ○ 事業報告書 100部作成

各年度で取組内容は異なるが、マイルストーンとして毎年度3回、事業推進委員会（1.4.3で詳説）を開催する。

## 1.4 事業推進

本事業は本年度（令和7年度）、令和7年9月8日から令和8年3月1日まで、6箇月弱に亘って実施された。その間、事業を実施するための事業推進委員会・分科会を東京都（千代田区と新宿区）と岡山県（瀬戸内市）、そして事業成果を共有するための成果報告会を東京都（新宿区）で開催した。

その上で、本事業の**実施内容**および**推進体制**は、次の通りである。

### 1.4.1 実施内容

本事業は令和7～9年度の3箇年度で推進するもので、各年度の年次計画は**1.3.2**で見た通りであるが、本年度事業では実際、次のようなことに取り組んだ。

会議	○ 事業推進委員会 3回開催	第1回：10月27日、第2回：12月15日、第3回：1月28日に開催
	○ 分科会 4回開催	各取組の進捗状況等に応じ、事業推進委員会の前後に開催
調査	① 地方企業生成AI活用実態調査	ネットリサーチ／人口減少県の地方中小企業の管理職 100人
	② IT“学び直し”ニーズ実態調査	ネットリサーチ／人口減少県の地方中小企業の管理職 100人
	③ 生成AIコーディングアシスタント事例調査	デスクリサーチ／各種公開資料（出版物・ウェブサイト等）
	④ AI活用エンジニアリング教育事例調査	デスクリサーチ／各種公開資料（出版物・ウェブサイト等） ヒヤリング／生成AI応用教育担当教員・生成AI活用開発チーム責任者
開発	01 AI駆動ITスキル指標設計	AI補完型ITエンジニアに必要なリテラシー&スキルを項目化・体系化した評価基準の設計書の作成
	02 GPT=TA評価システム設計	提出課題等を生成AIで自動採点し、BASISに照して学習到達度を可視化するクラウド型評価システムの設計書の作成
	03 教育プログラム全体カリキュラム開発	AI補完型IT人材養成プログラム全体（全60コマ）の教育体系の構築
	04 教育プログラムシラバス作成	AI補完型IT人材養成プログラム【S1】【S2前半】（12コマ）の指導計画の策定
	05 教育プログラム講義動画制作	AI補完型IT人材養成プログラム【S1】【S2前半】の講義（7コマ）の動画（AIアバター指導者/mp4）の制作
	06 教育プログラム講義資料作成	AI補完型IT人材養成プログラム【S1】【S2前半】の講義（7コマ）の資料（スライド・ノート等）の作成
	07 教育プログラム演習課題作成	AI補完型IT人材養成プログラム【S1】【S2前半】の演習（5コマ）の課題・指導ガイド・シナリオの作成
	08 教育プログラム自習問題作成	AI補完型IT人材養成プログラム【S1】【S2前半】の自習（12コマ）のCBT問題（全60問）の作成
	09 教育プログラム基盤要件定義	本事業で開発する教育プログラムをオンラインで提供するLMS・クラウドIDE等を統合した教育プラットフォームの設計書の作成
報告	○ 成果報告会 1回開催	1月28日に開催
	○ 事業報告書 100部作成	事業推進委員および関係者に配付、データはウェブサイトで公開予定

## 1.4.2 推進体制

本事業の推進体制として、**事業構成機関**（機関として本事業に参画した学校・企業・団体等）は次の通りである。

機関区分	機関名	役割等	都道府県
教育機関	学校法人電子学園 日本電子専門学校	事業統括	東京都
	学校法人三橋学園 船橋情報ビジネス専門学校	調査・実証	千葉県
	学校法人コンピュータ総合学園 神戸電子専門学校	開発・実証	兵庫県
	学校法人河合塾学園 トライデントコンピュータ専門学校	開発・実証	愛知県
	学校法人吉田学園 専門学校北海道サイバークリエイターズ大学校	調査・実証	北海道
	学校法人有坂中央学園 専門学校中央情報大学校	調査・開発	群馬県
	学校法人せとうち 日本ITビジネスカレッジ	開発・実証	岡山県
	学校法人石川学園 大育情報ビジネス専門学校	調査・実証	沖縄県
	学校法人電子学園 情報経営イノベーション専門職大学	調査・開発	東京都
業界団体 関連企業	JTP株式会社	開発・実証	東京都
	株式会社ファンコーポレーション	開発・実証	神奈川県
	株式会社学夢堂	調査・開発	東京都
	一般社団法人ソフトウェア協会	調査・開発	東京都
行政機関	京丹後市 市長公室 政策企画課	評価・助言	京都府
	善通寺市 総務部 デジタル推進課	評価・助言	香川県

以上の各機関には、事業推進のために次のような協力を要請した。

### 教育機関

- 専修学校における教育内容・AI/DX対応・就職状況等に関する情報提供
- カリキュラム・シラバス・教材・スキル指標等の開発
- 実証講座実施の際の受講者募集・講師調達・会場提供等

### 企業・団体

- AI補完型ITエンジニアの実態・課題・需要等に関する情報提供
- システム開発等におけるAI活用状況・就職事情・企業ニーズ等に関する情報提供

### 行政機関・有識者

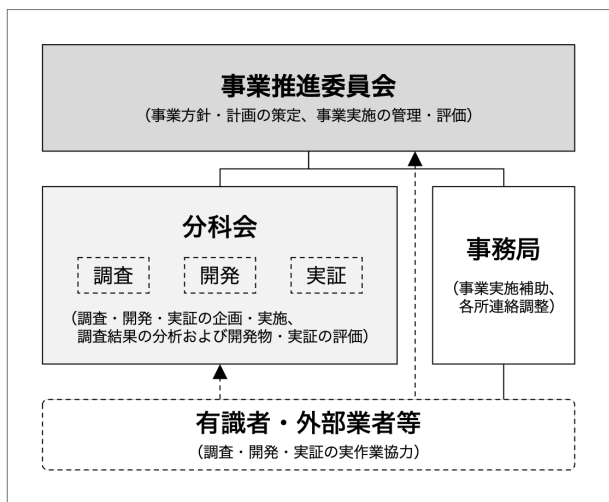
- 対象地域の当該分野の社会的課題・産業事情等に関する情報提供と助言

そして、開発した教育プログラムおよび各種成果物に対する評価と、事業終了後の展開策の検討に関する協力を全機関に要請した。

その上で、本事業の実施体制として、**事業推進委員会**と**分科会**を設置した。

**事業推進委員会**（当校主幹）は、事業推進の方針・計画・内容・方法等、本事業の重要事項を検討・決定し、事業全体のマネジメントや成果の評価等を担う推進主体である。その構成機関は、IT分野の専修学校の他、当該分野の専門教育や情報システム開発に関する豊富な知見・ノウハウ・実績等を有する団体・企業、そして人口減少地域の地方公共団体である。

**分科会**は、事業推進委員会が決定した方針・事項等に従い、調査・開発等の各活動の仕様策定や実作業を担った。



その際、実作業の一部は、事業推進委員会・分科会の管理の下、**有識者・外部企業等**の協力を得て進めた。また、文部科学省および事業推進委員会・分科会構成機関への連絡・調整の他、事業経費や各種文書の管理等の事務作業は、**事務局**が担った。

尚、本事業の**事業推進委員会構成員**（事業構成機関からの参画者）は次の通りである。

No.	氏名	所属・役職	都道府県
1	杉浦敦司	学校法人電子学園 日本電子専門学校 校長	東京都
2	鳥居高之	学校法人三橋学園 船橋情報ビジネス専門学校 校長	千葉県
3	福岡壯治	学校法人コンピュータ総合学園 神戸電子専門学校 校長	兵庫県
4	中原千賀子	学校法人河合塾学園 トライデントコンピュータ専門学校 校長	愛知県
5	橋本直樹	学校法人吉田学園 専門学校北海道サイバークリエイターズ大学校 校長	北海道
6	中島慎太郎	学校法人有坂中央学園 専門学校中央情報大学校 理事長・校長	群馬県
7	田中旬一	学校法人せとうち 日本ITビジネスカレッジ 理事長	岡山県
8	石川正剛	学校法人石川学園 大育情報ビジネス専門学校 理事長	沖縄県
9	古賀稔邦	学校法人電子学園 情報経営イノベーション専門職大学 副学長	東京都
10	古川宏幸	JTP株式会社 コーポレート本部 人事部長	東京都
11	石本則子	株式会社ファンコーポレーション 代表取締役社長	神奈川県
12	東方宏	株式会社学夢堂 代表取締役	東京都
13	木田徳彦	一般社団法人ソフトウェア協会 理事	東京都
14	堀江亮平	京丹後市 市長公室 政策企画課 係長	京都府
15	大喜多康彦	善通寺市 総務部 デジタル推進課 課長補佐	香川県
16	勝田雅人	学校法人河合塾学園 トライデントコンピュータ専門学校 前校長	愛知県
17	大川晃一	学校法人電子学園 日本電子専門学校 エンジニア教育 部長	東京都

### 1.4.3 事業推進委員会

事業推進の中心的役割を担う**事業推進委員会**（その構成員は1.4.2）は、次の日時・場所で本事業期間内に3回開催された。

第1回： 令和7年10月27日（月）15:00～17:00 @アルカディア市ヶ谷 （東京都千代田区）

第2回： 令和7年12月15日（月）15:00～16:15 @日本ITビジネスカレッジ（岡山県瀬戸内市）

第3回： 令和8年01月28日（水）15:00～17:00 @日本電子専門学校 （東京都新宿区）

各回の概要は次の通りである。

回次	第1回	第2回	第3回
日時	令和7年10月27日（月） 15:00～17:00	令和7年12月15日（月） 15:00～16:15	令和8年1月28日（水） 15:00～17:00
場所	アルカディア市ヶ谷 （東京都千代田区）	日本ITビジネスカレッジ （岡山県瀬戸内市）	日本電子専門学校 （東京都新宿区）
議事	1 委員長挨拶 2 企画者挨拶 3 各委員挨拶 4 事業概要説明 5 意見交換等	1 委員長挨拶 2 企画者挨拶 3 会場校挨拶 4 事業進捗状況共有 5 意見交換等	1 委員長挨拶 2 企画者挨拶 3 本年度事業成果報告 4 次年度事業展開検討 5 意見交換等
配付物	① 委員名簿 ② 事業計画書 ③ 第1回事業推進委員会検討資料	① 委員名簿 ② 第1回事業推進委員会議事録 ③ 第2回事業推進委員会検討資料 ④ 生成AI活用&IT“学び直し” ニーズ実態調査 ⑤ 生成AIコーディングアシスタント 事例調査	① 委員名簿 ② 第2回事業推進委員会議事録 ③ 第3回事業推進委員会検討資料
備考	会前後に調査・開発分科会開催	会前に現地企業等視察 会後に開発分科会開催	会前に開発分科会開催 会後に成果報告会開催

事業方針の検討や事業成果の共有、IT/DX/AI関連や人材育成等に関する活発な議論等が展開されるこの事業推進委員会の開催は、事業全体のマネジメントにおいて実施期間中3回のマイルストーンにもなった。

尚、事業推進委員会の第1回の会前後に調査分科会・開発分科会（該当委員等）、第2回の会前に現地企業等視察、会後に開発分科会（該当委員等）、そして第3回の会前に開発分科会（該当委員等）、会後に成果報告会（全委員）が開催された。

## 2.1 調査概要

本事業では本年度（令和7年度）、教育モデル構築に先立ち、それに資する次の**4調査**を実施した。

- 調査①： 地方企業生成AI活用実態調査
- 調査②： IT“学び直し”ニーズ実態調査
- 調査③： 生成AIコーディングアシスタント事例調査
- 調査④： AI活用エンジニアリング教育事例調査

この4調査は各々、本年度の活動（教育プログラムの基礎設計と初期教材開発）の基礎資料にするべく、次のような**目的**で実施した。

### 調査① 地方企業生成AI活用実態調査

人口減少地域における生成AI活用の現状や課題を把握することで、本教育プログラムの教育目標と教材重点を明かにし、開発の基礎資料にする。

### 調査② IT“学び直し”ニーズ実態調査

人口減少地域の若年層・社会人におけるIT“学び直し”ニーズの実情を把握することで、本教育プログラムの想定対象を明かにし、開発の基礎資料にする。

### 調査③ 生成AIコーディングアシスタント事例調査

国内外で利用されている生成AIコーディングアシスタントの事例や機能・導入形態等に関する情報を収集・整理することで、本教育プログラムの演習環境要件を明かにし、開発の基礎資料にする。

### 調査④ AI活用エンジニアリング教育事例調査

国内外のAI活用エンジニアリング教育の事例や指導法等に関する情報を収集・整理することで、本教育プログラムのカリキュラム・シラバス構成要件を明かにし、開発の基礎資料にする。

次節以降（2.2-2.4）、この4調査の各々について、調査内容・調査結果等を詳説する。

## 2.2 調査① 地方企業生成AI活用実態調査

調査①は**地方企業生成AI活用実態調査**である。

これは、人口減少地域における生成AI活用の現状や課題を把握することで、本教育プログラムの教育目標と教材重点を明かにし、開発の基礎資料にするためのものである。

その調査仕様や調査結果を以下に報告する。

### 2.2.1 調査仕様

本調査の**仕様**（目的・対象・手法・内容・反映方針等）は次の通りである

名称	地方企業生成AI活用実態調査
目的	人口減少地域における生成AI活用の現状や課題を把握することで、本教育プログラムの教育目標と教材重点を明かにし、開発の基礎資料にする。
対象	人口減少県の中小企業管理職100人（+首都圏中核都県の中小企業管理職40人）➡ 有効回答140件
手法	ネットリサーチ
項目	企業規模、生成AI導入有無・活用領域・導入効果、使用ツール、課題、必要スキル、希望学習形態 等

### 2.2.2 調査内容

本調査における具体的な**質問項目**は次の15項である。

回答者企業における――

- ① 企業規模（従業員数）
- ② 生成AI導入現況
- ③ 生成AI活用主要業務分野
- ④ 生成AI活用案件実施作業
- ⑤ 生成AI活用案件チーム規模
- ⑥ 利用中（導入検討中）生成AIツール・サービス
- ⑦ 生成活用が定着した開発工程
- ⑧ 生成AI入力可能データ範囲・方針
- ⑨ 生成AI利用ルール・体制
- ⑩ 生成AI導入・運用上のボトルネック
- ⑪ 生成AI活用による業務時間影響
- ⑫ 生成AI活用による開発品質影響
- ⑬ 不足・要強化エンジニアリングスキル領域
- ⑭ 有効性が高い人材養成手法
- ⑮ 今後重視するエンジニア要件

これに基づいて作成した**調査票**を次頁に掲載する。



**問06** 貴社における現在利用中 (or 導入検討中) の生成AI関連ツール・サービスとして該当するものは、次のうちどれですか。[上位3つまで選択]

- |  |   |
|--|---|
| ① ChatGPT (OpenAI)                       | ② OpenAI API (OpenAI)                   |
| ③ Azure OpenAI Service (Microsoft Azure) | ④ Copilot for Microsoft 365 (Microsoft) |
| ⑤ Gemini for Google Workspace (Google)   | ⑥ Vertex AI (Google Cloud)              |
| ⑦ Claude (Anthropic)                     | ⑧ GitHub Copilot (GitHub)               |
| ⑨ Amazon Bedrock (AWS)                   | ⑩ Atlassian Intelligence (Atlassian)    |
| ⑪ Slack AI (Slack)・Notion AI (Notion)    | ⑫ 自社LLM・RAG基盤 等                         |
| ⑬ その他 ( )                                |   |
| ⑭ 現在利用中 (導入検討中) の生成AI関連ツール・サービスがない       |   |

**問07** 貴社において生成AI活用が最もよく定着している開発サイクル上の工程として該当するものは、次のうちどれですか。[上位2つまで選択]

- |                    |               |
|--------------------|---------------|
| ① 企画・要件定義          | ② アーキテクチャ設計   |
| ③ 開発・実装            | ④ テスト・QA      |
| ⑤ セキュリティレビュー       | ⑥ デプロイ/リリース管理 |
| ⑦ 運用監視・インシデント対応    | ⑧ 運用改善・自動化    |
| ⑨ 非開発業務 (営業・管理 等)  | ⑩ その他 ( )     |
| ⑪ 生成AIを活用している工程がない |               |

**問08** 貴社において生成AIに入力できる業務データの範囲と入力先の標準方針として該当するものは、次のうちどれですか。[1つ選択]

- ① 利用禁止 (業務データは一切入力しない、評価は公開資料やダミーデータのみ)
- ② 公開資料・ダミーデータのみ可 (外部の生成AIサービスへの入力は可)
- ③ 匿名化・疑似化データのみ可 (個人情報・機密情報を確実に除去したものに限り、外部サービス可)
- ④ 社内非機密データまで可 (外部サービス可、ベンダーによる学習利用は拒否)
- ⑤ 社外秘データも一部可 (外部サービス可、事前承認+DLP+利用ログが必須)
- ⑥ 機密データは自社管理環境のみ可 (オンプレや専用クラウド等の隔離環境で可、外部SaaS送信は不可)
- ⑦ 機密データも外部サービスで可 (学習利用は禁止、暗号化・事前承認・監査ログが必須)
- ⑧ 機密データの学習利用も条件付きで可 (ファインチューニング等の目的限定で条件明示の上で可)
- ⑨ 共通の標準なし、案件毎に都度判断 (統一ルールは未整備)
- ⑩ その他 ( )
- ⑪ 生成AI活用業務がない

**問09** 貴社における生成AI利用のルールや体制の整備状況として該当するものは、次のうちどれですか。  
[1つ選択]

- ① 未整備 (禁止・許可の判断が個人に委ねられている)
- ② 注意喚起のみ (口頭・掲示による注意喚起はあるが、文書化された指針はない)
- ③ ガイドラインのみ (文書化された指針はあるが、教育・ログ運用はない)
- ④ ガイドライン+教育 (年1回以上の教育・周知を実施するが、ログ運用はない)
- ⑤ ポリシー+ログ (許可・禁止やデータ取扱い基準を明記した指針を制定し、最低限の利用ログを取得している)
- ⑥ 例外承認の運用 (上記に加え、申請→承認→証跡の例外承認プロセスを運用し、定期レビューを実施している)
- ⑦ 責任体制の整備 (上記に加え、責任者任命・委員会設置や運用KPIによるモニタリングを実施している)
- ⑧ 外部レビュー・監査 (上記に加え、法務・セキュリティ等の第三者レビューや監査を年1回以上実施している)
- ⑨ マネジメントシステム統合 (上記に加え、ISMS等のマネジメントシステムにAI利用を組み込み、リスク評価と是正のPDCAを実施している)
- ⑩ その他 ( )
- ⑪ 生成AI活用業務がない

**問10** 貴社における生成AI導入・運用で直面している主なボトルネックとして該当するものは、次のうちどれですか。[上位3つまで選択]

- |                   |                  |
|-------------------|------------------|
| ① セキュリティ・法令順守の懸念  | ② 著作権・知財の不確実性    |
| ③ 生成物の正確性・一貫性の不足  | ④ データ整備の不足       |
| ⑤ 品質保証・テスト自動化の未整備 | ⑥ 利用ルール・例外承認の未整備 |
| ⑦ 運用ログ・監査の未整備     | ⑧ ツール比較・選定の難航    |
| ⑨ コスト・ライセンス管理の負担  | ⑩ その他 ( )        |
| ⑪ 生成AIを導入・運用していない |                  |

**問11** 貴社において生成AI活用が平均的な業務時間に与えている影響の程度として該当するものは、次のうちどれですか。[1つ選択]

- |           |            |            |               |
|-----------|------------|------------|---------------|
| ① 50%以上向上 | ② 20~49%向上 | ③ 10~19%向上 | ④ 1~9%向上      |
| ⑤ 変化なし    | ⑥ 1~9%悪化   | ⑦ 10~19%悪化 | ⑧ 20~49%悪化    |
| ⑨ 50%以上悪化 | ⑩ その他 ( )  |            | ⑪ 生成AI活用業務がない |

**問12** 貴社において生成AI活用が開発品質（欠陥率・再現性・脆弱性等）に与えている影響の程度として該当するものは、次のうちどれですか。[1つ選択]

- |                   |                 |           |        |
|-------------------|-----------------|-----------|--------|
| ① 20%以上改善         | ② 10~19%改善      | ③ 1~9%改善  | ④ 変化なし |
| ⑤ 1~9%悪化          | ⑥ 10~19%悪化      | ⑦ 20%以上悪化 |        |
| ⑧ 改善と悪化が混在（チーム差大） | ⑨ 評価不可（測定未実施）   |           |        |
| ⑩ その他 ( )         | ⑪ 生成AI活用開発業務がない |           |        |

**問13** 貴社において不足している（or 強化したい）エンジニアリング系スキル領域として該当するものは、次のうちどれですか。[上位3つまで選択]

- |                         |                        |
|-------------------------|------------------------|
| ① プロンプト設計・出力評価          | ② LLM-API連携・RAG設計実装    |
| ③ データ前処理・ETL・クレンジング     | ④ 自動テスト設計・品質ゲート運用      |
| ⑤ セキュリティ設計・法令・ライセンス対応   | ⑥ CI/CD・IaC構築          |
| ⑦ クラウド基盤設計・Kubernetes運用 | ⑧ 運用監視・信頼性設計（AIOpsを含む） |
| ⑨ 要件定義・アーキテクチャ設計        | ⑩ その他 ( )              |
| ⑪ エンジニアリング系スキルは必要ない     |                        |

**問14** 貴社において有効性が高いと思える人材育成の手法として該当するものは、次のうちどれですか。

- |                          |                              |
|--------------------------|------------------------------|
| ① 短期集中ブートキャンプ（2~5日）      | ② 週次ワークショップ（3箇月程度）[上位2つまで選択] |
| ③ オンデマンドeラーニング（チェックテスト付） | ④ 座学研修（講義中心）                 |
| ⑤ ハイブリッド講義（オンライン+対面）     | ⑥ 集中ハンズオン（演習中心）              |
| ⑦ 実課題PBL（自社課題を教材化）       | ⑧ 自習・社内勉強会キット提供（自主運用）        |
| ⑨ メンタリング・コーチング（個別伴走）     | ⑩ その他 ( )                    |
| ⑪ 人材育成は実施しない（できない）       |                              |

**問15** 貴社が求めるエンジニア要件として今後重視する能力として該当するものは、次のうちどれですか。

- |                                    |              |
|------------------------------------|--------------|
| ① AIリスクを評価し、規程に沿って安全運用できる力         | [上位3つまで選択]   |
| ② 要件をプロンプトに変換して高品質出力を引き出せる力        |              |
| ③ 社内データをRAG/APIで連携して業務に実装できる力      |              |
| ④ テストを自動化して品質ゲートで欠陥を抑制できる力         |              |
| ⑤ 脅威を想定し、安全設計を実装して運用維持できる力         |              |
| ⑥ 業務要件を構造化して実装可能な設計に落とし込める力        |              |
| ⑦ CI/CDとIaCを整備し、変更の自動化と安定運用を実施できる力 |              |
| ⑧ KPIを設計し、成果を計測して改善を主導できる力         |              |
| ⑨ 目的を言語化し、合意を形成して協働を推進できる力         |              |
| ⑩ その他 ( )                          | ⑪ エンジニアは必要ない |

質問は以上です。ご協力ありがとうございました。

## 2.2.3 調査結果

本調査の結果は次の通りである

【調査対象】人口減少県の中小企業100社および首都圏中核都県の中小企業40社

### 「人口減少県」の選定過程

- ① 中長期の減少傾向： 2015年→2020年（直近の国勢調査区間）で人口が減った都道府県
- ② 直近の総人口増減： 2024年10月時点の人口増減率で減少を確認し、減少率1.0%以上を「重症県」と定義
- ③ 社会増減の符号： 2024年の〈自然減+社会減（転出超過）〉（二重減少）に該当する県を「人口減少県」と定義

その結果、下掲の23県を「人口減少県」とし、このうち下線付きの18県を「重症県」とした。

本調査の回答者は、この「人口減少県」に所在する中小企業100社の管理職である（各件の（ ）内は回答者数）。

[東北] 青森県 (4)・岩手県 (4)・秋田県 (4)・山形県 (4)・福島県 (6)  
 [北陸] 新潟県 (9)・富山県 (4)・石川県 (6) [東海・近畿] 三重県 (3)・和歌山県 (3)  
 [中国] 鳥取県 (3)・島根県 (3)・岡山県 (6)・広島県 (9)・山口県 (3)  
 [四国] 徳島県 (3)・香川県 (3)・愛媛県 (5)・高知県 (3)  
 [九州] 佐賀県 (2)・長崎県 (4)・大分県 (4)・鹿児島県 (5)

本調査では、「人口減少県」の回答結果と比較すべく、「首都圏中核都県」に所在する「**学校法人電子学園 後援会** 会員企業40社（東京都 33社・神奈川県 7社）からも回答を得た。  
 その際、略記として、「人口減少県」を**PD** (Population-Declining Prefectures)、「首都圏中核都県」を**MC** (Metropolitan Core Prefectures) と表記する。

問01 貴社の従業員（常用雇用者）数として該当するものは、次のうちどれですか。[1つ選択]

選択肢	全体		PD		MC	
	n	%	n	%	n	%
1～4人	12	8.6	12	12.0	0	0.0
5～9人	13	9.3	13	13.0	0	0.0
10～19人	24	17.1	19	19.0	5	12.5
20～29人	13	9.3	11	11.0	2	5.0
30～49人	25	17.9	17	17.0	8	20.0
50～99人	23	16.4	14	14.0	9	22.5
100～199人	8	5.7	7	7.0	1	2.5
200～299人	5	3.6	3	3.0	2	5.0
300～499人	6	4.3	2	2.0	4	10.0
500人以上	11	7.9	2	2.0	9	22.5

(N=全体140・PD100・MC40、SA)

問02 貴社における生成AI導入の現況として該当するものは、次のうちどれですか。[1つ選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
未導入（予定なし）	38	27.1	31	31.0	7	17.5
検討中（情報収集中・環境整備中）	25	17.9	21	21.0	4	10.0
試行中（PoC中）	17	12.1	13	13.0	4	10.0
限定導入（個人・有志）	28	20.0	17	17.0	11	27.5
部分導入（特定の部門・工程）	15	10.7	9	9.0	6	15.0
本格導入（複数部門）	7	5.0	4	4.0	3	7.5
全社展開（標準ツール化）	8	5.7	3	3.0	5	12.5
導入したが縮小・停止	1	0.7	1	1.0	0	0.0
その他（※）	1	0.7	1	1.0	0	0.0

(N=全体140・PD100・MC40、SA)

※〈その他〉の例：「顧客環境でのみ生成AIを利用することがある」

問03 貴社において生成AIを活用している主な業務分野として該当するものは、次のうちどれですか。

[3つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
文書作成・ナレッジ管理	55	39.3	33	33.0	22	55.0
システム開発・実装	46	32.9	29	29.0	17	42.5
テスト・品質保証	23	16.4	14	14.0	9	22.5
企画・要件定義・見積	21	15.0	17	17.0	4	10.0
人事・総務・法務文書	18	12.9	11	11.0	7	17.5
データ管理・ETL・検索	10	7.1	8	8.0	2	5.0
運用・保守（監視・AIOps）	9	6.4	6	6.0	3	7.5
営業・マーケティング	7	5.0	3	3.0	4	10.0
その他（※）	7	5.0	4	4.0	3	7.5
カスタマーサポート・FAQ	6	4.3	4	4.0	2	5.0
生成AI活用業務がない	44	31.4	37	37.0	7	17.5

(N=全体140・PD100・MC40、MA3)

※〈その他〉の例：「社内教育・研修での活用」「技術調査・R&D活動での活用」「経営会議資料・事業戦略立案での活用」「会社広報・採用ブランディングでの活用」「セミナー・イベント企画運営での活用」等

問04 貴社における生成AI活用案件で主に実施している作業の内容として該当するものは、次のうちどれですか。

[3つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
コーディング補完・リファクタ	40	28.6	26	26.0	14	35.0
ドキュメント／設計書生成・整備	32	22.9	24	24.0	8	20.0
コードレビュー・静的解析	20	14.3	12	12.0	8	20.0
運用手順書作成・一次対応自動化	18	12.9	11	11.0	7	17.5
要件定義・設計支援 (ユーザーストーリー化・プロンプト作成)	17	12.1	13	13.0	4	10.0
社内検索・RAGチャットボット	16	11.4	9	9.0	7	17.5
業務自動化 (ワークフロー・スクリプト)	14	10.0	9	9.0	5	12.5
テストケース自動生成・テスト計画	13	9.3	8	8.0	5	12.5
データ前処理・変換スクリプト生成	9	6.4	6	6.0	3	7.5
その他 (※)	5	3.6	3	3.0	2	5.0
生成AI活用案件がない	50	35.7	41	41.0	9	22.5

(N=全体140・PD100・MC40、MA3)

※〈その他〉の例：「生成AIの評価・ベンチマーク実験」「技術研修用の演習問題・模範解答作成」「採用選考用コーディングテストの問題作成」「WBS・工数見積りの叩き台作成」「社内ルール・規程案の叩き台作成」

問05 貴社における生成AI活用案件で最もよくあるチーム規模として該当するものは、次のうちどれですか。

[1つ選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
1人 (個人)	30	21.4	19	19.0	11	27.5
2人	12	8.6	9	9.0	3	7.5
3人	11	7.9	7	7.0	4	10.0
4～5人	20	14.3	11	11.0	9	22.5
6～9人	4	2.9	3	3.0	1	2.5
10～14人	2	1.4	2	2.0	0	0.0
15～19人	1	0.7	1	1.0	0	0.0
20～29人	2	1.4	1	1.0	1	2.5
30人以上	2	1.4	1	1.0	1	2.5
生成AI活用案件がない	56	40.0	46	46.0	10	25.0

(N=全体140・PD100・MC40、SA)

問06 貴社における現在利用中 (or 導入検討中) の生成AI関連ツール・サービスとして該当するものは、次のうちどれですか。[3つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
ChatGPT (OpenAI)	75	53.6	52	52.0	23	57.5
Copilot for Microsoft 365 (Microsoft)	33	23.6	18	18.0	15	37.5
Gemini for Google Workspace (Google)	16	11.4	8	8.0	8	20.0
OpenAI API (OpenAI)	14	10.0	9	9.0	5	12.5
GitHub Copilot (GitHub)	9	6.4	6	6.0	3	7.5
Azure OpenAI Service (Microsoft Azure)	7	5.0	4	4.0	3	7.5
その他 (※)	7	5.0	4	4.0	3	7.5
自社LLM・RAG基盤 等	6	4.3	2	2.0	4	10.0
Claude (Anthropic)	4	2.9	2	2.0	2	5.0
Vertex AI (Google Cloud)	0	0.0	0	0.0	0	0.0
Amazon Bedrock (AWS)	0	0.0	0	0.0	0	0.0
Atlassian Intelligence (Atlassian)	0	0.0	0	0.0	0	0.0
Slack AI (Slack)・Notion AI (Notion)	0	0.0	0	0.0	0	0.0
現在利用中 (導入検討中) の生成AI関連ツール・サービスがない	38	27.1	33	33.0	5	12.5

(N=全体140・PD100・MC40、MA3)

※〈その他〉の例：「国産LLMチャットサービス (日本語特化型)」「検索連動型QAサービス (Perplexity等)」「翻訳特化クラウドAIサービス」「画像生成AI (デザイン・資料作成用)」「ブラウザ拡張型AIアシスタント」等

問07 貴社において生成AI活用が最もよく定着している開発サイクル上の工程として該当するものは、次のうちどれですか。[2つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
開発・実装	35	25.0	21	21.0	14	35.0
非開発業務 (営業・管理 等)	26	18.6	19	19.0	7	17.5
テスト・QA	17	12.1	8	8.0	9	22.5
企画・要件定義	14	10.0	9	9.0	5	12.5
運用改善・自動化	11	7.9	7	7.0	4	10.0
その他 (※)	5	3.6	3	3.0	2	5.0
アーキテクチャ設計	4	2.9	2	2.0	2	5.0
セキュリティレビュー	0	0.0	0	0.0	0	0.0
デプロイ/リリース管理	0	0.0	0	0.0	0	0.0
運用監視・インシデント対応	0	0.0	0	0.0	0	0.0
生成AIを活用している工程がない	55	39.3	44	44.0	11	27.5

(N=全体140・PD100・MC40、MA2)

※〈その他〉の例：「プロジェクト計画立案・進捗管理」「品質保証全体のマネジメント」「システム/データ移行」「ベンダー選定・調達管理」「社内研修・人材育成」

問08 貴社において生成AIに入力できる業務データの範囲と入力先の標準方針として該当するものは、次のうちどれですか。[2つまで選択]

選択肢	全体		PD		MC	
	n	%	n	%	n	%
共通の標準なし、案件毎に都度判断（統一ルールは未整備）	22	15.7	13	13.0	9	22.5
利用禁止 （業務データは一切入力しない、評価は公開資料やダミーデータのみ）	21	15.0	16	16.0	5	12.5
公開資料・ダミーデータのみ可（外部の生成AIサービスへの入力は可）	18	12.9	13	13.0	5	12.5
匿名化・疑似化データのみ可 （個人情報・機密情報を確実に除去したものに限り、外部サービス可）	13	9.3	9	9.0	4	10.0
機密データは自社管理環境のみ可 （オンプレや専用クラウド等の隔離環境で可、外部SaaS送信は不可）	7	5.0	2	2.0	5	12.5
社内非機密データまで可 （外部サービス可、ベンダーによる学習利用は拒否）	4	2.9	3	3.0	1	2.5
その他（※）	2	1.4	2	2.0	0	0.0
社外秘データも一部可（外部サービス可、事前承認+DLP+利用ログが必須）	1	0.7	1	1.0	0	0.0
機密データも外部サービスで可 （学習利用は禁止、暗号化・事前承認・監査ログが必須）	0	0.0	0	0.0	0	0.0
機密データの学習利用も条件付きで可 （ファインチューニング等の目的限定で条件明示の上で可）	0	0.0	0	0.0	0	0.0
生成AI活用業務がない	52	37.1	41	41.0	11	27.5

(N=全体140・PD100・MC40、MA2)

※〈その他〉の例：「部門毎に独自ルールで運用」「顧客契約毎に入力可否を判断」

問09 貴社における生成AI利用のルールや体制の整備状況として該当するものは、次のうちどれですか。[1つ選択]

選択肢	全体		PD		MC	
	n	%	n	%	n	%
注意喚起のみ （口頭・掲示による注意喚起はあるが、文書化された指針はない）	36	25.7	23	23.0	13	32.5
未整備（禁止・許可の判断が個人に委ねられている）	27	19.3	21	21.0	6	15.0
ガイドライン+教育 （年1回以上の教育・周知を実施するが、ログ運用はない）	10	7.1	6	6.0	4	10.0
ガイドラインのみ （文書化された指針はあるが、教育・ログ運用はない）	9	6.4	7	7.0	2	5.0
その他（※）	9	6.4	6	6.0	3	7.5
ポリシー+ログ（許可・禁止やデータ取扱い基準を明記した指針を制定し、最低限の利用ログを取得している）	7	5.0	3	3.0	4	10.0
責任体制の整備（上記に加え、責任者任命・委員会設置や運用KPIによるモニタリングを実施している）	2	1.4	1	1.0	1	2.5
例外承認の運用（上記に加え、申請→承認→証跡の例外承認プロセスを運用し、定期レビューを実施している）	0	0.0	0	0.0	0	0.0
外部レビュー・監査（上記に加え、法務・セキュリティ等の第三者レビューや監査を年1回以上実施している）	0	0.0	0	0.0	0	0.0
マネジメントシステム統合（上記に加え、ISMS等のマネジメントシステムにAI利用を組み込み、リスク評価と是正のPDCAを実施している）	0	0.0	0	0.0	0	0.0
生成AI活用業務がない	40	28.6	33	33.0	7	17.5

(N=全体140・PD100・MC40、SA)

※〈その他〉の例：「親会社ポリシーを簡易準用」「事業部毎に独自ガイドラインを運用」「顧客契約条項ベースで案件毎に判断」「海外拠点のみ詳細な運用ルールあり」「AI専任チームが暫定ルールで試行」等

問10 貴社における生成AI導入・運用で直面している主なボトルネックとして該当するものは、次のうちどれですか。[3つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
セキュリティ・法令順守の懸念	67	47.9	43	43.0	24	60.0
生成物の正確性・一貫性の不足	41	29.3	27	27.0	14	35.0
コスト・ライセンス管理の負担	37	26.4	32	32.0	5	12.5
著作権・知財の不確実性	28	20.0	19	19.0	9	22.5
ツール比較・選定の難航	25	17.9	23	23.0	2	5.0
データ整備の不足	24	17.1	21	21.0	3	7.5
利用ルール・例外承認の未整備	23	16.4	18	18.0	5	12.5
品質保証・テスト自動化の未整備	18	12.9	16	16.0	2	5.0
運用ログ・監査の未整備	11	7.9	11	11.0	0	0.0
その他(※)	11	7.9	9	9.0	2	5.0
生成AIを導入・運用していない	42	30.0	33	33.0	9	22.5

(N=全体140・PD100・MC40、MA3)

※〈その他〉の例：「社内のAI推進人材・旗振り役の不在」「顧客側ポリシーでAI提案が制約」「現場のITリテラシーの不足」「経営層の理解・優先度の低さ」「地方拠点のネットワーク・PC性能の不足」等

問11 貴社において生成AI活用が平均的な業務時間に与えている影響の程度として該当するものは、次のうちどれですか。[1つ選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
50%以上向上	0	0.0	0	0.0	0	0.0
20～49%向上	3	2.1	2	2.0	1	2.5
10～19%向上	20	14.3	11	11.0	9	22.5
1～9%向上	26	18.6	17	17.0	9	22.5
変化なし	17	12.1	13	13.0	4	10.0
1～9%悪化	2	1.4	2	2.0	0	0.0
10～19%悪化	1	0.7	1	1.0	0	0.0
20～49%悪化	0	0.0	0	0.0	0	0.0
50%以上悪化	0	0.0	0	0.0	0	0.0
その他(※)	23	16.4	18	18.0	5	12.5
生成AI活用業務がない	48	34.3	36	36.0	12	30.0

(N=全体140・PD100・MC40、SA)

※〈その他〉の例：「各業務で短縮と増加が混在」「導入直後で影響を評価中」「一部の担当者のみ利用していて全体的には不明」「繁忙期のみ利用が集中して年平均を出し難い」「記録がなく定量評価を行っていない」等

問12 貴社において生成AI活用が開発品質（欠陥率・再現性・脆弱性等）に与えている影響の程度として該当するものは、次のうちどれですか。[1つ選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
20%以上改善	6	4.3	3	3.0	3	7.5
10～19%改善	11	7.9	6	6.0	5	12.5
1～9%改善	5	3.6	4	4.0	1	2.5
変化なし	18	12.9	12	12.0	6	15.0
1～9%悪化	1	0.7	1	1.0	0	0.0
10～19%悪化	1	0.7	1	1.0	0	0.0
20%以上悪化	0	0.0	0	0.0	0	0.0
改善と悪化が混在（チーム差大）	2	1.4	2	2.0	0	0.0
評価不可（測定未実施）	33	23.6	24	24.0	9	22.5
その他（※）	5	3.6	4	4.0	1	2.5
生成AI活用開発業務がない	58	41.4	43	43.0	15	37.5

(N=全体140・PD100・MC40、SA)

※〈その他〉の例：「ドキュメント品質のみ向上したが欠陥率は不変」「チーム間で品質差が大きく平均化できない」「品質指標の定義変更中で比較が困難」「顧客評価は改善したが社内指標は不変」「テスト設計は改善したが本番品質は不明」

問13 貴社において不足している（or 強化したい）エンジニアリング系スキル領域として該当するものは、次のうちどれですか。[3つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
要件定義・アーキテクチャ設計	54	38.6	37	37.0	17	42.5
セキュリティ設計・法令・ライセンス対応	41	29.3	29	29.0	12	30.0
クラウド基盤設計・Kubernetes運用	26	18.6	19	19.0	7	17.5
自動テスト設計・品質ゲート運用	24	17.1	21	21.0	3	7.5
CI/CD・IaC構築	23	16.4	23	23.0	0	0.0
データ前処理・ETL・クレンジング	22	15.7	17	17.0	5	12.5
プロンプト設計・出力評価	21	15.0	13	13.0	8	20.0
運用監視・信頼性設計（AIOpsを含む）	21	15.0	16	16.0	5	12.5
LLM-API連携・RAG設計実装	17	12.1	9	9.0	8	20.0
その他（※）	12	8.6	9	9.0	3	7.5
エンジニアリング系スキルは必要ない	18	12.9	11	11.0	7	17.5

(N=全体140・PD100・MC40、MA3)

※〈その他〉の例：「プロジェクトマネジメント・進行管理」「UI/UXデザイン・情報設計」「データガバナンス・データ管理」「レガシー刷新・モダナイゼーション」「業務領域とITを橋渡しするスキル」等

問14 貴社において有効性が高いと思える人材育成の手法として該当するものは、次のうちどれですか。

[2つまで選択]

選択肢	全体		PD		MC	
	n	%	n	%	n	%
オンデマンドeラーニング (チェックテスト付)	54	38.6	41	41.0	13	32.5
ハイブリッド講義 (オンライン+対面)	45	32.1	33	33.0	12	30.0
集中ハンズオン (演習中心)	34	24.3	27	27.0	7	17.5
メンタリング・コーチング (個別伴走)	30	21.4	17	17.0	13	32.5
座学研修 (講義中心)	27	19.3	19	19.0	8	20.0
自習・社内勉強会キット提供 (自主運用)	25	17.9	23	23.0	2	5.0
短期集中ブートキャンプ (2~5日)	20	14.3	13	13.0	7	17.5
実課題PBL (自社課題を教材化)	15	10.7	11	11.0	4	10.0
週次ワークショップ (3箇月程度)	9	6.4	7	7.0	2	5.0
その他 (※)	5	3.6	4	4.0	1	2.5
人材育成は実施しない (できない)	23	16.4	21	21.0	2	5.0

(N=全体140・PD100・MC40、MA2)

※〈その他〉の例：「大学・専門学校との連携講座への派遣」「業界カンファレンス参加費の補助」「社外コミュニティ・勉強会参加の支援」「社内ハッカソン・アイデアソン開催」「副業・兼業を通じた実務経験の容認」

問15 貴社が求めるエンジニア要件として今後重視する能力として該当するものは、次のうちどれですか。

[3つまで選択]

選択肢	全体		PD		MC	
	n	%	n	%	n	%
業務要件を構造化して実装可能な設計に落とし込める力	53	37.9	39	39.0	14	35.0
目的を言語化し、合意を形成して協働を推進できる力	51	36.4	34	34.0	17	42.5
AIリスクを評価し、規程に沿って安全運用できる力	46	32.9	31	31.0	15	37.5
脅威を想定し、安全設計を実装して運用維持できる力	35	25.0	27	27.0	8	20.0
要件をプロンプトに変換して高品質出力を引き出せる力	30	21.4	18	18.0	12	30.0
テストを自動化して品質ゲートで欠陥を抑制できる力	28	20.0	24	24.0	4	10.0
CI/CDとIaCを整備し、変更の自動化と安定運用を実施できる力	23	16.4	23	23.0	0	0.0
KPIを設計し、成果を計測して改善を主導できる力	22	15.7	17	17.0	5	12.5
社内データをRAG/APIで連携して業務に実装できる力	18	12.9	13	13.0	5	12.5
その他 (※)	5	3.6	4	4.0	1	2.5
エンジニアは必要ない	13	9.3	9	9.0	4	10.0

(N=全体140・PD100・MC40、MA3)

※〈その他〉の例：「特定業界の業務領域を深く理解する力」「海外拠点・顧客と協働できる英語力」「プロジェクト収支を理解して判断できる力」「レガシーと新技術を橋渡しする力」「自律的に学び続けて周囲に展開する力」

## 2.2.4 調査考察

### 導入は広がるが“限定導入”が主流

回答企業は、小規模が中心で、常用雇用者100人未満（78.6%）が多数である。この規模感では、生成AIの導入は「全社展開（標準ツール化）」より「限定導入（個人・有志）」（20.0%）として始まり易い。他方で、「未導入（予定なし）」「検討中（情報収集中・環境整備中）」（45.0%）が尚も厚く、導入の意思決定が先送りされている。その結果として、ツールの話は広がるが、組織の標準として“使える状態”に固定され難い（5.7%）。

その際、実際の業務では「生成AI活用業務がない」（31.4%）という回答も3割程度に達し、導入議論が現場の案件に落ちていない企業が残る。「生成AI活用案件がない」（40.0%）も4割を占め、活用が“個人の試行”から“会社の案件”に転換していない実態が顕れている。

そして、人口減少県（PD）では「未導入（予定なし）」が高い（31.0%）のに対し、首都圏中核都県（MC）では「全社展開（標準ツール化）」が高い（12.5%）。「現在利用中（導入検討中）」の生成AI関連ツール・サービスがない」もPDで高く、入口の時点で差が開いている（PD33.0%・MC12.5%）。

### 活用は文書・コーディングに偏り、DevOps・運用自動化が空白

活用業務は「文書作成・ナレッジ管理」（39.3%）が最多で、次いで「システム開発・実装」（32.9%）が続く。実施作業でも「コーディング補完・リファクタ」（28.6%）と「ドキュメント／設計書生成・整備」（22.9%）が上位を占め、生成AIは“書く”作業の省力化から浸透している。この傾向は、生成AIがまず“成果物を増やす道具”として受け入れられていることを示す。

しかし、開発サイクルで定着している工程は「開発・実装」に偏り、「セキュリティレビュー」「デプロイ／リリース管理」「運用監視・インシデント対応」では定着を確認できない。「運用改善・自動化」（7.9%）も定着が弱く、要件定義から運用改善までを一気通貫で回す段階に届いていない。この“運用の空白”は、本事業が掲げる“要件定義→実装→運用自動化”を自走させる人材像に照すと、最も戦略的に埋めるべきギャップである。

さらに注目すべき点は、作業として「コーディング補完・リファクタ」（28.6%）を挙げる企業が一定数いるのに対し、専用のコーディング支援である「GitHub Copilot (GitHub）」（6.4%）の利用は限定的である。ここに、“現場はコード生成をやっているのに、安全かつ効率的な実装形態（IDE統合・権限制御・ログ）に乗っていない”というねじれがある。ねじれが放置されるほど、貼り付け型運用が常態化し、セキュリティ・品質・再現性の問題が増幅し易い。

### 最大ボトルネックは“安心して使える条件”であり、ガバナンス設計が追い付いていない

導入・運用上のボトルネックは、「セキュリティ・法令順守の懸念」（47.9%）が突出して高い。それにも拘らず、利用ルールの整備は「注意喚起のみ」「未整備」（45.0%）が半数近くを占め、安心の前提条件が組織内に構築されていない。文書化された指針を持つ層（19.9%）も2割程度に留まり、ルール不在のまま現場利用のみが進む構図が透ける。

データ入力方針でも「利用禁止」「共通の標準なし、案件毎に都度判断（統一ルールは未整備）」（30.7%）が3割を超え、生成AIの価値源泉である業務データを扱い難い。機密データを自社管理環境で扱える層（5.7%）は

ごく少数であり、RAGやAPI連携に踏み込めない企業が多数派である。ここで必要なものは、“プロンプトの巧拙”より先に、“どのデータをどの環境で、誰がどう承認し、どう記録するか”を設計できる力である。将来に重視される能力として「AIリスクを評価し、規程に沿って安全運用できる力」(32.9%)が3割を超えるのは、この設計力の欠如が現場の痛点になっているためである。

### 効果は小さくないが、測れていないためにPoCから先に進み難い

業務時間への影響は「1～9%向上」(18.6%)と「10～19%向上」(14.3%)が中心であり、劇的改善は観測されない。ただし、少しでも何らかの短縮を実感している層(35.0%)は3割を超え、生成AIが現場のボトルネックを確かに削っている。他方で、「その他(各業務で短縮と増加が混在、評価中等)」(16.4%)も一定数あり、効果測定の設計が未成熟である。

開発品質への影響は更に不透明で、「評価不可(測定未実施)」(23.6%)が4分の1近くに達する。品質改善が10%以上と答える層(12.2%)は1割強に留まり、品質を“上げる”前に品質を“測る”設計が欠けている。この“測れない”は、PoCが永遠にPoCで終る典型的な構造であり、投資判断も標準化判断も進まない。そのため、本事業においてBASISやGPT-TA評価システムのような到達度評価と学習ログを設計に組み込むことは、教育の評価手段に留まらず、企業側の導入を“検討→運用”に押し上げる実装モデルになり得る。

### 求められる人材像は“AI操作が上手い人”でなく“要件・合意・リスクを訳せる人”

不足・強化したいスキルは「要件定義・アーキテクチャ設計」(38.6%)が最上位であり、生成AI時代でも“設計の骨格”が最重要課題である。将来に重視される能力も「業務要件を構造化して実装可能な設計に落とし込める力」(37.9%)が最多であり、AIがコードを書けても、何を書くべきかを決める能力が不足している。それと同時に、「目的を言語化し、合意を形成して協働を推進できる力」(36.4%)も拮抗して高い。ここには、生成AIが“言語で動く”という技術条件が、人材要件を変えている在り様が顕れている。すなわち、言語化は、単なるソフトスキルでなく、要件をプロンプトに変換し、成果物に落とし込むための中核技術になっている(21.4%)。

リスク面では、「セキュリティ設計・法令・ライセンス対応」(29.3%)が不足スキルの上位にあり、将来に重視される能力としても安全運用と脅威想定が強く求められる(32.9%)。そのため、AI補完型ITエンジニアは“AIで速く作る人”でなく、“速さを出しながら事故にならない条件を設計できる人”として定義し直す必要がある。

### 教育設計はオンデマンド×伴走のハイブリッドが適合

有効性が高い育成手法は「オンデマンドeラーニング(チェックテスト付)」(38.6%)と「ハイブリッド講義(オンライン+対面)」(32.1%)が上位であり、時間制約が強い現場に適合した分散学習が求められる。他方で、「人材育成は実施しない(できない)」(16.4%)も一定数あり、社内だけで学習機会を確保できない企業が残る。特にPDではこの割合が高く、人口減少地域ほど“学べないこと”が課題になる(21.0%)。

そのため、本事業で開発するカリキュラムは、オンデマンドで基礎を高速に揃え、短期ハンズオン(24.3%)で生成AI×DevOpsの型を身体化し、メンタリング(21.4%)で自社課題に接続して定着させる“三段構え”で設計することが合理的である。この設計により、現場の「限定導入」を組織の運用に引き上げ、人口減少地域の小規模チームでも“要件定義→実装→運用自動化”を回せる状態に移行させる。

## 2.3 調査② IT “学び直し” ニーズ実態調査

調査②はIT “学び直し” ニーズ実態調査である。

これは、人口減少地域の若年層・社会人におけるIT “学び直し” ニーズの実情を把握することで、本教育プログラムの想定対象を明かにし、開発の基礎資料にするためのものである。

その調査仕様や調査結果を以下に報告する。

### 2.3.1 調査仕様

本調査の**仕様**（目的・対象・手法・内容・反映方針等）は次の通りである

名称	IT “学び直し” ニーズ実態調査
目的	人口減少地域の若年層・社会人におけるIT “学び直し” ニーズの実情を把握することで、本教育プログラムの想定対象を明かにし、開発の基礎資料にする。
対象	人口減少県の中小企業管理職100人（+首都圏中核都県の中小企業管理職40人）➡ 有効回答140件
手法	ネットリサーチ
項目	ITスキル自己評価、“学び直し” 経験有無、志望分野、学習障壁、希望受講形態、受講時間余力 等

### 2.3.2 調査内容

本調査における具体的な**質問項目**は次の10項である。

回答者企業における――

- ① 18～45歳層のIT基礎スキル水準
- ② 不足が目立つ基礎スキル
- ③ 今後12箇月で“学び直し”を希望（必要とする）者の割合
- ④ “学び直し”で優先度が高い内容
- ⑤ “学び直し”の到達目標水準
- ⑥ “学び直し”を希望する理由
- ⑦ “学び直し”に取り組む際の主な障壁
- ⑧ 企業として許容できる支援（時間付与・費用補助等）
- ⑨ 週当りの確保可能な学習時間
- ⑩ 標準的に取り組める学習期間

これに基づいて作成した**調査票**を次頁に掲載する。

## ▼【調査票】調査② IT “学び直し” ニーズ実態調査

令和7年度 文部科学省委託 地域産業中核の人材養成事業——人口減少地域の職業人材を確保するための専修学校振興プログラム  
人口減少地域を支えるAI補完型ITエンジニア養成プログラムの開発

### IT“学び直し”ニーズ実態調査

この調査は、人口減少県および首都圏中核都県に所在する中小企業におけるIT “学び直し” ニーズの実情を把握することで、本教育プログラムの想定対象を明かにし、当該開発の基礎資料にすることを目的とするものです。

**問01** 貴社の従業員（18～45歳）の平均的なIT基礎スキルの水準として該当するものは、次のうちどれですか。  
[1つ選択]

- |                          |                              |
|--------------------------|------------------------------|
| ① 初歩レベル（基本用語を理解できる程度）    | ② 基礎レベル（簡単な設定や自動化を行える程度）     |
| ③ 実務初級レベル（補助があれば実装できる程度） | ④ 実務中級レベル（標準に沿って実装や改修を行える程度） |
| ⑤ 実務上級（設計やレビューを主導できる程度）  | ⑥ その他（ ）                     |
| ⑦ ITスキルは必要ない             |                              |

**問02** 貴社の従業員（18～45歳）で不足が目立つ基礎スキルとして該当するものは、次のうちどれですか。  
[上位2つまで選択]

- |                     |                   |
|---------------------|-------------------|
| ① 仕様読解・要約の基礎力       | ② 論理設計・アルゴリズムの基礎力 |
| ③ 数理・統計の基礎力         | ④ 英語ドキュメントの読解力    |
| ⑤ バージョン管理・チーム開発の運用力 | ⑥ 設計書・手順書の作成力     |
| ⑦ トラブルシューティングの対応力   | ⑧ セキュリティの基礎知識     |
| ⑨ コミュニケーション・報連相の基礎力 | ⑩ その他（ ）          |
| ⑪ 不足した基礎スキルはない      |                   |

**問03** 貴社の従業員（18～45歳）のうち、今後12箇月でIT “学び直し” を希望する（or必要とする）者の割合として該当するものは、次のうちどれですか。[1つ選択]

- |          |          |          |          |
|----------|----------|----------|----------|
| ① 1～9%   | ② 10～19% | ③ 20～29% | ④ 30～39% |
| ⑤ 40～49% | ⑥ 50～59% | ⑦ 60～79% | ⑧ 80%以上  |
| ⑨ 該当者なし  |          |          |          |

**問04** 貴社の従業員（18～45歳）がIT “学び直し” を希望する（or 必要とする）内容として優先度が高いものは、次のうちどれですか。[上位3つまで選択]

- |                                 |                                 |
|---------------------------------|---------------------------------|
| ① プログラミング基礎・アルゴリズム              | ② 開発言語（Python・Java・TypeScript等） |
| ③ Webアプリ設計・API開発                | ④ データベース・SQL                    |
| ⑤ クラウド基礎（AWS・Azure・GCP等）        | ⑥ DevOps・CI・CD・Git              |
| ⑦ テスト自動化・品質保証                   | ⑧ セキュリティ・法令・ライセンス               |
| ⑨ データ分析・機械学習・生成AI               | ⑩ その他（ ）                        |
| ⑪ 希望する（or 必要とする）IT “学び直し” 内容はない |                                 |

**問05** 貴社の従業員（18～45歳）のIT “学び直し” における到達目標として最も妥当と思える水準は、次のうちどれですか。[1つ選択]

- |                               |                   |
|-------------------------------|-------------------|
| ① 基礎操作（環境準備・設定 等）を行える程度       |                   |
| ② 既存手順に沿った修正・補助作業を行える程度       |                   |
| ③ 小規模機能を単独で実装できる程度            |                   |
| ④ コード品質・テストを含めて小規模機能を完結できる程度  |                   |
| ⑤ チーム内の機能実装を主担当できる程度          |                   |
| ⑥ 既存システムの改修・リファクタを計画的に推進できる程度 |                   |
| ⑦ 設計とレビューを担当できる程度             |                   |
| ⑧ 要件定義・設計方針を主導できる程度           |                   |
| ⑨ その他（ ）                      | ⑩ 目標設定なし（or 実施なし） |



## 2.3.3 調査結果

本調査の結果は次の通りである

【調査対象】人口減少県の中小企業100社および首都圏中核都県の中小企業40社

### 「人口減少県」の選定過程

- ① 中長期の減少傾向： 2015年→2020年（直近の国勢調査区間）で人口が減った都道府県
- ② 直近の総人口増減： 2024年10月時点の人口増減率で減少を確認し、減少率1.0%以上を「重症県」と定義
- ③ 社会増減の符号： 2024年の〈自然減+社会減（転出超過）〉（二重減少）に該当する県を「人口減少県」と定義

その結果、下掲の23県を「人口減少県」とし、このうち下線付きの18県を「重症県」とした。

本調査の回答者は、この「人口減少県」に所在する中小企業100社の管理職である（各件の（ ）内は回答者数）。

[東北] 青森県 (4)・岩手県 (4)・秋田県 (4)・山形県 (4)・福島県 (6)  
 [北陸] 新潟県 (9)・富山県 (4)・石川県 (6) [東海・近畿] 三重県 (3)・和歌山県 (3)  
 [中国] 鳥取県 (3)・島根県 (3)・岡山県 (6)・広島県 (9)・山口県 (3)  
 [四国] 徳島県 (3)・香川県 (3)・愛媛県 (5)・高知県 (3)  
 [九州] 佐賀県 (2)・長崎県 (4)・大分県 (4)・鹿児島県 (5)

本調査では、「人口減少県」の回答結果と比較すべく、「首都圏中核都県」に所在する「学校法人電子学園 後援会」  
**会員企業40社**（東京都 33社・神奈川県 7社）からも回答を得た。  
 その際、略記として、「人口減少県」を**PD** (Population-Declining Prefectures)、「首都圏中核都県」を**MC** (Metropolitan Core Prefectures) と表記する。

問01 貴社の従業員（18～45歳）の平均的なIT基礎スキルの水準として該当するものは、次のうちどれですか。

[1つ選択]

選択肢	全体		PD		MC	
	n	%	n	%	n	%
初歩レベル（基本用語を理解できる程度）	15	10.7	14	14.0	1	2.5
基礎レベル（簡単な設定や自動化を行える程度）	30	21.4	27	27.0	3	7.5
実務初級レベル（補助があれば実装できる程度）	34	24.3	26	26.0	8	20.0
実務中級レベル（標準に沿って実装や改修を行える程度）	37	26.4	21	21.0	16	40.0
実務上級（設計やレビューを主導できる程度）	11	7.9	6	6.0	5	12.5
その他（※）	6	4.3	4	4.0	2	5.0
ITスキルは必要ない	7	5.0	2	2.0	5	12.5

(N=全体140・PD100・MC40、SA)

※〈その他〉の例：「職種間スキル差が大きく平均化が困難」「拠点毎にスキル水準が大きく異なる」「若手とベテランでスキルが二極化」「協力会社依存のため自社だけで評価が困難」「少数の高スキル人材に依存」等

問02 貴社の従業員（18～45歳）で不足が目立つ基礎スキルとして該当するものは、次のうちどれですか。

[2つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
コミュニケーション・報連相の基礎力	53	37.9	31	31.0	22	55.0
仕様読解・要約の基礎力	38	27.1	33	33.0	5	12.5
論理設計・アルゴリズムの基礎力	38	27.1	29	29.0	9	22.5
英語ドキュメントの読解力	37	26.4	27	27.0	10	25.0
トラブルシューティングの対応力	29	20.7	21	21.0	8	20.0
バージョン管理・チーム開発の運用力	28	20.0	24	24.0	4	10.0
設計書・手順書の作成力	25	17.9	23	23.0	2	5.0
数理・統計の基礎力	22	15.7	19	19.0	3	7.5
セキュリティの基礎知識	21	15.0	17	17.0	4	10.0
その他（※）	10	7.1	7	7.0	3	7.5
不足した基礎スキルはない	10	7.1	9	9.0	1	2.5

(N=全体140・PD100・MC40、MA2)

※〈その他〉の例：「スケジュール・タスク管理の基礎力」「日本語での説明・文章作成力」「オンライン会議・チャットの基本マナー」「Excel等による基本的なデータ分析力」「顧客要件のヒアリング・整理力」等

問03 貴社の従業員（18～45歳）のうち、今後12箇月でIT“学び直し”を希望する（or必要とする）者の割合として該当するものは、次のうちどれですか。[1つ選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
1～9%	15	10.7	11	11.0	4	10.0
10～19%	17	12.1	13	13.0	4	10.0
20～29%	23	16.4	17	17.0	6	15.0
30～39%	20	14.3	16	16.0	4	10.0
40～49%	12	8.6	12	12.0	0	0.0
50～59%	12	8.6	9	9.0	3	7.5
60～79%	8	5.7	7	7.0	1	2.5
80%以上	8	5.7	6	6.0	2	5.0
該当者なし	25	17.9	9	9.0	16	40.0

(N=全体140・PD100・MC40、SA)

問04 貴社の従業員（18～45歳）がIT“学び直し”を希望する（or 必要とする）内容として優先度が高いものは、次のうちどれですか。[3つまで選択]

選択肢	全体		PD		MC	
	n	%	n	%	n	%
プログラミング基礎・アルゴリズム	44	31.4	37	37.0	7	17.5
開発言語（Python・Java・TypeScript等）	42	30.0	33	33.0	9	22.5
クラウド基礎（AWS・Azure・GCP等）	41	29.3	29	29.0	12	30.0
データベース・SQL	34	24.3	31	31.0	3	7.5
セキュリティ・法令・ライセンス	34	24.3	24	24.0	10	25.0
データ分析・機械学習・生成AI	31	22.1	17	17.0	14	35.0
Webアプリ設計・API開発	30	21.4	27	27.0	3	7.5
DevOps・CI・CD・Git	26	18.6	23	23.0	3	7.5
テスト自動化・品質保証	23	16.4	21	21.0	2	5.0
その他（※）	13	9.3	9	9.0	4	10.0
希望する（or 必要とする）IT“学び直し”内容はない	20	14.3	11	11.0	9	22.5

(N=全体140・PD100・MC40、MA3)

※〈その他〉の例：「プロジェクトマネジメントの体系的学習」「UI/UXデザイン・プロトタイプング」「特定業界の業務知識（医療・製造等）」「英語による技術情報収集・発信」「チームマネジメント・リーダーシップ」等

問05 貴社の従業員（18～45歳）のIT“学び直し”における到達目標として最も妥当と思える水準は、次のうちどれですか。[1つ選択]

選択肢	全体		PD		MC	
	n	%	n	%	n	%
基礎操作（環境準備・設定等）を行える程度	18	12.9	13	13.0	5	12.5
既存手順に沿った修正・補助作業を行える程度	13	9.3	11	11.0	2	5.0
小規模機能を単独で実装できる程度	23	16.4	19	19.0	4	10.0
コード品質・テストを含めて小規模機能を完結できる程度	21	15.0	18	18.0	3	7.5
チーム内の機能実装を主担当できる程度	23	16.4	16	16.0	7	17.5
既存システムの改修・リファクタを計画的に推進できる程度	3	2.1	3	3.0	0	0.0
設計とレビューを担当できる程度	9	6.4	6	6.0	3	7.5
要件定義・設計方針を主導できる程度	8	5.7	3	3.0	5	12.5
その他（※）	3	2.1	2	2.0	1	2.5
目標設定なし（or 実施なし）	19	13.6	9	9.0	10	25.0

(N=全体140・PD100・MC40、SA)

※〈その他〉の例：「外部資格試験合格レベル」「特定プロジェクトで即戦力として稼働できる程度」「上長と合意した個別目標水準」

問06 貴社の従業員（18～45歳）が“学び直し”を希望する理由として該当するものは、次のうちどれですか。

[2つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
業務効率・生産性の向上	51	36.4	41	41.0	10	25.0
新技術のキャッチアップ	40	28.6	27	27.0	13	32.5
自信不足の解消、基礎固め	33	23.6	29	29.0	4	10.0
市場価値の向上、将来不安の低減	31	22.1	23	23.0	8	20.0
顧客要件・監査対応の強化	28	20.0	21	21.0	7	17.5
配置転換・役割拡張への備え	21	15.0	19	19.0	2	5.0
昇進・評価・賃金の向上	14	10.0	9	9.0	5	12.5
チーム内影響力の向上	7	5.0	7	7.0	0	0.0
社内評価の可視化	6	4.3	6	6.0	0	0.0
その他（※）	5	3.6	4	4.0	1	2.5
“学び直し”希望なし	27	19.3	17	17.0	10	25.0

(N=全体140・PD100・MC40、MA2)

※〈その他〉の例：「自社の事業転換・再編に備えるため」「自社プロダクト開発部門への異動希望のため」「海外案件・海外赴任の機会に備えるため」「リモート勤務を継続できるようにするため」「将来の独立・フリーランスを視野に入れて」

問07 貴社の従業員（18～45歳）が“学び直し”に取り組む際の主な障壁として該当するものは、次のうち

どれですか。[3つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
時間の不足（業務都合）	88	62.9	63	63.0	25	62.5
継続（モチベーション維持）の困難	48	34.3	31	31.0	17	42.5
基礎知識の不足	47	33.6	37	37.0	10	25.0
時間の不足（私事都合）	37	26.4	29	29.0	8	20.0
受講費用の負担	30	21.4	27	27.0	3	7.5
学習方法の不明	26	18.6	23	23.0	3	7.5
上長の理解・承認の不足	24	17.1	21	21.0	3	7.5
適切な教材・講座の不足	24	17.1	19	19.0	5	12.5
社内制度・手続きの制約	19	13.6	17	17.0	2	5.0
その他（※）	4	2.9	4	4.0	0	0.0
障壁なし（or 取組なし）	18	12.9	11	11.0	7	17.5

(N=全体140・PD100・MC40、MA3)

※〈その他〉の例：「学び直しが評価・昇格と連動していない」「学ぶべき分野が多く優先順位を決め難い」「勤務シフト上、研修日程を合せ難い」「地方在住で対面研修会場が遠い」

問08 貴社において従業員（18～45歳）の“学び直し”に対応・許容できる支援として該当するものは、次のうちどれですか。[2つまで選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
受講費の一部補助	41	29.3	29	29.0	12	30.0
資格試験の受験補助	35	25.0	23	23.0	12	30.0
就業時間内での学習時間の付与	27	19.3	19	19.0	8	20.0
成果の評価・昇格への反映	23	16.4	11	11.0	12	30.0
学習アカウント・環境の提供	22	15.7	17	17.0	5	12.5
受講費の全額補助	17	12.1	7	7.0	10	25.0
異動・OJTでの実務適用	9	6.4	9	9.0	0	0.0
休暇・時短の付与	6	4.3	6	6.0	0	0.0
その他（※）	6	4.3	4	4.0	2	5.0
社内メンターの伴走支援	3	2.1	3	3.0	0	0.0
支援なし	31	22.1	27	27.0	4	10.0

(N=全体140・PD100・MC40、MA2)

※〈その他〉の例：「技術書購入・資格受験費の会社補助」「社内勉強会主催者への報奨・評価加点」「学習期間中の業務量・残業の調整」「外部コミュニティ参加時間の勤務扱い」「OSS活動や技術発信を評価項目に含める」等

問09 貴社の従業員（18～45歳）が“学び直し”に充当できる1週間当りの学習時間として最も妥当と思えるものは、次のうちどれですか。[1つ選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
15分以上 1時間未満	27	19.3	21	21.0	6	15.0
1時間以上 2時間未満	33	23.6	24	24.0	9	22.5
2時間以上 3時間未満	26	18.6	18	18.0	8	20.0
3時間以上 5時間未満	19	13.6	13	13.0	6	15.0
5時間以上 8時間未満	11	7.9	7	7.0	4	10.0
8時間以上 10時間未満	3	2.1	3	3.0	0	0.0
10時間以上 15時間未満	1	0.7	1	1.0	0	0.0
15時間以上	0	0.0	0	0.0	0	0.0
学習時間を確保できない	20	14.3	13	13.0	7	17.5

(N=全体140・PD100・MC40、SA)

問10 貴社の従業員（18～45歳）が“学び直し”に取り組める標準的な学習期間として最も妥当と思えるものは、次のうちどれですか。[1つ選択]

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
1～2日間	7	5.0	4	4.0	3	7.5
3～5日間	12	8.6	7	7.0	5	12.5
1週間	9	6.4	6	6.0	3	7.5
2週間	5	3.6	4	4.0	1	2.5
1箇月間	22	15.7	18	18.0	4	10.0
2箇月間	15	10.7	14	14.0	1	2.5
3箇月間	21	15.0	16	16.0	5	12.5
6箇月間	17	12.1	11	11.0	6	15.0
1.5～2年間	2	1.4	2	2.0	0	0.0
2年間超	1	0.7	1	1.0	0	0.0
期間設定なし (or 関知しない)	29	20.7	17	17.0	12	30.0

(N=全体140・PD100・MC40、SA)

\*

[回答者役職] 回答者の貴社における役職として該当するものは、次のうちどれですか。[1つ選択]

※ 重複する役職がある場合、日常業務において最も近いものを選んでください。

選択肢	全 体		PD		MC	
	n	%	n	%	n	%
① 経営層（代表・役員）	63	45.0	47	47.0	16	40.0
② 本部長・事業部長・部長	27	19.3	19	19.0	8	20.0
③ 課長・マネジャー	31	22.1	21	21.0	10	25.0
④ 係長・主任・チームリーダー	9	6.4	7	7.0	2	5.0
⑤ 非管理職（一般社員）	8	5.7	4	4.0	4	10.0
⑥ その他（「グループ会社からの出向責任者」「外部技術顧問」）	2	1.4	2	2.0	0	0.0

(N=全体140・PD100・MC40、SA)

## 2.3.4 調査考察

### “学び直し” 需要は二極化し、人口減少県の企業でより厚い

今後12箇月で“学び直し”希望 (or 必要) 者比率が「20～29%」以上 (59.3%) と見積もる企業が過半を占め、“学び直し”が一部の“熱心層”に閉じた話でないことが判る。他方で、「該当者なし」(17.9%)も一定割合を占め、企業内で需要が立ち上がらないケースも併存する。

この二極化は、地域で更に強まる。首都圏中核都県 (MC) では「該当者なし」(40.0%)が突出して高く、需要が“立ち上がらない企業”が大きな塊として存在する。それに対し、人口減少県 (PD) では同選択 (9.0%)が低く、“学び直し”が“点”でなく“面”として要請される。この差は、教育設計を“個別最適の小規模実装”と“一括展開の量産”に分岐させる必要性を示す。

また、“学び直し”の理由でも、「“学び直し”希望なし」(19.3%)が2割近く、需要の非連続性を裏付ける。需要がある層に“届ける”前に、需要が立ち上がらない層に“立ち上げる”設計が別建てで求められる。

### 既存スキルは“実務初級～中級”が中核で、PDとMCの勾配が大きい

従業員 (18～45歳) の平均的IT基礎スキル水準は、「実務中級レベル」(26.4%)が最多であるが、「実務初級レベル」(24.3%)も同時に厚く、実装フェーズに片足が掛る層は過半に達する。しかし、「初歩レベル」(10.7%) + 「基礎レベル」(21.4%)も3割を超え、実務に届かない層も相当に厚い。

地域差は明確である。MCでは「実務中級レベル」(40.0%)が高く、相対的に実務側に寄った組織像が見えるのに対し、PDでは「基礎レベル」(27.0%)が高く、“学び直し”の入口がより手前にある。また、MCでは「ITスキルは必要ない」(12.5%)も一定割合が存在し、職種横断の一律教育が空振りするリスクがある。

そのため、“学び直し”は、「初歩～基礎」からの引き上げと、「実務初級～中級」からの底上げを同時に成立させる“二段設計”が要る。単一カリキュラム“一本槍”では、どちらかが“置き去り”になる。

### 不足が目立つのは技術より“言語化・読解・協働”の基礎

不足が目立つ基礎スキルの筆頭が「コミュニケーション・報連相の基礎力」(37.9%)であり、技術以前の“仕事の運び”がボトルネックになっている。次いで「仕様読解・要約の基礎力」(27.1%)と「論理設計・アルゴリズムの基礎力」(27.1%)が並び、入力 (読解) と構造化 (論理) の同時弱体が見える。

ここで重要なことは、“学び直し”を「技術習得」と「業務遂行」の二層で設計することである。報連相や読解・要約が弱い状態では、学習が継続し難く、学習内容が現場で“使える形”に変換されない確率が高い (「仕様読解・要約の基礎力」27.1%、「継続 (モチベーション維持) の困難」34.3%)。逆に言えば、ここは短時間でも改善が見え易い領域である。要約・説明文作成・レビュー補助等、AI支援を“学習の補助輪”として組み込めば、技術教材の理解速度を押し上げ、継続障壁の一部を構造的に下げられる。

また、MCで「コミュニケーション・報連相の基礎力」(55.0%)が特に高い点は示唆的である。都市部ほど技術者の流動性が高く、チーム協働の品質が差別化になる一方で、その基礎が不足として認識され易い構図が見える。“学び直し”を“個人のスキルアップ”で閉じず、チーム協働の生産性改善として設計する視点が求められる。

## “学び直し”内容は基礎回帰が強く、AI/データは上積み需要として顕れ

優先度が高い“学び直し”内容は「プログラミング基礎・アルゴリズム」(31.4%)・「開発言語」(30.0%)・「クラウド基礎」(29.3%)が上位を占め、まづ“作れる土台”を求める傾向が強い。ここに「データベース・SQL」(24.3%)や「セキュリティ・法令・ライセンス」(24.3%)が続き、実務の基本パーツを固めたい意図が見える。

他方で、「データ分析・機械学習・生成AI」(35.0%)も一定の優先度を持ち、特にMCで高い。PDでは同領域(17.0%)が相対的に低く、基礎固めを優先する傾向が強い。この差は、AI/データを“全員必修”として押し付けるより、共通コア(基礎言語・クラウド・セキュリティ)にAI活用を自然に織り込み、AI/データは選択トラックとして厚く提供の方が合理的であることを示す。

また、「希望する(or必要とする)内容はない」(14.3%)も一定割合あり、“学び直し”テーマの設定自体が空転している企業が存在する。この層には、教材を増やす前に“何を学べば業務が短縮されるか”を診断し、テーマ設定を外部から補助する仕掛けが求められる。

## 最大障壁は時間で、“学び直し”は時間パラドックスに陥る

“学び直し”希望の理由は「業務効率・生産性の向上」(36.4%)が最多である一方で、障壁は「時間の不足(業務都合)」(62.9%)が突出して高い。“時間を増やすために学ぶのに、学ぶ時間がない”という矛盾がここにある。そのため、教育設計は、学習を“追加タスク”として積むのではなく、学習によって当日の業務が即短縮される体験を先に置く必要がある。

実際、学習時間は「15分以上 1時間未満」(19.3%)と「1時間以上 2時間未満」(23.6%)が4割強を占め、週2時間未満が標準である。「2時間以上 3時間未満」(18.6%)まで含めると週3時間未満が6割を超え、纏った学習を前提にし難い。この条件下で「継続(モチベーション維持)の困難」(34.3%)が大きな障壁になるのは自然である。短時間で達成感を得られる単位設計と、伴走・コミュニティ等による継続支援が成否を分ける。

学習期間は「1箇月間」(15.7%)・「2箇月間」(10.7%)・「3箇月間」(15.0%)が中心で、短い学習時間で回しながら中期で変化を作るニーズが強い。他方で、「期間設定なし(or関知しない)」(20.7%)も2割を超え、“学び直し”が“気分と都合”で途切れ易い構造が残る。ここは、到達目標の分散にも連動する。到達目標は「小規模機能を単独で実装できる程度」(16.4%)と「チーム内の機能実装を主担当できる程度」(16.4%)が同率で最多であるが、いずれも2割未満に留まり、企業内で標準目標が定まっていない。“学び直し”を制度に変えるためには、目標の言語化と進捗の可視化を外部から補助する設計が求められる。

企業に対応・許容可能な“学び直し”支援は、最多である「受講費の一部補助」(29.3%)さえ3割未満で、そもそも「支援なし」(22.1%)が2割を超える。特にPDでは「支援なし」(27.0%)が高く、同時に障壁として「受講費用の負担」(27.0%)が挙り易い。需要が厚い地域ほど支援余力が薄いという逆転が起きている以上、“学び直し”を“個人の自助努力”に回収するモデルは成立し難い。低コスト・短時間で回せる学習体験を標準装備にしつつ、企業には制度設計テンプレートや評価連動の仕組みまで含めて提供する必要がある。

## 2.4 調査③ 生成AIコーディングアシスタント事例調査

調査③は**生成AIコーディングアシスタント事例調査**である。

これは、国内外で利用されている生成AIコーディングアシスタントの事例や機能・導入形態等に関する情報を収集・整理することで、本教育プログラムの演習環境要件を明かにし、開発の基礎資料にするためのものである。

その調査仕様や調査結果を以下に報告する。

### 2.4.1 調査仕様

本調査の**仕様**（目的・対象・手法・内容・反映方針等）は次の通りである

名称	生成AIコーディングアシスタント事例調査
目的	国内外で利用されている生成AIコーディングアシスタントの事例や機能・導入形態等に関する情報を収集・整理することで、本教育プログラムの演習環境要件を明かにし、開発の基礎資料にする。
対象	各種公開資料（出版物・ウェブサイト等）
手法	デスクリサーチ
項目	ツール名・提供者・URL・提供形態・日本語対応・利用料・概要・主要機能・主な活用例・国内可用性（導入ハードル・利用難点等）

### 2.4.2 調査結果

本調査では、生成AIコーディングアシスタントの国内外の事例について、**デスクリサーチ**を実施した。

その結果として、**20事例**を確認した。その基本情報や仕様・特徴等を次頁以降に整理する。

▼【生成AIコーディングアシスタント】事例01

ツール	GitHub Copilot	
提供者	GitHub, Inc. (Microsoft 100%子会社)	
URL	<a href="https://github.com/features/copilot">https://github.com/features/copilot</a>	
提供形態	SaaS (IDE/CLI/エディタ拡張として提供)	
日本語対応	UIは英語中心 (IDE側の日本語ローカライズには依存)。設定変更でプルリクエストレビュー、コードレビュー機能、ウェブサイトも日本語対応。	
利用料	GitHub Global Campusに申し込むことで、学生 / 教師が無料 チーム：\$4/ユーザー/月、企業：\$21/ユーザー/月 通貨はUSD。	
概要	<p>GitHub Copilotは、IDE等に統合されるAIペアプログラマで、コード補完・生成・説明・テスト作成などを支援する。特筆点として、GitHub・Codespaces・PRレビュー等のGitHub製品連携と法人向けのIP保護/管理機能を提供し、組織導入に適する。業界では「汎用IDE統合の標準的選択肢」の一つとして広く普及している。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ コード補完と多行提案を行う。</li> <li>○ チャットでコード説明・修正方針提案・テスト生成を行う。</li> <li>○ CLI/PR/Docs連携で開発タスクを横断的に支援する。</li> <li>○ 個人／組織向けで利用制限と管理機能を切り替える。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 既存コードのテストケース自動生成・補完。</li> <li>○ PR要約やレビュー補助の効率化。</li> <li>○ 新規実装の骨子生成とドキュメント補助。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ GitHub環境前提のため、非GitHub運用では統合メリットが相対的に小さい。</li> <li>○ 管理者によるモデル利用・ログ管理ポリシー策定が必要。</li> <li>○ IDE/拡張の組織設定・権限設計に初期工数を要する。</li> </ul>	

▼【生成AIコーディングアシスタント】事例02

ツール	Amazon Q Developer	
提供者	Amazon Web Services, Inc.	
URL	<a href="https://aws.amazon.com/q/developer/">https://aws.amazon.com/q/developer/</a>	
提供形態	SaaS (IDE拡張/CLI/コンソール)	
日本語対応	UIとサポート：UIは英語中心（AWSコンソールは日本語表示可）。サポートはAWS Support 経由（契約プランに依存、地域対応あり）。	
利用料	Free Tier（恒久無料枠あり）：エージェント対話等の月間上限付き、 Pro：\$19/ユーザー/月（上限拡大・追加機能） 通貨はUSD。	
概要	Amazon Q Developerは、AWS/非AWS双方の開発でコード生成・修正・テスト・リファクタリング等を支援する。特筆点として、AWS各種サービスとの統合と“エージェント”による半自律的な作業を提供し、運用設計にも踏み込んで支援する。業界では、クラウド統合の強みと価格の明確さで選好される場面が多い。	
主要機能	<ul style="list-style-type: none"> <li>○ コード変換・生成・リファクタリングをIDE/CLIから行う。</li> <li>○ チャット／エージェントでデバッグや手順実行を支援する。</li> <li>○ リポジトリ・ドキュメント・設定からコンテキスト検索を行う。</li> <li>○ 料金/上限をFree/Proで明確に切替える。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 既存JavaアプリのAWS移行時のコード修正支援。</li> <li>○ IaCテンプレートやCI/CDの雛形生成。</li> <li>○ 既存障害の手順化とエージェントによる半自動化。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 企業ポリシーに応じたログ・データ取り扱いの設計が必要。</li> <li>○ AWS以外メインの開発組織では連携メリットが限定される。</li> <li>○ IDE/権限設定の初期調整と上限管理の運用が必要。</li> </ul>	


▼【生成AIコーディングアシスタント】事例03

ツール	GitLab Duo	
提供者	GitLab Inc.	
URL	<a href="https://about.gitlab.com/gitlab-duo/">https://about.gitlab.com/gitlab-duo/</a>	
提供形態	SaaS+セルフホスト（GitLab.com／Self-Managed／Dedicatedに付随提供）	
日本語対応	UIはGitLabの言語設定に依存（日本語UIありの領域もあるが英語優位）。サポートはGitLabのサポート契約（英語中心、上位プランで強化）。	
利用料	Premium：\$291/ユーザー/月（年一括払い）、アドオン体系のUltimate：詳細は見積 通貨はUSD。	
概要	GitLab Duoは、計画・実装・テスト・セキュリティ・デプロイまでのDevSecOps全体をAIで補助する。特筆点として、AI機能を“GitLab一体”で提供し、コード/チケット/セキュリティスキャンの文脈横断で支援する。業界では「プラットフォーム内完結のAIアシスト」として評価され、規模の大きいSaaS/自社ホスト双方で導入される。	
主要機能	<ul style="list-style-type: none"> <li>○ コード提案・説明・テスト生成を行う。</li> <li>○ セキュリティ診断・修正案提示をパイプラインに組み込む。</li> <li>○ Issue/MR/リポジトリ全体を横断するチャット/コンテキスト検索を行う。</li> <li>○ Self-Managed/Dedicated等で運用形態を選択できる。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ MRレビュー時の説明・要約・修正提案。</li> <li>○ 脆弱性検出と修正パッチ案の自動提示。</li> <li>○ Issueからのブランチ作成と実装雛形生成の迅速化。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 既存GitLab運用での権限・情報露出設計が必要。</li> <li>○ アドオン課金・サブスク階層の理解と見積調整が必要。</li> <li>○ モデル/データ取り扱いポリシーを組織基準に合わせる必要。</li> </ul>	

▼【生成AIコーディングアシスタント】事例04

ツール	JetBrains AI Assistant	
提供者	JetBrains s.r.o.	
URL	<a href="https://www.jetbrains.com/ai-assistant/">https://www.jetbrains.com/ai-assistant/</a>	
提供形態	SaaS (IDEプラグインとして提供)	
日本語対応	2024.2以降より日本語対応。	
利用料	<p>無料、            AI Pro：14000円/ユーザー/年、            AI Ultimate：42000円/ユーザー/年、            AIエンタープライズ：100800円/ユーザー/年等個人・組織で契約形態あり。</p>	
概要	<p>JetBrains AIは、IntelliJ系IDEに統合されたAI支援で、補完・生成・コード解説・リファクタリング等を提供する。特筆点として、JetBrains製の解析基盤とIDE内部メタ情報を深く活用して文脈理解に優れる。業界では、「JetBrains利用者に最適化されたAI」としてIDEネイティブ体験が評価される。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ エディタ内チャットでコード説明・修正を行う。</li> <li>○ 自動補完・ドキュメント生成・テスト支援を行う。</li> <li>○ IDEプロジェクト情報（型・依存等）を活用して精度を高める。</li> <li>○ プラン別にAIクレジットや機能上限を制御する。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 複雑なリファクタリング前の影響範囲説明。</li> <li>○ テストコード雛形の自動生成。</li> <li>○ 既存コードの技術負債箇所のガイド付き修正。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 既存JetBrains IDEの運用前提での導入コスト。</li> <li>○ モデル利用のクレジット制・課金モデルの理解が必要。</li> <li>○ 企業ポリシーに沿ったプライバシー設定の検討が必要。</li> </ul>	

▼【生成AIコーディングアシスタント】事例05

ツール	Sourcegraph Cody	
提供者	Sourcegraph, Inc.	
URL	<a href="https://sourcegraph.com/cody">https://sourcegraph.com/cody</a>	
提供形態	SaaS／セルフホスト（Enterpriseで自社運用可）	
日本語対応	<p>UIは英語。 サポートはエンタープライズ契約に付随（英語中心）。日本語での問い合わせサポートあり。</p>	
利用料	<p>アンプ：詳細は見積、 エンタープライズスターター：\$19/ユーザー/月（最大50人の開発者）、 エンタープライズ：\$49/ユーザー/月（25人以上の開発者） 通貨はUSD。</p>	
概要	<p>Codyは、Sourcegraphのコードグラフを活用し、巨大リポジトリ横断での正確なコンテキスト把握に強みを持つAIコーディングアシスタントである。特筆点として、公開コードとのマッチ検査（ガードレール）等の企業向けコンプライアンス機能を備え、著作権リスク低減に配慮する。業界では“大規模コードベースに強い企業向け選択肢”として位置付けられる。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ コード検索/セマンティック検索と結合した生成・修正を行う。</li> <li>○ 公開コードとの類似性検査で著作権リスクを低減する。</li> <li>○ セルフホスト/シングルテナント等でデプロイ選択を可能にする。</li> <li>○ 主要IDE対応の拡張で開発体験を統一する。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ モノレポ/多数サービス横断の仕様探索と変更影響分析。</li> <li>○ ガバナンス要件下での安全な自動補完/生成。</li> <li>○ サポート/SEのインシデント自己解決支援。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 自社ホストやモデル選定を含む初期設計コスト。</li> <li>○ 価格とLLM選択の運用最適化が必要。</li> <li>○ 新規ユーザーの学習に時間がかかる。</li> </ul>	

▼【生成AIコーディングアシスタント】事例06

ツール	Cursor	
提供者	Anysphere, Inc.	
URL	<a href="https://cursor.com/">https://cursor.com/</a>	
提供形態	ローカル（デスクトップアプリ）+SaaS（モデル推論/バックエンド）	
日本語対応	UI、サポート共に、日本語変更可能。一部機能や新機能については英語表示のまま。	
利用料	個人→Hobby：無料、Pro：\$20/月（\$20分の使用量を含む）、Ultra：\$200/月 チーム→Teams：\$40/ユーザー/月、Enterprise：詳細は見積 通貨はUSD。	
概要	<p>Cursorは、VS Code互換のAIネイティブIDEで、補完・チャット・エージェント・背景実行等を統合する。特筆点として、モデルのAPI価格に連動した“使用量込み”課金と自動モデル選択を提供し、重いタスクも扱える。業界では“IDE自体がAIエージェント”という新潮流の有力例として注目を集める。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ タブ補完・マルチライン編集・Bugbot等で実装を加速する。</li> <li>○ 背景エージェントが長時間タスクを継続実行する。</li> <li>○ 自動モデル選択（Auto）で高負荷タスクの成功率を高める。</li> <li>○ 学生割引やチーム向け追加容量管理を提供する。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 新規Webアプリの“1プロンプト雛形”生成。</li> <li>○ 既存バグ修正の再現・修正・テストの半自動化。</li> <li>○ レガシーコードの段階的な型付与やリファクタの伴走。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 使用量連動のためコスト見通し管理が必要。</li> <li>○ 企業のデータ保持方針に合わせたプライバシーモード設定が必要。</li> <li>○ より精度の高い結果を得たい場合は、英語で記述が必要。</li> </ul>	

▼【生成AIコーディングアシスタント】事例07

ツール	Tabnine	
提供者	Tabnine Ltd.	
URL	<a href="https://www.tabnine.com/">https://www.tabnine.com/</a>	
提供形態	SaaS／オンプレ／エアギャップ（企業向け）	
日本語対応	<p>UIは英語。 サポートは英語（EnterpriseでSLA/導入支援、Trust Centerあり）。</p>	
利用料	<p>開発：\$9/月（個人）、 エンタープライズ：\$39/ユーザー/月（企業） 通貨はUSD。</p>	
概要	<p>Tabnineは、“自社データを学習に用いない”ポリシーを前面に出したAIコード支援で、プライバシーとオンプレ対応に強みがある。特筆点として、ゼロデータ保持・エアギャップ運用・GDPR/SOC2等のセキュリティ重視の実装を提供し、厳格な企業要件で評価される。業界では、“プライバシー優先の実務導入”に適した選択肢として確立している。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ ローカル/サーバでのコード補完・生成を行う。</li> <li>○ コードレビューエージェントで品質・セキュリティの指摘を行う。</li> <li>○ 企業向けにオンプレ/エアギャップ構成を提供する。</li> <li>○ データ不使用方針とTrust Centerで透明性を担保する。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 金融/公共向け閉域ネットワークでのAI補完。</li> <li>○ OSS混入抑制を意識した安全な自動補完。</li> <li>○ 脆弱性検出方針に沿ったPRレビュー補助。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ エアギャップ/オンプレ構築の初期コスト。</li> <li>○ 英語サポート中心のため社内運用設計が必要。</li> <li>○ モデル/社内規程適合の監査対応に運用負荷。</li> </ul>	

▼【生成AIコーディングアシスタント】事例08

ツール	Snyk Code (DeepCode AI)	
提供者	Snyk Ltd.	
URL	<a href="https://snyk.io/product/snyk-code/">https://snyk.io/product/snyk-code/</a> (日本支社)	
提供形態	SaaS (IDE/CI連携)	
日本語対応	UIは英語。検出結果の解説やアドバイスの日本語提供あり。	
利用料	<p>無料、</p> <p>チーム：\$25/月（最低5人、最大10名の開発者まで）、</p> <p>エンタープライズ：詳細は見積 通貨はUSD。</p>	
概要	<p>Snyk Codeは、AI/ルールを組み合わせ、アプリコードのセキュリティ問題を検出・修正提案する。特筆点として、IDE/PRでのインクリメンタル解析や学習済み知見に基づく修正案提示が強みである。業界では、DevSecOpsの標準ツール群の一角として広範な導入実績がある。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ IDE/PRでのコードスキャンと即時フィードバックを行う。</li> <li>○ AI修正提案 (DeepCode AI) で自動修復を促進する。</li> <li>○ CI/CDに統合してブロッキング/ゲーティングを行う。</li> <li>○ ルール/ポリシー管理で組織標準に合わせる。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 新規脆弱性対応のIDE内自動修正。</li> <li>○ リポジトリ横断のリファクタ時の回帰防止。</li> <li>○ セキュアコーディング教育の実装支援。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 初期設計コストとセットアップの手間。</li> <li>○ 新規ユーザーの学習に時間がかかる。</li> <li>○ 既存SCA/DASTとの役割分担設計が必要。</li> </ul>	

▼【生成AIコーディングアシスタント】事例09

ツール	IBM watsonx Code Assistant	
提供者	International Business Machines Corporation	
URL	<a href="https://www.ibm.com/products/watsonx-code-assistant">https://www.ibm.com/products/watsonx-code-assistant</a>	
提供形態	SaaS中心（IBM Cloud）＋一部環境での拡張構成	
日本語対応	UIは英語中心。 サポートはIBMのサポート契約（地域体制あり）。	
利用料	詳細は見積（製品ファミリー/利用形態により大きく変動） 通貨はUSD。	
概要	<p>watsonx Code Assistantは、エンタープライズ向けのコード生成・変換・テスト支援に加え、メインフレームやAnsible自動化など運用現場の実務をカバーする。特筆点として、IBMの大規模顧客現場を想定したガバナンス/コンプライアンス前提の設計が強みである。業界では、“レガシー資産と現代化”の領域での適用が注目される。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ Java/COBOL等のコード変換・モダナイゼーションを行う。</li> <li>○ Ansible自動化プレイブックの生成・提案を行う。</li> <li>○ 企業標準に合わせたポリシー・監査支援を提供する。</li> <li>○ IBM Cloud/オンプレ連携で運用要件に合わせる。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ メインフレーム資産の段階的モダナイゼーション。</li> <li>○ 運用自動化（Ansible）での反復作業削減。</li> <li>○ Javaアプリのリファクタ/性能改善支援。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ エンタープライズ要件固有の要件定義とPoCが必要。</li> <li>○ 変換後の品質担保/検証に相応のプロセス設計が必要。</li> <li>○ 価格・構成が会社ごと見積となり比較が難しい。</li> </ul>	


▼【生成AIコーディングアシスタント】事例10

ツール	ChatGPT Canvas	 <b>ChatGPT</b>
提供者	OpenAI	
URL	<a href="https://openai.com/index/introducing-canvas/">https://openai.com/index/introducing-canvas/</a>	
提供形態	SaaS (ChatGPTのWebインターフェース機能)	
日本語対応	ChatGPT UIは日本語表示可。 サポートはヘルプセンター/ビジネス契約に依存 (いずれも英語中心)。	
利用料	無料版、 Plus：\$20/月、 Pro：\$200/月、 Business：\$25/ユーザー/月 (年額課金) または\$30/ユーザー/月 (月額課金)、 Enterprise：詳細は見積 通貨はUSD。	
概要	<p>Canvasは、チャット横に作業ペインを持ち、コードや文章を並行編集しながらモデルと共同作業できる。特筆点として、コード片のハイライト編集・その場適用・スニペット検証が容易で、“プロンプト→コード→即修正”の反復が速い。業界では、“エディタ的UIを備えた汎用LLM”として、軽量の試作・検証や説明に好適と評価される。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ キャンバス上でコード/文書を生成・編集・差分適用する。</li> <li>○ Webプレビュー/実行 (安全ガイドに従った範囲) を行う。</li> <li>○ ChatGPTプランのモデル/上限に準拠して動作する。</li> <li>○ リンクを共有し、同作業・管理を行う。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 小規模スクリプトの設計・実装・微修正の反復。</li> <li>○ README/記事/仕様の同時編集とサンプル更新。</li> <li>○ UIコンポーネントの雛形とスタイル微調整。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ ブラウザベースのため社内データ持込み方針の整備が必要。</li> <li>○ 実行/プレビューの安全ガイドに従う制約。</li> <li>○ 組織運用ではBusiness/Enterpriseの権限・ログ管理が推奨。</li> </ul>	

▼【生成AIコーディングアシスタント】事例11

ツール	Google Gemini Code Assist	
提供者	Google LLC (Google Cloud)	
URL	<a href="https://codeassist.google/">https://codeassist.google/</a>	
提供形態	SaaS (Cloud/IDE拡張)	
日本語対応	Cloud コンソール/Docsは日本語あり (英語併記)。サポートはGoogle Cloudのサポート契約。	
利用料	<p>無料：個人向け、</p> <p>標準：\$19/ユーザー/月 (年契約)、</p> <p>エンタープライズ：\$45/ユーザー/月 (年契約) 通貨はUSD。</p>	
概要	<p>Gemini Code Assistは、Cloud CodeやCloud Workstationsに統合され、コード生成・修正・レビュー・テスト支援を行う。特筆点として、GCPサービスへの深い統合と企業運用のアイデンティティ/セキュリティ管理に親和性が高い。業界では、GCP採用組織での“第一候補”になり易い。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ 生成・補完・テスト・レビューをIDEから行う。</li> <li>○ GCP資産 (IAM/Secret/Artifact) 連携の開発体験を提供する。</li> <li>○ リポジトリとドキュメントを横断するコンテキスト付与を行う。</li> <li>○ Cloud Workstations等と組み合わせ安全な開発環境を提供する。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ GKE/Cloud Run向けの雛形生成とCI設定。</li> <li>○ Terraform/Configの自動生成とレビュー。</li> <li>○ セキュリティ検出に基づく修正提案。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ Google Cloud組織設定/IAM設計の初期コスト。</li> <li>○ 利用モデルと請求/クォータの整合調整。</li> <li>○ 既存IDE/ツールチェーンとの重複機能の整理。</li> </ul>	

▼【生成AIコーディングアシスタント】事例12

ツール	Windsurf	
提供者	Cognition AI	
URL	<a href="https://windsurf.com/">https://windsurf.com/</a>	
提供形態	ローカル（デスクトップアプリ）+SaaS（クレジット制）	
日本語対応	UIは英語。 サポートは日本語対応。	
利用料	Free：25クレジット/月、 Pro：\$15/月（500クレジット）、 Teams：\$30/ユーザー/月、 Enterprise：\$60/ユーザー/月（ユーザー数/年契約等で変動） 通貨はUSD。	
概要	<p>Windsurfは、“エージェント的IDE”を標榜し、Cascade/Tab/Command等のモードで考える→書く→直すを一貫支援する。特筆点として、クレジット制で複数ベンダのフロントエンドモデルを横断利用でき、企業向けにゼロデータ保持・RBAC・ハイブリッド展開等を提供する。業界では、Cursorと並ぶ“AI IDE”の主要選択肢として急速に存在感を高めている。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ Cascade（エージェント）が大規模編集/新規実装を遂行する。</li> <li>○ Tab補完/Command/Browser/Terminal統合で作業を完結する。</li> <li>○ 複数モデルのAPI/クレジット課金を統合管理する。</li> <li>○ SOC2/ゼロデータ保持/SSO/RBAC等の企業要件を満たす。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 新規サービスの一気通貫な雛形作成とプレビュー。</li> <li>○ 大量ファイルを跨ぐ仕様変更の半自律実装。</li> <li>○ コード品質診断と修正の繰り返し実行。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ クレジット消費の見える化/運用ガイド整備が必要。</li> <li>○ コーディング支援に特化しており、用途が限定される。</li> </ul>	


▼【生成AIコーディングアシスタント】事例13

ツール	Cline	
提供者	Cline Bot Inc.	
URL	<a href="https://cline.bot/">https://cline.bot/</a>	
提供形態	ローカル（VS Code拡張）+BYO API（外部LLMを接続）	
日本語対応	UIは英語。 サポートは日本語対応。	
利用料	拡張自体は無料（Cline Teams等の有償サービスは別途、公式情報参照） 通貨はUSD。	
概要	<p>Clineは、VS Code内でリポジトリを横断して作業するマルチステップ・エージェントである。特筆点として、BYO API方式によって任意のモデル/プロバイダを使い分けられ、コスト/ポリシーを自社基準に合わせやすい。業界では、“拡張で擬似IDE化するエージェント”の代表格として広がっている。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ 指示に応じた調査→編集→実行→検証の反復を行う。</li> <li>○ タスク分解とログを可視化して追跡可能にする。</li> <li>○ 外部LLMのキー接続でコスト/モデルを選択可能にする。</li> <li>○ Teams等のコラボ/管理機能（別サービス）を併用できる。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ バグ再現から修正・PR作成までの半自動化。</li> <li>○ OSS課題のIssue→Patchの作業補助。</li> <li>○ ドキュメント整備とリンク切れ修正の一括対応。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ BYO APIの課金/レート管理が必要。</li> <li>○ 権限設定と誤操作対策（自動編集範囲の制御）が必要。</li> <li>○ 大規模プロジェクトではローカル環境のLLMで対応が難しい場合がある。</li> </ul>	

▼【生成AIコーディングアシスタント】事例14

ツール	Replit Agent	
提供者	Replit, Inc.	
URL	<a href="https://replit.com/">https://replit.com/</a>	
提供形態	SaaS（クラウドIDE内のエージェント機能）	
日本語対応	UIは英語。サポートは日本語対応。	
利用料	<p>Starter：無料、</p> <p>ReplitCore：\$25/月または\$20/年、</p> <p>Teams：\$40/ユーザー/月または\$35/ユーザー/年、</p> <p>Enterprise：詳細は見積 通貨はUSD。</p>	
概要	<p>Replit Agentは、クラウドIDE上でコード作成・実行・デプロイまで一気通貫で支援する。特筆点として、ブラウザだけで“作る→動かす→直す”を完結でき、教育/試作での摩擦が小さい。業界では、クラウドIDE×エージェントの実用例として普及が進む。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ チャット指示からプロジェクト生成・実行を行う。</li> <li>○ バグ修正・テスト・再実行の反復を自動化する。</li> <li>○ リアルタイムで共同編集できる。</li> <li>○ Replitのデプロイ機能を統合。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 学習用教材の演習プロジェクト自動生成。</li> <li>○ ハッカソン/PoCの短期プロトタイピング。</li> <li>○ Webスクレイピングや簡易APIの即時公開。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 学内/企業ネットワーク方針で外部SaaS使用の承認が必要。</li> <li>○ 従量制のコスト把握と監督が必要。</li> <li>○ 大規模プロジェクトでは、機能が不足する場合がある。</li> </ul>	

▼【生成AIコーディングアシスタント】事例15

ツール	aider	
提供者	Aider AI LLC	
URL	<a href="https://aider.chat/">https://aider.chat/</a>	
提供形態	ローカル (CLI) + BYO API (各LLM接続)	
日本語対応	<p>CLI/ドキュメントは英語。 サポートはGitHub/Discord等のコミュニティ中心。</p>	
利用料	<p>ツール本体は無料 (OSS)、 LLM利用は外部API等の従量課金。</p>	
概要	<p>aiderは、ローカルGitリポジトリを直接編集・コミットまで行う“端末常駐型ペアプロ”である。特筆点として、編集対象ファイル選択・差分管理・自動コミット等の開発実務に即した操作が充実する。業界では、“軽量・制御性が高い” AI編集フローとしてエンジニアに支持される。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ 指定ファイルへのパッチ生成・適用とコミットを行う。</li> <li>○ コメントによる編集指示 (watch mode) を受け付ける。</li> <li>○ 主要LLMの接続設定と切替を行う。</li> <li>○ 画像/URLを含む文脈入力に対応する。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 既存リポジトリの小規模修正反復とコミット駆動。</li> <li>○ 単体テスト追加と失敗修正の連続実行。</li> <li>○ ドキュメント整備とサンプルコード更新の同時進行。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ BYO APIの課金/秘匿管理が必要。</li> <li>○ CLI前提のためIDE慣れのチームでは教育が必要。</li> <li>○ 英語中心ドキュメントの内製ガイド化が必要。</li> </ul>	

▼【生成AIコーディングアシスタント】事例16

ツール	Hugging Face	
提供者	Hugging Face, Inc.	
URL	<a href="https://huggingface.co/">https://huggingface.co/</a>	
提供形態	SaaS（推論API/エンドポイント）+マネージド/セルフホスト選択	
日本語対応	UIは日本語対応あり。日本語を利用し易い環境が整っている。	
利用料	無料、 PRO（個人）：\$9/月、 チーム：\$20/ユーザー/月、 企業：\$50/ユーザー/月 通貨はUSD。	
概要	<p>Hugging Faceは、オープンモデル群の配布・推論基盤を提供し、開発者は多様なモデルを同一UI/課金で利用できる。特筆点として、プロバイダ横断で“中央集約の従量課金”を提供し、試験導入～本番の移行が容易である。業界では、“モデル選定/試行のデファクト・ハブ”として広く利用される。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ 多数モデルの推論APIを一元提供する。</li> <li>○ 自然言語処理や画像認識等データセットを公開・検索できる。</li> <li>○ Hugging Faceの機能や公開されているリソースの詳細を学べるドキュメントがある。</li> <li>○ 企業向けのSLA/セキュリティ要件に対応する。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 新旧モデル比較の迅速なベンチ実施。</li> <li>○ 本番系の特定エンドポイント移行・冗長化。</li> <li>○ 自社LLMと外部モデルのハイブリッド運用。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ モデルごとのレイテンシ・コスト最適化が必要。</li> <li>○ コンプライアンス要件に応じた接続/ログ設計。</li> <li>○ 使用しているライブラリの更新に注意する必要。</li> </ul>	

▼【生成AIコーディングアシスタント】事例17

ツール	Vercel v0	
提供者	Vercel Inc.	
URL	https://v0.app/	
提供形態	SaaS (ジェネレーティブUI/AIエージェント)	
日本語対応	UIは英語。日本語に完全対応。	
利用料	従量課金制 無料 (\$5相当クレジット/月付与)、 プレミアム: \$20/月 (\$20相当クレジット/月付与)、 チーム: \$30/ユーザー/月 (\$30相当クレジット/月付与) 通貨はUSD。	
概要	v0は、UI/アプリをプロンプト→生成→修正で構築できる“Generative UI”プラットフォームである。特筆点として、Vercelデプロイと密結合し、Next.js等のフロント実装を高速に試作・反復できる。業界では、“UI生成の先行例”かつ“Vercel堅牢基盤”でPoC～小規模本番の迅速な立ち上げに使われる。	
主要機能	<ul style="list-style-type: none"> <li>○ プロンプトからReact/Tailwind等のUIコードを生成する。</li> <li>○ 生成物をVercelに即デプロイする。</li> <li>○ クレジット制で高機能・長文生成を段階的に利用する。</li> <li>○ 企業向け利用条件/AIポリシーを提供する。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ LP/ダッシュボード等のUI雛形高速生成。</li> <li>○ デザイナー/PMと開発者のプロトタイピング共有。</li> <li>○ 小規模SaaSの初期版を素早く公開。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 英語UI/ポリシー読み解きとクレジット管理が必要。</li> <li>○ 複雑な業務UIは生成後の手修正前提。</li> <li>○ データ/秘密情報の扱いはVercel AI規約に準拠。</li> </ul>	

▼【生成AIコーディングアシスタント】事例18

ツール	Jitera	
提供者	株式会社Jitera	
URL	<a href="https://jitera.com/">https://jitera.com/</a>	
提供形態	SaaS+IDE拡張（企業向けカスタマイズ/支援）	
日本語対応	日本語サイト/資料あり。 サポートは日本語対応（企業導入サポート表記）。	
利用料	カスタム見積（料金ページに“カスタマイズ可能な料金プラン”の記載） 通貨はJPY/USD混在（問い合わせ）。	
概要	<p>Jiteraは、コードベースを理解し、仕様/設計ドキュメントの自動生成・更新を中心にして開発エージェント的にコーディング/テストも支援する。特筆点として、ドキュメント駆動の現場運用に合わせた“文脈管理”を重視し、日本企業の開発様式に適合し易い。業界では、“設計/運用ドキュメントの最新化”ニーズに合致した導入事例が増えている。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ 既存コードから設計/仕様ドキュメントを自動生成・更新する。</li> <li>○ 最適なデータベース構造やAPI仕様を自動生成する。</li> <li>○ 企業データ連携（GitHub/Drive/CRM等）で文脈を強化する。</li> <li>○ 逆解析/可視化機能でレガシー理解を加速する。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ レガシーシステムの逆解析と設計書再構築。</li> <li>○ 設計変更時のドキュメント自動追従。</li> <li>○ 新機能の仕様→エンドポイント雛形生成。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 個社要件に合わせた初期コンテキスト整備が必要。</li> <li>○ 機密データ取り扱い要件の精査と接続設計が必要。</li> <li>○ 現場ドキュメント様式との工数すり合わせが必要。</li> </ul>	

▼【生成AIコーディングアシスタント】事例19

ツール	AskCodi	
提供者	Assistiv.ai	
URL	<a href="https://www.askcodi.com/">https://www.askcodi.com/</a>	
提供形態	SaaS+IDE拡張	
日本語対応	UIは英語。 サポートは英語。	
利用料	無料、 フレキシブルプラン：\$25/月（必要に応じて拡張可能/いつでもキャンセル可能）、 企業：詳細は見積 通貨はUSD。	
概要	AskCodiは、コード生成・補完・説明・テスト作成をまとめて提供する軽量SaaSである。 特筆点として、拡張/オートコンプリート/“Talk to Project”等を束ねた一体UIで初心者～ 個人開発者に導入し易い。業界では、“低価格の総合型”として位置付けられる。	
主要機能	<ul style="list-style-type: none"> <li>○ 複数のプログラミング言語・コード生成/補完を行う。</li> <li>○ コード説明/ドキュメント化/テスト生成を行う。</li> <li>○ VS Code等のIDE拡張を提供する。</li> <li>○ リアルタイムデバッグとバグ検出を行う。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 個人学習プロジェクトの雛形作成。</li> <li>○ 小規模WebアプリのCRUD実装補助。</li> <li>○ 既存コードの解説とテスト追加。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ 英語UI/サポート中心。</li> <li>○ 個人/小規模向けで企業統制機能は限定的。</li> <li>○ クレジット制の把握/不足時の運用が必要。</li> </ul>	

▼【生成AIコーディングアシスタント】事例20

ツール	Devin	
提供者	Cognition Labs.	
URL	<a href="https://devin.ai/">https://devin.ai/</a>	
提供形態	SaaS (VS Code拡張機能)	
日本語対応	<p>UIは英語。 サポートは英語 (エンタープライズ導入は個別対応)。</p>	
利用料	<p>コア：\$20から始まる従量制支払い、 チーム：\$500/月、 企業：詳細は見積。</p>	
概要	<p>Devinは、チケット連携 (Jira/Linear/Slack) →計画→実装/テスト→PRを“並列クラウドエージェント”で処理できることを志向する。特筆点として、長時間・複合タスクの自律実行を目指す設計である。業界では、“本格的エージェント”の代表例として多数の検証・比較が行われている。</p>	
主要機能	<ul style="list-style-type: none"> <li>○ チケットから計画・作業・PR作成までの自動化を行う。</li> <li>○ 自己テスト/回帰検証を組み込む。</li> <li>○ 組織のワークフロー/知識に学習して適応する。</li> <li>○ セキュリティ/導入は個別エンタープライズ対応。</li> </ul>	
主な活用例	<ul style="list-style-type: none"> <li>○ 技術負債の継続的返済タスクの常時実行。</li> <li>○ 小規模機能の実装～テスト～PRの自動作成。</li> <li>○ 運用スクリプト修正や監視対応の自律処理。</li> </ul>	
国内可用性 導入ハードル 利用難点等	<ul style="list-style-type: none"> <li>○ API利用料が大きくなる傾向があり、ROI (費用対効果) の事前試算が必要。</li> <li>○ 複雑なUI構築には時間がかかる。</li> <li>○ 英語中心のため社内導入ガイド整備が必要。</li> </ul>	

### 2.4.3 調査考察

本調査で見た20事例から、生成AIコーディングアシスタントは“補完”から“工程横断 (SDLC) ×統制×エージェント”に重心が移っていることが判る。つまり、AIコーディングは、“便利機能”でなく、開発の進め方自体を規定するインフラになりつつある。

本事業との接続可能性の観点から好事例を幾つか取り上げると――

#### Copilot――普及＝ガバナンス (IP・情報管理・評価) の必須化

補完・生成・説明・テスト作成までを担い、GitHub連携や法人向けIP保護・管理が特筆点として整理される。“使える”だけでなく、安全活用 (正確性・著作権・情報管理) を到達目標に組み込む必要がある。

#### Amazon Q Developer/Gemini Code Assist――クラウド実務 (移行・IaC・運用) まで拡張

AWSサービス統合やエージェント的支援が整理され、GCP側もIDE統合+クラウド資産連携の方向性が示される。本事業の“要件定義→実装→運用自動化”像に直結する。

#### GitLab Duo/Snyk Code――AIが品質ゲート (DevSecOps) に入った

Issue/MR/スキャン文脈を横断して支援し、IDE/PRスキャン+AI修正提案+CI/CDゲーティングまで含む。“速く書く”ほど“速く検証・修正する”訓練が不可欠である。

#### Sourcegraph Cody――大規模コード読解+公開コード照合が導入の分水嶺

巨大リポジトリ横断の文脈把握と公開コード照合等のガードレールが整理される。生成より読解・影響分析・リスク低減が価値になる。

#### Cursor/Windsurf/Cline/aider――エージェント化で“運用設計”が技能になる

背景エージェント等を含むAIネイティブIDE、マルチモデル×クレジット制のエージェントIDE、BYO API+ログ可視化、Git差分・コミット駆動の端末ペアプロ――いずれも“どこまで委ね、どこで検証し、どう痕跡を残すか”が本体になる。

#### Tabnine/IBM watsonx――閉域・機密・レガシーという“現場制約”に寄り添う

Tabnineはゼロデータ保持やエアギャップ運用等、セキュリティ志向を前面に出し、IBM watsonxはJava/COBOL変換やAnsible生成等、モダナイゼーションと運用自動化を含む支援として整理される。

#### Replit Agent/v0/Jitera――試作で合意形成を前倒しする

ブラウザ完結の反復、UI生成→即デプロイ、設計/仕様ドキュメント自動生成・更新は、実装速度以上に“仕様確定を早める”効果を示す。

#### Devin――究極系 (チケット→PR) であるが、教育では責任分界と検証設計が核心

チケット連携→計画→実装/テスト→PRを並列クラウドエージェントで処理する志向が整理される。

地方企業では、汎用チャット利用が中心であるが、コーディング補完・リファクタ需要自体は存在する(調査①)。そのため、本事業で企図する教育価値は、特定ツールの操作でなく、工程横断でAIを位置付け、制約を充し、品質を保証する“選定・統制・検証”の型を、地方の小規模チームでも再現可能にする点にある。その実装として、本調査の狙い通り、これを演習環境要件と演習設計に落とし込むべきである。具体策は次の3点である。

- クラウドIDEに補完/チャット/エージェントの複数形態と、差分・実行ログ・テスト結果の保存を必須化する(評価・振り返り接続)。
- “速く書く”(Copilot/JetBrains) × “安全に直す”(Snyk/GitLab) × “読む/探す”(Cody) × “合意を取る”(v0/Replit/Jitera) を4点セットの演習にする。
- 自律エージェント(Cursor/Windsurf/Devin)は“タスク分解・責任分界・検証設計”を学ばせる教材として扱う。

これにより、“要件定義→実装→運用自動化”を短期間で自走できる人材像を、具体的ツール運用技能として教育・評価に接続できる。

## 2.5 調査④ AI活用エンジニアリング教育事例調査

調査④はAI活用エンジニアリング教育事例調査である。

これは、国内外のAI活用エンジニアリング教育の事例や指導法等に関する情報を収集・整理することで、本教育プログラムのカリキュラム・シラバス構成要件を明かにし、開発の基礎資料にするためのものである。

その調査仕様や調査結果を以下に報告する。

### 2.5.1 調査仕様

本調査の仕様（目的・対象・手法・内容・反映方針等）は次の通りである

名称	AI活用エンジニアリング教育事例調査
目的	国内外のAI活用エンジニアリング教育の事例や指導法等に関する情報を収集・整理することで、本教育プログラムのカリキュラム・シラバス構成要件を明かにし、開発の基礎資料にする。
対象	【A】 各種公開資料（出版物・ウェブサイト等） 【B】 生成AI応用教育担当教員・生成AI活用開発チーム責任者
手法	【A】 デスクリサーチ 【B】 ヒヤリング
項目	ツール名・提供者・URL・提供形態・日本語対応・利用料・概要・主要機能・主な活用例・国内可用性（導入ハードル・利用難点等）

### 2.5.2 調査結果

本調査では、AI活用エンジニアリング教育の国内外の事例や指導法等について、【A】 デスクリサーチと【B】 ヒヤリングを実施した。

その結果として、まづ【A】 20事例を確認した。その基本情報や仕様・特徴等を次頁以降に整理する。そして、【B】 AI活用エンジニアリングに関する有識者（2人）の見解を得た。それを【A】の後に掲載する。

## 【A】 デスクリサーチ

### ▼【AI活用エンジニアリング教育】事例01

講座名	IoTプログラミング with AI
提供者	AI・高度プログラミングコンソーシアム (AIC) (慶應義塾大学グローバルリサーチインスティテュート (KGRI) 内に設置されている研究センター)
講数	6回 (1日3コマ×2日間)
概要	本講座では、生成AI (GitHub Copilot) を実装支援に用い、IoTを題材としてハンズオンで小型システムを製作する。Raspberry Pi Pico (非無線) またはPico W (無線LAN搭載) を用い、モーターおよび各種センサーを組み合わせた動作系を構築する。
内容	<p>01 ガイダンス・環境構築・Copilotの使い方 講座の概要とスケジュール、進め方等を説明した後、組み込みプログラミングを概観する。</p> <p>02 ハードウェア・ソフトウェア Raspberry Pi Picoが電流を制御するため機能であるGPIO、PWM、電圧を計測するため機能であるADCの概要と使い方を解説し、これらを用いてLEDとモーターを制御し、測距センサーを用いるためのプログラムおよび配線の方法を解説する。</p> <p>03 アイデア決定・実装 アイデアシートを配布し、受講生が各々実装してみたい内容を考え出す。アイデアが決まったら、TAがチェックした後、各自実装に入る。</p> <p>04 実装 (続き) 実装を行う。</p> <p>05 実装 (続き) 実装を行う。</p> <p>06 発表 参加者に、スライド1～2枚程度の資料を作成してもらい、約3分の発表を行う。</p>
評価	全員の発表が終わり次第、講評および今後の学習の進め方等について解説する。
所見	<ul style="list-style-type: none"><li>○ 企業と連携し、AI技術の実社会応用をテーマとしたプロジェクト型学習を実施する。</li><li>○ 産学連携によるプロジェクトを通じて、塾生がより実践的なAI開発に取り組む機会を増やし、技術の社会実装に貢献できる人材の育成を目指す。</li></ul>

## ▼【AI活用エンジニアリング教育】事例02

講座名	AI活用UX/UIデザインプログラミング演習
提供者	関西学院大学・AI活用人材育成プログラム
講数	14回
概要	本講座では、UX/UIデザインおよびデザイン思考の基礎を踏まえ、生成AIを活用したWebアプリケーションの画面設計と実装（HTML/CSS/JavaScript等）を段階的に演習する。
内容	<p>01 ユーザーエクスペリエンス ユーザーエクスペリエンス（UX：User Experience）の概念を整理し、画面設計の前提となる観点を確認する。</p> <p>02 デザイン思考の基礎 デザイン思考の枠組みと基本プロセスを扱い、課題設定と解決の進め方を整理する。</p> <p>03 Design Thinkingを用いたUX/UI デザイン思考を用いたUX/UI設計の進め方を扱い、手順と観点を確認する。</p> <p>04 コーディング HTML/CSSによる画面実装の基礎を扱い、最小構成でのページ構築を行う。</p> <p>05 コーディング2 HTML/CSS実装を発展させ、レイアウトおよびコンポーネント表現を整理する。</p> <p>06 コーディング3 HTML/CSS実装の発展として、レスポンシブ等の観点を含めた画面調整を行う。</p> <p>07 JavaScript JavaScriptの基本概念を整理し、画面の振る舞いを制御する基礎を扱う。</p> <p>08 JavaScript 2 イベント処理等を扱い、動的な画面更新の基本を確認する。</p> <p>09 JavaScript 3 状態管理等の観点を含め、画面ロジックの実装を段階的に行う。</p> <p>10 総合演習（テーマ設定） テーマを設定し、利用者像および利用文脈を定義する。</p> <p>11 総合演習（ワイヤーフレーム） テーマに即したワイヤーフレームを作成し、画面構成を整理する。</p> <p>12 総合演習（画面設計） テーマに即したUX/UIをデザインし、画面設計を確定する。</p> <p>13 総合演習2（画面開発） ワイヤーフレームに基づき、Webサイトを実装する。</p> <p>14 総合演習3（画面機能の拡張） HTML/CSS実装を基に、JavaScriptを用いてロジックを実装する。</p>
評価	評価方法は非公開または記載なし
所見	<ul style="list-style-type: none"> <li>○ 予備知識がなくても段階的に学べる構成で、理系文系を問わず初学者が取り組める設計である。</li> <li>○ 既存内容の流用ではなく、全体を通して実装と設計を往還する構成である。</li> <li>○ 日本IBMとの共同プロジェクトにより、ビジネス現場で即戦力となることを念頭に授業内容を設計している。</li> </ul>

### ▼【AI活用エンジニアリング教育】事例03

講座名	AIを活用したWebアプリケーション開発演習（入門／実践）
提供者	GMOインターネット株式会社 エンジニア（九州大学で講義）
講 数	夏学期・冬学期ともに週1回／全8回（合計16回）
概 要	本講座では、夏学期にノーコードを用いた生成AIアプリ開発を通じて基礎を固め、冬学期に自由開発形式で実装の実践へ接続する。現場エンジニアによる講義・演習を通じ、要件、品質、運用の観点を含む開発の型を学ぶ。
内 容	<ul style="list-style-type: none"> <li>01 AI概論 AI技術の歴史と基本概念を扱う。</li> <li>02 LLM概論 大規模言語モデル（LLM）の基本概念と機能を扱う。</li> <li>03 ITエンジニアの業務とAIとの関わり 1 ITエンジニア業務の概要を整理し、インフラ領域の事例を用いてAIとの関わりを扱う。</li> <li>04 ITエンジニアの業務とAIとの関わり 2 セキュリティおよびカスタマー領域の事例を用いて、AIの活用局面を扱う。</li> <li>05 ITエンジニアの業務とAIとの関わり 3 Webアプリ開発におけるCopilot活用およびプロトタイプ駆動の観点を扱う。</li> <li>06 ITエンジニアの業務とAIとの関わり 4 DevSecOpsおよびSRE等、品質と運用を含めた実践手法を扱う。</li> <li>07 ノーコードアプリ開発 Dify等を用い、ノーコードによるアプリ開発を演習する。</li> <li>08 発表 成果物を発表し、講評を行う。</li> <li>09 自由開発の準備 テーマ設定と設計方針を整理し、開発計画を立てる。</li> <li>10 自由開発 設計に基づき実装を進める。</li> <li>11 自由開発（続き） 実装を継続し、品質および運用観点を整理する。</li> <li>12 自由開発（続き） 実装を継続し、改善点を反映する。</li> <li>13～14 自由開発（続き） 成果物の完成度を高め、発表準備を行う。</li> <li>15 個人開発 個人開発を継続する。</li> <li>16 発表 各自の成果物を発表する。</li> </ul>
評 価	各回で小テストを実施し、出席は2／3以上とする。
所 見	<ul style="list-style-type: none"> <li>○ ハンズオンおよび自由開発形式により、基礎から実践までを接続する構成である。</li> <li>○ 現場で活躍するエンジニアが授業を担当し、実務経験に基づく指導を行う。</li> <li>○ 産学連携を通じた実践的な教育プログラムを実施している。</li> </ul>

▼【AI活用エンジニアリング教育】事例04

講座名	先端メディアゼミナールⅡ [人工知能を利用したサービス開発]
提供者	東京工科大学・メディア学部
講数	14回
概要	本講座では、深層学習を基盤とする人工知能技術を概観し、実装を通じてメディア領域のサービス開発に接続する。
内容	<p>01 全体概要・人工知能とは（機械学習・深層学習） 全体概要および人工知能の基本概念に関する講義および演習を行う。</p> <p>02 ニューラルネットワーク（仕組みと構造） ニューラルネットワークの仕組みと構造に関する講義および演習を行う。</p> <p>03 ニューラルネットワーク（バックプロパゲーション） バックプロパゲーションに関する講義および演習を行う。</p> <p>04 深層学習（畳み込みニューラルネットワーク） 畳み込みニューラルネットワークに関する講義および演習を行う。</p> <p>05～06 深層学習（TensorFlow） 深層学習（TensorFlow）に関する講義および演習を行う。</p> <p>07～09 クラウドサービス活用方法（IBM Cloud、Watson） クラウドサービス活用方法に関する講義および演習を行う。</p> <p>10～13 サービス開発 サービス開発に関する講義および演習を行う。</p> <p>14 発表とリリース 成果物を発表し、講評およびリリースの観点を整理する。</p>
評価	前半の基礎内容における演習課題は都度フィードバックを行う。後半の開発課題では個別にフィードバックを行い、総合してサービスの実装状況（50点）と最終レポート（50点）で評価し、合計60点以上を合格とする。
所見	<ul style="list-style-type: none"> <li>○ 学生は豊富な専門科目から興味に応じた科目を選択できる。</li> <li>○ 国籍、性別、年齢、障害の有無等に関わらず学ぶ環境が整っている。</li> <li>○ 1年次から全学生に担当教員がつき、「フレッシューズゼミ」での指導等を担当するアドバイザー制度がある。</li> </ul>

## ▼【AI活用エンジニアリング教育】事例05

講座名	人工知能システム開発実習
提供者	大阪国際工科専門職大学・工科学部・情報工学科
講 数	15回（通年、1回あたり2～4コマ程度）
概 要	本講座では、各種ライブラリおよびフレームワークを用いたシステム構築を通じ、マルチメディア情報処理の理解を深める。AIフレームワークはTensorFlow/PyTorch等を中心に扱い、必要に応じてCaffe等に触れる。Chainerは保守中心であることを踏まえ、比較対象として取り扱う。あわせて、業務活用を想定した人工知能システムの考案から開発・評価・改善までを行い、PDCAサイクルを体験する。
内 容	<p>01 ガイダンス： 科目の趣旨、到達目標、進め方、評価の枠組み等を確認する。</p> <p>02 課題発見： 企業の業務を想定し、課題設定と検討観点を整理する。</p> <p>03 課題発見（続き）： 課題設定を精緻化し、実現可能性の観点を整理する。</p> <p>04 課題発見（続き）： 課題設定を確定し、要件化の方針を整理する。</p> <p>05 課題発見（続き）： 要件化の方針を整理し、実装対象を確定する。</p> <p>06 計画策定： 実装に向けた前提整理と計画策定の枠組みを扱う。</p> <p>07 計画策定（続き）： タスク分解、役割分担、スケジュールの観点を整理する。</p> <p>08 計画策定（続き）： 進捗管理の観点を整理し、計画を確定する。</p> <p>09 計画策定（続き）： 計画を確定し、実装準備を行う。</p> <p>10 中間発表： 進捗と設計方針を共有し、フィードバックを得る。</p> <p>11 システム構築（1）： 実装を開始し、基盤構築および初期検証を行う。</p> <p>12 システム構築（2）： 実装を進め、機能実装と検証を行う。</p> <p>13 システム構築（3）： 実装を進め、性能および品質観点を整理する。</p> <p>14～15 システム構築（4）（5）： 実装を進め、改善点を反映する。</p> <p>16 評価（1）： 評価観点と評価手順を整理し、評価を実施する。</p> <p>17 評価（2）： 評価を継続し、結果の整理を行う。</p> <p>18 評価（3）： 評価結果を分析し、改善方針を整理する。</p> <p>19 評価（4）： 改善方針に基づき、修正と再評価を行う。</p> <p>20 計画策定（1）： 人工知能システムを実現するために、技術要素（OpenCVライブラリ、ディープラーニング、畳み込みニューラルネットワーク、Caffe等）、タスク洗い出し、優先順位決定、役割分担決定とスケジュール（WBS等）の作成を行う。</p> <p>21 計画策定（2）： 計画の具体化を行い、実装の段取りを整理する。</p> <p>22 開発（1）： 実装を進め、機能単位で検証を行う。</p> <p>23 開発（2）： 実装を進め、品質と性能の観点を整理する。</p> <p>24～43 開発（3）～（22）： 実装を進め、改善点を反映する。</p> <p>44 最終発表： 成果物の発表を行い、総括する。</p> <p>45 最終レポート： 成果および学修内容を整理し、最終レポートを作成する。</p>
評 価	科目認定条件は、出席率80%以上、提出物80%以上の提出、成績評価60点以上とする。評価点は、レポート平均点30%、中間発表評価30%、プレゼン40%とする。
所 見	○ システム構築と評価・改善を往還し、開発プロセスの一連を演習する構成である。

▼【AI活用エンジニアリング教育】事例06

講座名	プログラミング発展B
提供者	武蔵野大学・副専攻・「AI活用エキスパートコース」
講数	14回
概要	本講座では、HTMLおよびJavaScriptを中心に、生成AIの補助を活用しつつ、Webアプリケーションを設計・実装するための基礎を演習する。
内容	<ul style="list-style-type: none"> <li>01 ガイダンス 授業概要、到達目標、進め方を確認する。</li> <li>02 HTMLの概要 HTMLの役割と基本要素を整理する。</li> <li>03 HTMLを用いたページ作成 HTMLによりページ構造を記述する演習を行う。</li> <li>04 CSSの概要 CSSの役割と基本表現を整理する。</li> <li>05 CSSを用いたページ作成 CSSにより装飾およびレイアウトを行う演習を行う。</li> <li>06 JavaScriptの概要 JavaScriptの役割と基本構文を整理する。</li> <li>07 JavaScriptの基本 変数、条件分岐、繰り返し等の基本を扱う。</li> <li>08 JavaScriptによるDOM操作 DOM操作を通じた画面制御の基礎を扱う。</li> <li>09 イベント処理 イベント処理の基本を扱う。</li> <li>10 総合演習（要件整理） 課題の要件を整理し、設計方針を立てる。</li> <li>11 総合演習（実装） 設計に基づき実装を行う。</li> <li>12 総合演習（実装・改善） 実装を継続し、改善点を反映する。</li> <li>13 総合演習（仕上げ） 成果物を整え、提出準備を行う。</li> <li>14 発表・講評 成果物を発表し、講評を行う。</li> </ul>
評価	評価方法は非公開または記載なし
所見	<ul style="list-style-type: none"> <li>○ AIを活用し、学修者が自ら課題を設定し、Web用アプリケーションを作成する演習を中心とする。</li> <li>○ 演習を通じて、HTMLおよびJavaScriptを利用したツールやサービスの作成方法を扱う。</li> </ul>

▼【AI活用エンジニアリング教育】事例07

講座名	応用プログラミング演習
提供者	埼玉工業大学・工学部・情報システム学科 AI専攻
講数	14回
概要	本講座では、Pythonの基礎および機械学習の実装を踏まえ、深層学習フレームワークを用いたモデル構築と実装の要点を扱う。プログラミング環境としてGoogle Colabを主に用いる。
内容	<p>01 ガイダンス 授業の狙い、到達目標、進め方を確認する。</p> <p>02 線形回帰とその実装 線形回帰の概念と実装手順を扱う。</p> <p>03 ロジスティック回帰とその実装 ロジスティック回帰の概念と実装手順を扱う。</p> <p>04 ニューラルネットワークとその実装 ニューラルネットワークの基本と実装の枠組みを扱う。</p> <p>05 畳み込みニューラルネットワークとその実装 畳み込みニューラルネットワークの基本と実装を扱う。</p> <p>06 計算グラフと逆伝播 計算グラフの考え方と逆伝播の概念を扱う。</p> <p>07 誤差逆伝播法 誤差逆伝播法の概念と実装の観点を扱う。</p> <p>08 バックプロパゲーションによる学習とその実装①： 計算グラフと逆伝播 計算グラフと逆伝播を用いた学習手順を扱う。</p> <p>09 バックプロパゲーションによる学習とその実装②： 誤差逆伝播法 誤差逆伝播法を用いた学習手順を扱う。</p> <p>10 深層学習フレームワークの基礎 深層学習フレームワークの基本機能と使い方を扱う。</p> <p>11 深層学習フレームワークの応用 実装の効率化と再利用の観点を扱う。</p> <p>12 モデル設計と実装 モデル設計の観点と実装の進め方を扱う。</p> <p>13 総合演習 テーマに基づき、実装と検証を行う。</p> <p>14 発表・講評 成果物を発表し、講評を行う。</p>
評価	評価方法は非公開または記載なし
所見	<p>○ Pythonの入門と機械学習の実装を踏まえ、深層学習を中心にフレームワークを用いた実装手法を扱う。</p> <p>○ プログラミング環境としてGoogle Colabを主に用いる。</p>

▼【AI活用エンジニアリング教育】事例08

講座名	先端データサイエンス・AI発展第四
提供者	東京科学大学データサイエンス・AI教育
講数	7回
概要	本講座では、デジタル・AI技術のビジネス活用を題材に、基礎概念の理解とビジネスプラン策定をPBL形式で行う。
内容	<p>01 講義： デジタル・AIによるビジネス変革 デジタル・AI技術をビジネスに適用する観点を整理する。</p> <p>02 講義： 経営の基礎 経営に係る基礎概念を整理する。</p> <p>03 講義： ビジネスプランの作成方法 ビジネスプラン策定の枠組みを扱う。</p> <p>04 PBL： テーマ設定 グループワークによりテーマ設定を行う。</p> <p>05 PBL： ビジネスプラン作成 ビジネスプランの作成を進める。</p> <p>06 PBL： 発表準備 発表資料を整え、説明の構成を整理する。</p> <p>07 PBL： 発表 ビジネスプランを発表し、講評を行う。</p>
評価	評価方法は非公開または記載なし
所見	<ul style="list-style-type: none"> <li>○ DS&amp;AI全学教育の一環として位置づけられている。</li> <li>○ PBL形式のグループワークにより、ビジネスプラン作成を行う。</li> </ul>

▼【AI活用エンジニアリング教育】事例09

講座名	プログラミング演習1 (C/C++)
提供者	法政大学・情報科学部・メディア科学
講数	14回
概要	本講座では、C/C++による一定規模のプログラム開発を通じ、実行環境・開発環境を含む実践的なプログラミング能力を養う。
内容	<ul style="list-style-type: none"> <li>01 ガイダンス 授業概要、到達目標、進め方を確認する。</li> <li>02 開発環境の確認 実行環境および開発環境の構成を確認する。</li> <li>03 基礎演習 構文および基本データ構造を扱う。</li> <li>04 関数とモジュール化 関数設計とモジュール化の観点を扱う。</li> <li>05 ポインタとメモリ メモリ管理とポインタの観点を扱う。</li> <li>06 構造体とデータ設計 構造体を用いたデータ設計を扱う。</li> <li>07 ファイル入出力 ファイル入出力の基本を扱う。</li> <li>08 総合演習 (設計) 課題に対する設計方針を整理する。</li> <li>09 総合演習 (実装) 設計に基づき実装を行う。</li> <li>10 総合演習 (実装・検証) 実装を継続し、検証を行う。</li> <li>11 総合演習 (改善) 改善点を反映する。</li> <li>12 総合演習 (仕上げ) 成果物を整える。</li> <li>13 発表準備 説明資料を整える。</li> <li>14 発表・講評 成果物を発表し、講評を行う。</li> </ul>
評価	評価方法は非公開または記載なし
所見	<ul style="list-style-type: none"> <li>○ 一定規模のプログラム開発を通じ、実践的プログラミングを学ぶ構成である。</li> <li>○ 文法のみならず、実行環境・開発環境を含めたプログラミング環境に慣れることを狙いとする。</li> </ul>

▼【AI活用エンジニアリング教育】事例10

講座名	機械学習Ⅱ
提供者	日本工業大学・先進工学部・データサイエンス学科
講数	14回
概要	本講座では、強化学習および深層強化学習を対象に、数式とPython実装の双方から理解を深める。問題の定式化、行動価値関数の導出、解決手法の選択までを演習を主体として扱う。
内容	<ul style="list-style-type: none"> <li>01 強化学習の概要 強化学習の基本概念を扱う。</li> <li>02 マルコフ決定過程 状態遷移モデルの観点を整理する。</li> <li>03 方策と価値 方策および価値の概念を整理する。</li> <li>04 行動価値関数 行動価値関数の導出と扱い方を整理する。</li> <li>05 価値反復と方策反復 代表的アルゴリズムを扱う。</li> <li>06 関数近似 関数近似の観点を扱う。</li> <li>07 深層強化学習の概要 深層強化学習の枠組みを扱う。</li> <li>08 DQN DQNの考え方と実装観点を扱う。</li> <li>09 方策勾配法 方策勾配法の考え方を扱う。</li> <li>10 Actor-Critic Actor-Criticの枠組みを扱う。</li> <li>11 応用課題（定式化） 課題の定式化を行う。</li> <li>12 応用課題（実装） 実装と検証を行う。</li> <li>13 応用課題（改善） 改善点を反映する。</li> <li>14 発表・講評 成果物を発表し、講評を行う。</li> </ul>
評価	評価方法は非公開または記載なし
所見	<ul style="list-style-type: none"> <li>○ 概念・理論は数式表現を中心に演習を主体として学修する。</li> <li>○ 実装はPythonによる実装方法を演習を主体として学修する。</li> <li>○ 強化学習および深層強化学習を用いた問題解決の一連を扱う。</li> </ul>

▼【AI活用エンジニアリング教育】事例11

講座名	システム総合演習 I
提供者	会津大学大学院
講 数	14回
概 要	本講座では、Raspberry Piを用いたシステム構築を通じ、センサー入出力、ネットワーク連携、深層学習等を含む一連の実装を演習する。
内 容	<p>01 ガイダンス 授業概要、到達目標、進め方を確認する。</p> <p>02 Raspberry Piの概要 Raspberry Piの基本構成と利用方法を扱う。</p> <p>03 Linux基礎 Linux環境での基本操作を扱う。</p> <p>04 センサー入出力 センサーおよびアクチュエータ制御の基礎を扱う。</p> <p>05 ネットワーク連携 ネットワークを介したデータ連携の基礎を扱う。</p> <p>06 データ処理 取得データの処理と可視化の観点を扱う。</p> <p>07 深層学習の概要 深層学習の基本概念を整理する。</p> <p>08 モデル利用 学習済みモデルの利用方法を扱う。</p> <p>09 応用演習（設計） 応用テーマを設定し、設計方針を整理する。</p> <p>10 応用演習（実装） 実装と検証を行う。</p> <p>11 応用演習（改善） 改善点を反映する。</p> <p>12 発表準備 説明資料を整える。</p> <p>13 発表 成果物を発表する。</p> <p>14 総括 全体の振り返りと講評を行う。</p>
評 価	評価方法は非公開または記載なし
所 見	<p>○ Raspberry Piを用いて、プログラム開発、センサーやアクチュエータとの連携、深層学習等を含む実装を扱う。</p> <p>○ 本演習は、Raspberry Piを用いたコンピュータシステムの構築を目的とする。</p>

▼【AI活用エンジニアリング教育】事例12

講座名	業界研究ⅢA
提供者	TECH.C.東京デザインテクノロジーセンター専門学校
講数	15回
概要	本講座では、クラウド技術の基礎を踏まえ、AWS (Amazon Web Services) 上でのWebアプリケーション開発を通じて、Web技術およびネットワーク技術の基礎を修得する。
内容	<p>01 クラウドとAWSの概要 クラウドおよびAWSに関する基本概念と用語を整理する。</p> <p>02 AWSの基本操作 (コンソール、CLI、SDK) AWSへのログインおよび基本操作を扱う。</p> <p>03 仮想サーバー (EC2) の作成と管理 EC2の概要と基本操作を扱う。</p> <p>04 ネットワーク基礎 (VPC、サブネット、セキュリティグループ) Webサービスを作るうえで必要なネットワークの基礎を扱う。</p> <p>05 ストレージ (S3) を活用したWebサイトホスティング S3を用いたWebサイトホスティングの基礎を扱う。</p> <p>06 データベース (RDS) の基礎 RDSの概要と基本操作を扱う。</p> <p>07 負荷分散 (ELB) とオートスケーリング 可用性確保の観点として負荷分散とスケーリングを扱う。</p> <p>08 サーバーレス (Lambda) サーバーレスの基本概念とLambdaの利用方法を扱う。</p> <p>09 API (API Gateway) APIの基本概念とAPI Gatewayの利用方法を扱う。</p> <p>10 認証 (Cognito) 認証の基本概念とCognitoの利用方法を扱う。</p> <p>11 セキュリティ基礎 クラウド利用におけるセキュリティの基礎を扱う。</p> <p>12～15 AWSを使ったWebアプリケーションの作製 1～4 AWS上で簡単なWebサービスを作成し、理解を深める。</p>
評価	最終提出物 (80%)、出席 (10%)、授業中の提出物 (10%)。
所見	<ul style="list-style-type: none"> <li>○ マイクロソフトの教育向けカリキュラムを学ぶことができる。</li> <li>○ 世界共通のIT資格「CompTIA」の教育プログラムも学ぶことができる。</li> <li>○ 夢や目標に合わせて、専攻以外の授業も履修することができる。</li> </ul>

▼【AI活用エンジニアリング教育】事例13

講座名	システム開発演習 I
提供者	大阪情報コンピュータ専門学校・IT系・ITスペシャリスト専攻
講 数	15回
概 要	本講座では、実務を想定したチーム開発により、要件定義から設計・構築までのシステム開発プロセスを疑似体験する。
内 容	<p>01 ガイダンス 授業概要、進め方、評価の観点を確認する。</p> <p>02 課題説明 課題の背景と要件を確認する。</p> <p>03 要件整理 クライアント視点と開発者視点の双方から要件を整理する。</p> <p>04 基本設計 基本設計の観点を整理し、設計方針を確定する。</p> <p>05 詳細設計 詳細設計を行い、実装方針を整理する。</p> <p>06 実装 実装を行い、動作確認を行う。</p> <p>07 実装（続き） 実装を継続し、機能を拡張する。</p> <p>08 テスト テスト観点を整理し、検証を行う。</p> <p>09 改善 改善点を反映する。</p> <p>10 ドキュメント整備 成果物に係るドキュメントを整備する。</p> <p>11 発表準備 発表資料を整える。</p> <p>12 発表 成果物を発表し、講評を行う。</p> <p>13 振り返り 学修内容を整理し、振り返りを行う。</p> <p>14 予備 補講又は調整を行う。</p> <p>15 総括 全体の総括を行う。</p>
評 価	評価方法は非公開または記載なし
所 見	<p>○ 実際のシステム開発現場および実務を想定し、設計から構築までのプロセスをチームで疑似体験する。</p> <p>○ 発注者（クライアント）と開発者（エンジニア）の双方の視点に立ち、開発の全過程を通じて学ぶ構成である。</p>

▼【AI活用エンジニアリング教育】事例14

講座名	モバイルアプリ開発2
提供者	日本工学院専門学校・ITカレッジ・ITスペシャリスト科
講数	15回
概要	本講座では、スマートデバイス導入に必要な基礎用語および構成要素を整理し、iOSアプリケーション開発を中心に演習する。
内容	<p>01 ガイダンス 授業概要、到達目標、進め方を確認する。</p> <p>02 モバイルシステムの基礎 用語と構成要素を整理する。</p> <p>03 iOS開発環境 開発環境の構成と基本操作を扱う。</p> <p>04 UI設計 画面設計の基礎を扱う。</p> <p>05 データ処理 データ入出力の基礎を扱う。</p> <p>06 ネットワーク通信 通信処理の基礎を扱う。</p> <p>07 機能実装 機能実装を進める。</p> <p>08 機能実装（続き） 機能実装を継続する。</p> <p>09 テスト テスト観点を整理し検証を行う。</p> <p>10 改善 改善点を反映する。</p> <p>11 発表準備 説明資料を整える。</p> <p>12 発表 成果物を発表する。</p> <p>13 総括 振り返りと講評を行う。</p> <p>14 予備 補講又は調整を行う。</p> <p>15 予備 補講又は調整を行う。</p>
評価	評価方法は非公開または記載なし
所見	<ul style="list-style-type: none"> <li>○ 文部科学大臣が認定する制度、「職業実践専門課程」が導入されている。</li> <li>○ PBL (Project Based Learning) を取り入れている。</li> <li>○ 学習内容をステップ式に明示し、就労に必要なスキルを卒業までに段階ごとに着実に身に付ける「教育設計図」を導入している。</li> </ul>

▼【AI活用エンジニアリング教育】事例15

講座名	AIクラウドプログラミング
提供者	東京町田情報ITクリエイター専門学校・情報IT+東京経営大学卒業コース
講数	30回
概要	本講座では、AWSのAIサービスを活用したプログラム実装を通じ、複数サービスの組合せによる開発手順を習得する。
内容	<p>01 クラウドの基礎 クラウド利用の基本概念を整理する。</p> <p>02 AWSの基本 AWSの基本構成と利用方法を整理する。</p> <p>03 AWS AIサービス概観 AWSのAIサービス群の位置づけを整理する。</p> <p>04 画像認識 画像認識系サービスの利用方法を扱う。</p> <p>05 音声認識 音声認識系サービスの利用方法を扱う。</p> <p>06 自然言語処理 自然言語処理系サービスの利用方法を扱う。</p> <p>07 チャットボット チャットボット構築の観点を扱う。</p> <p>08 データ分析 データ分析の観点を扱う。</p> <p>09 統合演習（設計） テーマ設定と設計方針を整理する。</p> <p>10 統合演習（実装） 実装と検証を行う。</p> <p>11～28 統合演習（改善） 改善点を反映する。</p> <p>29 発表準備 説明資料を整える。</p> <p>30 発表・講評 成果物を発表し、講評を行う。</p>
評価	評価方法は非公開または記載なし
所見	○ AWSのAIサービスを利用した実装方法を扱い、複数サービスを用いた開発を通じて理解を深める構成である。

▼【AI活用エンジニアリング教育】事例16

講座名	AIアプリコース
提供者	SAMURAI ENGINEER (侍エンジニア)
講数	最大1年5箇月
概要	本コースでは、環境構築、データ収集、モデル構築からアプリケーション実装までを一気通貫で扱う。発注者を想定した要求定義を起点に、要件定義書・設計書の作成を含む開発プロセスを演習する。
内容	<p>01 基礎学習 (1～2箇月)： Google Colaboratory/Jupyter Notebook/Python 手を動かしながら体系的に基本的なコーディング技術を習得し、章末テストにより定着を確認する。</p> <p>02 応用学習 (3～4箇月)： データ取得/pandas/モデル学習/画像処理/Flask データ取得、データクレンジング、機械学習基礎、ディープラーニングによる解析方法を扱い、アプリ開発を演習する。</p> <p>03 発展学習 (5箇月)： Kaggle「タイタニックの生存率予測」 Kaggleの利用方法を学びながら実装を行う。</p> <p>04 実践学習 (6箇月)： オリジナルアプリ開発 テーマ選定、分析モデル作成、AIアプリ実装、レポート作成までを一気通貫で行う。</p>
評価	評価方法は非公開または記載なし
所見	<p>○「独力で課題を発見して解決できるAIエンジニア」「独力で企画・設計・開発までできるAIエンジニア」を目指すカリキュラムである。</p> <p>○ 学習によって作成するオリジナルアプリは、転職活動時のポートフォリオとして位置づけられている。</p>

▼【AI活用エンジニアリング教育】事例17

講座名	AZ-400： Azure PipelinesとGitHub Actionsを使用してCIを実装する
提供者	Microsoft Learn
講数	8回
概要	本講座では、Azure PipelinesおよびGitHub Actionsによる継続的インテグレーションの概念と実装手順を整理し、AZ-400の学習範囲に沿って段階的に学ぶ。
内容	<p>01 Azure Pipelinesを探索する Azure Pipelinesの概念、用語、主要コンポーネントを扱い、パイプライン戦略と責任を整理する。</p> <p>02 Azure Pipelineのエージェントとプールを管理する ホスト型エージェントとセルフホステッドエージェントの違い、ジョブ種別、エージェントプール構成を扱う。</p> <p>03 パイプラインとコンカレンシーについて説明する 並列ジョブの見積り、オープンソース向けAzure Pipelines、Visual DesignerとYAMLパイプラインを扱う。</p> <p>04 パイプライン戦略を設計して実装する パイプライン戦略、構成、マルチエージェントビルド、ソース管理の観点を扱う。</p> <p>05 Azure Pipelinesとの統合 テンプレート、YAMLリソース、複数リポジトリ利用等の観点を扱う。</p> <p>06 Azure Pipelinesを使用してパイプラインを実装する YAMLの基本、ステージ、ジョブ、タスクの観点を扱う。</p> <p>07 GitHub Actionsを探索する GitHub Actionsの概念、用語、主要機能を扱う。</p> <p>08 GitHub Actionsを使用してワークフローを実装する ワークフロー、イベント、ジョブ、アクションの観点を扱う。</p>
評価	評価方法は非公開または記載なし
所見	<ul style="list-style-type: none"> <li>○ Azure、Dynamics、Power Apps、AI等、多岐にわたるMicrosoft製品や最新技術のコースが用意されている。</li> <li>○ 学習コンテンツは無料。演習でAzureを利用する場合は、サンドボックスや無料試用／従量課金の案内に従う。</li> <li>○ ラーニングパスとモジュールにより、目的に応じた学習方法を選択できる。</li> <li>○ 手を動かして学べるサンドボックス環境が提供されている。</li> </ul>

▼【AI活用エンジニアリング教育】事例18

講座名	AWS SimuLearn： ジェネレーティブ AI 実践者
提供者	AWS Skill Builder
講 数	10回
概 要	本講座では、Generative AIの基礎概念と実装の要点を、シナリオ型の学習（SimuLearn）を通じて扱う。
内 容	<p>01 導入 学習全体の位置づけと進め方を整理する。</p> <p>02 基礎概念 基礎概念を整理する。</p> <p>03 ユースケース 実際のアプリケーションの観点を扱う。</p> <p>04 設計観点 エンタープライズ対応の観点を扱う。</p> <p>05 実装観点 実装の要点を扱う。</p> <p>06 検証観点 検証と改善の観点を扱う。</p> <p>07 応用 応用トピックを扱う。</p> <p>08 総合演習 シナリオに沿って実装演習を行う。</p> <p>09 発表 成果を整理し共有する。</p> <p>10 総括 学修内容を整理し総括する。</p>
評 価	評価方法は非公開または記載なし
所 見	<ul style="list-style-type: none"> <li>○ 数百以上の無料デジタルコースと、有料のサブスクリプションのコンテンツがある。</li> <li>○ セルフペース学習や法人向け機能がある。</li> <li>○ デジタルコース、ハンズオンラボ、ゲーム形式等、多彩な学習形式がある。</li> </ul>

▼【AI活用エンジニアリング教育】事例19

講座名	生成AIを使用したアプリケーションの構築とモダナイズ
提供者	Google Cloud Skills Boost
講数	6回（主要6項目）
概要	本講座では、生成AIを開発ワークフローへ統合するためのテクニックおよびツールを整理し、ハンズオンでGeminiとStreamlitを用いたアプリケーション構築を行う。全体は12アクティビティで構成されるが、本整理では主要6項目を講数として扱う。
内容	<p>01 アプリケーション開発者向けのGemini Geminiがアプリケーション構築にどのように役立つかを扱う。</p> <p>02 エンドツーエンドのSDLCのためのGemini 開発、テスト、デプロイ、管理の各局面におけるGeminiの活用観点を扱う。</p> <p>03 Google Cloudでの生成AIアプリの作成 プロンプト設計およびRAGの観点を含め、生成AIアプリケーション構築の考え方を扱う。</p> <p>04 Google Cloudにおける生成AIを使用したウェブサイトのモダナイゼーション Vertex AI Search等を用いた生成検索機能の提供方法を扱う。</p> <p>05 Vertex AIとFlutterによる生成AIエージェントの構築 FlutterアプリとGeminiの統合方法を扱う。</p> <p>06 GeminiとStreamlitによる生成AIアプリケーション構築 ハンズオンにより、生成AIアプリケーションの構築を行う。</p>
評価	評価方法は非公開または記載なし
所見	<ul style="list-style-type: none"> <li>○ 生成AIの機能を開発ワークフローに統合するためのテクニックおよびツールを扱う。</li> <li>○ ハンズオンラボにより、GeminiとStreamlitを用いたアプリケーション構築を行う。</li> </ul>

▼【AI活用エンジニアリング教育】事例20

講座名	大規模言語モデル (LLM) による迅速なアプリケーション開発
提供者	NVIDIA DLI
講 数	8回
概 要	本講座では、オープンソースのLLMエコシステムを俯瞰し、LLMアプリケーション開発を迅速に開始するための基礎と実装を扱う。
内 容	<p>01 LLMアプリケーション開発の概観 LLMアプリケーション開発の全体像を整理する。</p> <p>02 オープンソースLLMエコシステム 主要要素と位置づけを整理する。</p> <p>03 環境構築 実行環境の構成と基本操作を扱う。</p> <p>04 推論と評価 推論実行と評価観点を扱う。</p> <p>05 プロンプト設計 プロンプト設計の基本を扱う。</p> <p>06 RAGの基礎 検索拡張生成の基本を扱う。</p> <p>07 アプリケーション実装 LLMを用いたアプリケーション実装を扱う。</p> <p>08 総括 学修内容を整理し総括する。</p>
評 価	評価方法は非公開または記載なし
所 見	<ul style="list-style-type: none"> <li>○ LLMを含むオープンソースのエコシステムを探求し、開発に必要な理解と実践知を扱う。</li> <li>○ LLMベースのアプリケーション開発を迅速に開始することを狙いとする。</li> </ul>

## 【B】ヒヤリング

本調査【B】では、AI活用エンジニアリング（教育）の設計原理や設計要件について理解を深めるために、教育現場および開発現場からの知見を得るべく、2人の有識者に**ヒヤリング**を行った。

具体的には、AI活用エンジニアリングの定義と責任分界、そこでの品質・生産性の再定義、人材要件の層構造、教育設計原理、評価の正当性等に関する5テーマについて、両者の見解を得た。

### 回答者

#### 【教育現場】私立大学 情報学部 准教授（44歳）

東京都の私立大学情報学部所属。専門はソフトウェア工学・開発プロセスおよび工学教育。かつてSIerでの開発経験があり、現在は「AI活用ソフトウェア開発演習」を科目責任者として運用。生成AIは教育上“前提技術”と捉え、学修成果の真正性、評価の妥当性、学生の説明責任（何を・なぜ・どう作ったか）の担保を重視。学内では生成AI利用方針の整備・FDにも関与し、学習ログや評価の透明性に意識が高い。

#### 【開発現場】システム開発企業 プロダクト事業部 課長（41歳）

東京都のシステム開発企業（受託+自社開発）プロダクト事業部所属。課長として15人規模の開発組織を管理し、品質・納期・顧客説明の責任を負う。若手研修や社内勉強会を継続的に主宰し、自身も外部研修・コミュニティに積極的に参加。生成AIは生産性向上の“必須インフラ”と捉える一方で、情報管理・契約・著作権/ライセンス・監査可能性を経営リスクとして重く見る。

## 1) AI活用エンジニアリングの定義と責任分界の設計原理

「AI活用エンジニアリング」を貴職の立場で定義すると、それはどのようなものでしょうか。その必要条件と十分条件は何でしょうか。

また、AI活用を前提にした時、その責任分担（誰が・何に・どこまで責任を負うべきか）は、どうルール付けることが可能でしょうか。

例えば、成果物の正しさ・安全性・説明可能性・再現性のうち、最終的に“人が引き受けるべき”ものは何で、それはなぜでしょうか。

さらに、AIの介在を前提にした時、“能力がある”とは、いかなる状態を言うのでしょうか。作れること・検証できること・説明できること・運用できること等のどれを核に据えるべきでしょうか。

### 【教育現場】回答：

私の立場では、AI活用エンジニアリングは「AIを工程に組み込みつつ、成果物の根拠・検証・説明の設計を人が引き受ける開発実践」です。必要条件は「AI利用が明示され、検証が工程化されていること」、十分条件は「第三者が追試できる証跡で品質を再現できること」です。責任は最終的に人が負い、とりわけ正しさ・安全性・説明可能性は不可譲です。能力の核は「検証でき、説明できる」ことで、作れることはその従属に置きます。本事業のBASISで責任分界をスキル要件に落とし、PBLで実務の説明責任を訓練する思想は妥当です。他方で、AI操作が上手いことを能力と誤認する設計は避ける必要があります。

### 【開発現場】回答：

私は、AI活用エンジニアリングを「AIを手段として使い、価値・リスク・説明責任を人が管理して成果を出す体系」と定義します。必要条件は「入力統制と承認線が定義されていること」、十分条件は

「品質・納期・法務を含めて安定運用できること」です。正しさ・安全性・再現性は契約責任の中核なので、最後は人が引き受けるしかありません。能力の核は「要件を翻訳し、AI出力を査読し、運用に耐える形に整える」ことで、運用まで含めて見ます。本事業が指標化（BASIS）と実践（PBL）を併置しているのは、現場の“責任の型”を作る上で有効です。懸念は、説明責任が強い業界ではルールが重くなり、学修設計が軽量化されないと定着し難い点です。

## 2) 品質と生産性の再定義（最適化対象の更新）

AIが開発に介入する時、品質と生産性は従来の定義のまま最適化できない局面が増えると思われまます。責職の文脈では、今後の品質をいかなる要素（正確性・安全性・保守性・監査可能性・社会的信用等）の集合と捉え、どの要素を最上位の目的（譲歩できない基準）に置くべきでしょうか。そして、速度と品質のトレードオフを議論する場合、意思決定の基準（どの価値をどの重みで評価するか）をどう明文化すべきでしょうか。また、AI導入によって“改善したように見え易い指標”と“悪化しても気づき難い指標”は何でしょうか。それはなぜ可視化され難いのでしょうか。

### [教育現場] 回答：

私の文脈では、品質は「正確性・安全性・保守性」に加え、「監査可能性（根拠が辿れる）と学修の誠実性」を束として捉えます。最上位は安全性と説明可能性で、教育では“説明できない成果”は品質未達と見做します。価値の重み付けは、成果物価値だけでなく学修価値（どの能力証拠が残ったか）を明文化して合意する必要があります。改善して見え易いのは“出力量”や“短期の作業速度”で、悪化に気づき難いのは“理解の空洞化”と“保守負債”です。後者は後工程でしか症状化しないためです。本事業が到達指標と演習を接続し、改善サイクルを回す設計はこの歪みの補正に資します。注意点は、指標が目的化すると学修がゲーム化することです。

### [開発現場] 回答：

企業の品質は「正確性・安全性・保守性・監査可能性・対外信用」の総体で、譲歩できないのは安全性と信用です。速度と品質の重み付けは、リスクと影響範囲で決めるしかなく、重要度に応じて“止める基準”を先に定義します。AIで良く見える指標はベロシティや消化量で、悪化し易いのに見え難いのは再作業率・潜在不具合・レビュー負荷・属人性などです。短期では数字が良くても、後でコストとして噴き出すためです。本事業が“AI+DevOps”を前提にして品質と速度を同時に扱う思想は開発現場に合っています。課題は、評価が軽量でないと中小企業ほど運用に乗り難い点です。

### 3) 人材要件の層構造と希少性の移動

今後2～3年、5年で、AI普及によって「価値を生む能力」(希少性の源泉)はどこに移ると見立てますか。その人材要件を次の3層で整理してください。

① AIに吸収され、相対的にコモディティ化する能力

② 時代が変わっても普遍的な基盤として残る能力

③ AI時代に新たに重要度が上がる“メタ能力”(問題設定・検証・統治・説明責任の引受け等)

その上で、③の中核になる最小要件を優先順に挙げてください。その際、その各々について“出来ていると判断できる具体的な兆候”(観測可能な行動や成果)も挙げてください。

#### [教育現場] 回答:

①は定型実装、雛形生成、表層的な文章化です。②は計算機基礎、データ構造、設計思考、テスト観、そして倫理です。③が希少性の核で、優先順は(1)問題設定(何を解くべきかを定義する)＝要求を仮説として書ける、(2)検証設計＝反証可能な評価基準と手順を作れる、(3)説明責任＝前提・根拠・限界を文章と口頭で示せる、(4)統治＝利用ルールを守りつつ改善提案できる、です。観測可能な兆候は、仕様・設計・テストが一貫し、他者が追試できる成果物が残っていることです。本事業がBASISで③を明文化し、PBLで能力証拠を作らせるのは有効です。懸念は、題材が局所に偏ると普遍化が弱くなるため、抽象化の指導が要ります。

#### [開発現場] 回答:

①はコーディングの多く、単体テストの雛形、ドキュメントの一次稿などです。②はアーキテクチャ、データモデリング、ネットワーク/セキュリティ基礎、顧客理解です。③は(1)要件を価値と制約に翻訳する力＝曖昧な依頼を仕様で落せる、(2)リスク見積りと品質戦略＝どこを機械に任せ、どこを人が握るかを決められる、(3)監査可能な実装運用＝ログと根拠を残し、運用まで面倒を見られる、(4)合意形成＝ステークホルダーを動かせる、が最小セットです。兆候は、レビューで論点を言語化し、事故の芽を早期に潰せることです。本事業の“AI補完型”という捉え方は現実的で、地方の中小企業でも再現し易いと思います。課題は、万能人材像に寄せると浅くなるため、到達像を役割別に切る設計が要ります。

### 4) 学びの設計原理(順序でなく設計思想)

「何をどう学ぶか」を(科目やツールの列挙でなく)教育設計の原理として捉えた場合、それはいかなるものと考えますか。

例えば、生成AIを早期に使わせる設計と、基礎を固めてから使わせる設計であれば、いずれを基本方針に置くべきでしょうか。その方針が成立する前提条件と破綻し易い条件は何でしょうか。

また、学びを「0～3箇月」「3～12箇月」「1年以降」に区分した場合、各期の到達を“知識”でなく、いかなる能力証拠(アウトプットの性質)で示すのが妥当でしょうか(説明責任を果せること、検証を設計できること等)。

#### [教育現場] 回答:

原理は「作る・検証する・説明する」を反復し、証拠を残しながら抽象化することです。AIは早期に触れさせますが、前提は“制約付き”です(利用申告、根拠提示、一定区間の非AI演習)。破綻は、AIが

思考の代替になるときに起きます。0～3箇月は小さく作って根拠と限界を書けること、3～12箇月は仕様・テスト・運用観点を含む成果物を残せること、1年以降は改善提案とガバナンスを含めて自走できることが証拠になります。本事業のハンズオン中心設計は合致します。注意は、指導者の設計力が不足するとAI依存が加速する点です。

#### [開発現場] 回答：

原理は「再現可能な仕事の型」を身に付け、成果の説明責任を果せる状態にすることです。基本は“基礎の最低ラインを作った上でAIを使う”ですが、早期活用も否定しません（条件は入力統制とレビュー文化です）。破綻は、基礎なしで速度だけを追う時です。0～3箇月は小さな変更を安全に通す証拠、3～12箇月はチーム開発で品質ゲートを回せる証拠、1年以降は設計判断と運用の引受けの証拠が妥当です。本事業が実践と指標を往還させる構想は育成に効きます。懸念は、就労者は学習時間が希少ですから負荷設計が鍵です。

### 5) 評価・指標化・バッジの正当性と、AI利用ルールの社会契約

本事業で開発を試みるBASISのような指標化やバッジ等による可視化は、設計を誤ると形骸化・ゲーム化し易いと考えます。AI時代の能力評価を正当化するためには、最低限いかなる条件（証拠・透明性・異議申立て可能性等）が必要でしょうか。

指標を置くことで生じ得る典型的な歪み（例えば測定が目的化する問題）を、設計段階でどう抑制すべきでしょうか。また、生成AI利用に関する最低限のルールを（“守るべき倫理”でなく）能力形成を可能にする制度的前提（社会契約）として定義する場合、いかなる原則を中核に据えるべきでしょうか（情報管理・著作権・ライセンス・ログ管理・説明責任等を含む）。

#### [教育現場] 回答：

正当化の条件は、能力証拠（成果物+プロセス）の提示、評価基準の透明性、異議申立て可能性、そして機械評価の人手による校正です。歪みは“点を取る行動”が最適化されることなので、単発試験よりポートフォリオ・口頭防衛・相互レビューで三角測量します。社会契約の中核は「利用の開示」「根拠と限界の明記」「情報の最小化」「著作権/ライセンス尊重」「ログで説明可能にする」の5点です。本事業のBASISは、これを教育設計に埋め込める点が強みです。懸念は、ログ収集が過度になると監視に見えて学修者の自律性を損なう点です。

#### [開発現場] 回答：

企業で正当化するためには、職務との対応、評価の一貫性（誰が見ても同じ判断）、監査可能性、異議申立ての手順が必要です。ゲーム化はバッジ取得が目的になることなので、実務成果・レビュー評価・上長判断と必ず接続します。社会契約の原則は「投入してよい情報の線引き」「成果物責任は人が負う」「AI利用の記録と再現性」「IPとコンプライアンス順守」「逸脱時のエスカレーション」です。本事業がこれらを標準化し、地域で共通言語にする狙いは高く評価します。課題は、運用が重いと中小企業が形だけになり易いので、最小実装の提示が要ります。

## 2.5.3 調査考察

本調査は、AI活用エンジニアリング教育の現状を教育内容（何を教えるか）と教育設計（どう教え、どう測るか）の両面から捉え、本事業で開発する教育プログラムの構成要件を抽出するための基礎資料として有効である。

特に、**事例01**（Copilot×IoT短期ハンズオン）・**事例03**（Dify等を含む生成AIアプリ開発を“入門／実践”で接続）・**事例17～20**（Microsoft Learn／AWS Skill Builder／Google Cloud Skills Boost／NVIDIA DLIのモジュール学習）を同一フレームで並置したことで、AI活用が単一のスキルでなく対象（AIを作る、AIで作る、AIを統治して作る）と責任（誰が何を引き受けるか）の組合せで成立することが明確になった。

### 20事例が示すAI活用の3類型と教育が陥り易い錯覚

本調査で扱った20事例は概ね次の3種に分類できる。

- i) **AIを作る教育**： **事例07**（深層学習フレームワーク実装）・**事例10**（強化学習Ⅱ）等。理論・実装の正統性を核に据える一方で、生成AI時代の工程統合（品質保証・運用）までは射程外になり易い。
- ii) **AIで作る教育**： **事例01**（Copilotで実装支援）・**事例02**（UX/UI設計から実装まで）・**事例06・09**（Web/C/C++の演習）等。学習速度を上げ易い反面、“作れた＝能力がある”という錯覚を生み易い。
- iii) **AIを組み込んだプロダクトを作る教育**： **事例19**（Gemini+Streamlit等での生成AIアプリ構築）・**事例20**（LLMアプリ開発の概観～RAG）・**事例15**（AWS AIサービス統合）等。実装は現代的であるが、統治（情報管理・説明責任）を“別科目扱い”にすると現場移転が弱くなる。

この3類型が混在する現状で最も危険なことは、ii) iii) の学習成果を“出力物の見栄え”で判定し、検証・説明・再現の能力を取り逃すことである。**事例04**は開発課題の実装状況と最終レポートを50点ずつ配点して合計60点以上合格という形で“成果物+説明”を最低限に接続しているが、他事例では評価情報が「非公開または記載なし」のものが多く、教育設計としては“学び方はあるが、測り方が弱い”傾向が窺える。**事例12**は最終提出物・出席・授業中提出物という配点を明記して透明性の点で優れるが、生成AI介在時代の中核である根拠提示・検証設計・再現可能性をどこでどう測るかは、さらに踏み込む余地がある。

### ヒヤリングが補った“責任分界・品質観・人材要件”のメタ論点

ヒヤリング（教育現場／開発現場）で収穫が最も大きかったのは、両者が共通してAI活用を“生成の巧さ”でなく“責任を引き受ける設計”として定義した点である。教育の立場では、能力の核を“検証でき、説明できる”に置き、成果物だけでなくプロセス（根拠・限界・追試可能性）を能力証拠にすべきと整理した。開発の立場では、“入力統制と承認線”と“安全性と対外信用”を最上位に置き、AI活用は開発技術というより“統治を含む体系”と位置付けた。

この一致は、20事例の“技術領域の広さ”と対照的に、教育として収斂すべき中核が“説明責任を果せるエンジニアリング”にあることを示唆する。すなわち、AI時代の希少性は“書ける”から“何を解くべきかを定義し、どう正しさを担保するかを設計し、なぜそうしたかを説明できる”に移る。これは、**事例03**のようにDevSecOps/SREまで視野に入れて“実装の先”を教える設計が持つ意義とも整合する。

## 本事業への含意（鋭さが要るのは“技術選定”でなく“証拠設計”）

本調査の価値は、個別ツールや流行の棚卸しでなく、教育設計上の盲点——「AIが介在するほど品質は“監査可能性”（証拠）としてしか担保できない」——を浮き彫りにした点にある。ここで言う証拠は、単なるログ収集でなく、〈要求→設計→実装→検証→運用判断〉を一貫して辿れる“説明の鎖”である。有識者が言う「作れることを核に置かない」（教育）と「責任の型を回す」（開発）という見立てを合せると、本事業の教育プログラムは次の要件に収斂する。

- BASISは、ツール技能の羅列でなく、問題設定・検証設計・説明責任・統治の“メタ能力”を中心に据える（事例の多様性に耐える唯一の設計）。
- 演習・PBLは“成果物提出”で終えず、“採用／棄却の判断根拠”・“検証の設計”・“再現手順”を必須成果物にする（事例01の短期開発、事例19のハンズオンを、責任ある工程に昇格させる）。
- 企業現場（特に地方中小）に実装可能にするため、有識者（開発）が懸念した通り、統治は重装備にせず、最小実装の社会契約（投入禁止情報／利用開示／IP配慮／ログで説明可能）として提示する。

結論として、本調査は“教育事例の収集”に留まらず、AI活用エンジニアリング教育の本丸が“生成”でなく“責任・品質・説明の設計”にあることを、事例分布と有識者意見の両面から確認できた。次段階では、20事例の広さを維持しつつ、到達目標を“証拠生成型の開発実践”に一点収束させ、評価（BASIS・バッジ）を“成果物＋プロセス＋説明”で三角測量する設計に落とし込むことが、本事業の差別化と地域実装の成否を決める。

\*

本章では、教育モデル構築に先立ち、本年度の活動である「教育プログラムの基礎設計と初期教材開発」の前提を固めるために実施した4調査（地方企業生成AI活用実態調査（2.2）・IT“学び直し”ニーズ実態調査（2.3）・生成AIコーディングアシスタント事例調査（2.4）・AI活用エンジニアリング教育事例調査（2.5））の結果を整理した。

調査①では、生成AIの導入は一定程度進む一方で、利用はチャットボットや汎用利用に限定され易く、活用領域も“情報収集・文書作成”等に偏り、業務プロセスに組み込む“運用”が薄いまま局所最適に留まり易いことが明かになった。また、現場の主なボトルネックは、ツール機能そのものでなく、“安心して使える条件”（セキュリティ・法令順守、社内ルール未整備等）にあり、DevOps／運用自動化まで視野を拡げた実装・定着の難しさが示唆された。

調査②では、人口減少地域における“学び直し”需要は二極化しつつも、最大の障壁は「時間（業務都合）」であることが確認され、短時間・低負荷で継続できる学習設計が不可欠であることが浮き彫りになった。受講形態は、オンデマンド（講義・自習）を基軸にし、要所でライブ（伴走支援）を組み合わせるハイブリッドが現実解で、学習“意欲”だけでなく学習“条件”の整備を前提にした設計が求められる。

調査③は、生成AIコーディングアシスタントが“補完”から“工程横断（SDLC）×統制×エージェント”に拡張している現状を踏まえ、教育として扱うべき中核が特定ツールの操作でなく、地方の小規模チームでも

再現可能な“選定・統制・検証”の型であることが明らかになった。併せ、差分・実行ログ・テスト結果を保存できるクラウドIDE等、振り返りと評価に接続できる演習環境要件に落とし込む方向性が示された。

調査④では、AI活用エンジニアリング教育の20事例を“AIを作る／AIで作る／AIを組み込んだプロダクトを作る”に類型化し、出力物の見栄えのみで成果を判定すると、検証・説明・再現の能力を取り逃す危険を指摘した。ヒヤリング結果も踏まえ、AI時代の中核は“生成の巧み”でなく“責任を引き受ける設計”であり、成果物に加えて根拠・検証設計・再現手順までを能力証拠として要求する評価観点（＝監査可能性としての“証拠”設計）が重要であることが、本章全体を通じて一貫して示された。

以上から、本事業が目指す「AI補完型ITエンジニア」の養成には、特定ツールの操作習得ではなく、地域制約下でも実装できる最小統治の前提を整えつつ、工程全体でAIを扱い、証拠（ログ・差分・テスト等）に基づいて判断と説明を行える実践力を育む学習・評価設計が要件になる。



## 3.1 開発概要

本事業では、生成AIを駆使して情報システム開発の速度と質を飛躍的に向上させた次世代型IT人材（AI補完型ITエンジニア）を人口減少地域において持続的に養成し、地域産業のDXを自走させる基盤を構築することを目指す。

そのために、この人材のスキル要件を定義し、その修得度を評価するための指標（AI駆動ITスキル指標）と、それを“礎”にした教育プログラム（AI補完型ITエンジニア養成プログラム）を開発する。

そして、3箇年度計画の初年度である本年度（令和7年度）事業では、その教育プログラムの基礎設計と初期教材開発に取り組んだ。具体的には、次の9件が本年度事業における開発案件である。

### 開発01 AI駆動ITスキル指標設計

AI補完型ITエンジニアに必要なリテラシー&スキルを項目化・体系化した評価基準の設計

### 開発02 GPT=TA評価システム設計

提出課題等を生成AIで自動採点し、BASISに照して学習到達度を可視化するクラウド型評価システムの設計

### 開発03 教育プログラム全体カリキュラム開発

AI補完型IT人材養成プログラム（全60コマ）の教育体系の構築

### 開発04 教育プログラムシラバス作成

同プログラム【科目1】【科目2前半】（12コマ）の指導計画の策定

### 開発05 教育プログラム講義動画制作

同プログラム同科目の講義（7コマ）の動画（AIアバター指導者/mp4）の制作

### 開発06 教育プログラム講義資料作成

同プログラム同科目の講義（7コマ）の資料（スライド・ノート等）の作成

### 開発07 教育プログラム演習課題作成

同プログラム同科目の演習（5コマ）の課題・指導ガイド・シナリオの作成

### 開発08 教育プログラム自習問題作成

同プログラム同科目の自習（12コマ）のCBT問題の作成

### 開発09 教育プログラム基盤要件定義

同プログラムをオンラインで提供するLMS・クラウドIDE等を統合した教育プラットフォームの設計

さて、本書では次節以降、この**開発01～09**の各件について詳説するが、本書における“教育プログラムの捉え方”に基づき、各件を次のように各節に配置する。

#### 教育プログラムの目標を定める——3.3 (教育目標)

**開発01** AI駆動ITスキル指標設計

#### 教育プログラムの体系を組む——3.4 (教育体系)

**開発03** 教育プログラム全体カリキュラム開発

**開発04** 教育プログラムシラバス作成

#### 教育プログラムの内容 (コンテンツ) を作る——3.5 (教育素材)

**開発05** 教育プログラム講義動画制作

**開発06** 教育プログラム講義資料作成

**開発07** 教育プログラム演習課題作成

**開発08** 教育プログラム自習問題作成

#### 教育プログラムの基盤を整える——3.6 (教育基盤)

**開発09** 教育プログラム基盤 要件定義

**開発02** GPT-TA評価システム設計

この“教育プログラムの捉え方”は、本事業における**教育モデルの構造**に基づくものである。その道筋を次節 (3.2) で辿る。

## 3.2 教育モデルの構造

### 教育モデルの目標——AI駆動ITスキル指標

本事業の目的は、人口減少地域で不足・流出しがちなIT人材の課題を踏まえ、生成AIを実務に組み込みつつ要件定義から運用改善までを継続的に遂行・主導できる次世代型高度専門職=**AI補完型ITエンジニア**を養成する**教育モデル（地域DX次世代型AI×IT人材循環モデル）**を構築することにある。

この目的を教育機関（専修学校等）での実装可能な形に落とし込むため、本事業では「何を以て〈到達〉と見做すか」を先に固定し、教育の設計と運用をその〈到達〉の定義に従属させる構造を設計する。

この中核に据えるのが、**AI駆動ITスキル指標（BASIS）**である。これは、単なるスキル項目一覧でなく、〈到達〉レベルの定義と評価可能性を含む“教育目標の体系”として機能する。BASISが教育目標として成立し得るのは、例えば既存の公的なスキル標準（iCD・DSS・ITSS等）が職能をドメイン化して能力を構造化し、それを評価可能な形で社会に提示するという枠組を既に正当化しているのと同様である。そのため、本事業のBASISは、その公的枠組に照して説明可能であること、必要に応じて対応付け（マッピング）が可能であることにより、本教育モデルの規範性と対外説明力を担保する。

### 教育モデルの内部機構——教育システムと教育プログラム

この事業目的とBASISを土台にして設計される最上位の枠組が教育モデルである。教育モデルとは、教育内部の仕組を閉じた形で完結させるのではなく、〈教育→評価→社会実装→定着〉という循環を成立させる全体設計を言う。

その際、教育モデルは、**教育システム**と**社会接続施策**で構成される。教育システムは教育の内部で人材を養成して到達度を測定・保証する仕組であり、社会接続施策は教育の成果を就職・定着や地域企業の実装課題に接続して循環を閉じるための外部施策群である。特に後者は、「修了後メンタリング」「人材×企業マッチング」「企業DXクリニック」等を含み、教育で生れた能力証拠や成果物が就職時の評価や地域企業の課題解決に結び付く経路を提供する。これにより、教育が単なる学習活動に終らず、地域の内製力強化と人材定着にも因果的に繋がる構造が確保される。

他方で、教育モデルの内部核である教育システムは、BASISと**教育プログラム（AI補完型ITエンジニア養成プログラム）**で構成される。その際、BASISは教育システム内の上位規範であり、教育プログラムはBASISを達成させる手段装置である。つまり、教育プログラムは、BASISを目標に掲げ、その目標に整合する形で教育内容と評価方法と運用環境を組み立てることで初めて、教育システムとしての一貫性を獲得するのである。

そのため、教育プログラムは更に二分され、**教育体系**（内容部）と**教育基盤**（運用部）で構成される。教育体系は“何を学ばせるか”を構造的に配置したもので、教育基盤は“それをどう提供し、どう測り、どう回すか”を実装する環境である。教育体系と教育基盤を分離することで、内容の改訂と環境の更新を別管理でき、継続的改善の実務性や可搬性（他校展開）が高まる。

## 教育プログラムの体系——カリキュラムとシラバス

この教育体系は**カリキュラム**として具体化される。カリキュラムは、科目構成・講配列・時間配分・到達目標の配列により、BASISで定義された〈到達〉をいかなる順序で実現するかを設計する骨格である。

そして、カリキュラムは各科目・各講の**シラバス**に分解され、シラバスは、科目目的、各講の教育内容と科目内位置付け、学習項目（重要概念・キーワード等）、評価方法、評価項目等を明示することで、カリキュラムの設計を授業運用に落とし込む。その際、本事業で言うシラバスは単なる“科目概要文書”でない。それは同時に**指導計画**でなければならない。指導計画は、授業担当者がその教育内容について、どの順で説明し、どの点で躓きを想定し、それをどう確認し、いかなる基準で評価するかを、運用手順として具体化したものである。

かくして、教育体系の階層は、概念設計（方針：到達目標）から運用設計（実装：授業実践）に粒度を段階的に上げながら、BASISとの整合を失わないように相互に接続している。

## 教育プログラムの内容——教育5形態と教育コンテンツ

この指導計画が具体化されたものとして、受講者が学習時に直接に触れる実体が**教育コンテンツ**である。教育コンテンツは、本教育プログラムの教育5形態（講義・演習・実習・プロジェクト型学習（PBL）・自習）の各教材・問題・課題等から成る。

講義は、教育形態の中核で、その動画と教材は概念枠組や開発工程の理解を担保する。

演習は、講義で得た理解を手順化し、小規模実装として確かめることで、知識を技能に転換する。

実習は、講義と演習で培った知識と手順を統合し、一定の機能やサブシステムを自律的に（時に指導と共に）設計・実装・検証することで、個別技術を一体の成果物に結実させる。

プロジェクト型学習（PBL：Project-Based Learning）は、実在の地域企業（クライアント）の実課題に対し、要件定義から設計・実装・評価まで通貫して遂行し、社会的責任を伴う成果物として納品可能な水準に仕上げる。

自習は、講義に附随・準拠してその内容の理解度を測定するCBT（Computer-Based Testing）で、理解の空洞化を早期に検知し、受講者に再学習を促すための評価・点検装置として配置される。その際、自習問題は、単なる“おまけ”でなく、BASISが求める〈到達〉を日々の学習行動に落とし込み、受講者の到達度を連続的に可視化するための末端機構である。そのため、自習問題が教育体系の最下層に位置するのは偶然でなく、BASISを到達目標として成立させるために到達評価の粒度を細分化し、学習の循環を回すために必須の構成要素として設計されている。

## 教育プログラムの基盤——教育プラットフォームとGPT-TA評価システム

他方で、**教育基盤**は、教育体系の運用環境で、**教育プラットフォーム**と**GPT-TA評価システム**を中核に据える。

教育プラットフォームは、データベース・ストレージ・学習管理システム（LMS：Learning Management System）・クラウド統合開発環境（IDE：Integrated Development Environment）・配信機能・課題提出機能等を統合し、教育コンテンツの提供と共に学習行動ログ・成果物等を回収するオンライン教育基盤である。

GPT-TA評価システムは、教育プラットフォームに搭載された、生成AIを活用した指導支援・評価支援装置で、CBT解答や提出課題を自動採点するだけでなく、外部の大規模言語モデル（LLM：Large Language Model）と連携し答案内容に応じた記述フィードバックを生成し、受講者に対して具体的な改善点を自然言語で提示する。

さらに、学習ログや得点推移を横断的に分析し、受講者の理解傾向や躓き箇所を抽出することで、教育提供側に指導方針や教材内容の改善提案を行う機能を有する。これにより、本システムは単なる採点装置に留まらず、教育プログラム全体の更新優先順位を可視化する循環型の評価・改善システムとして機能する。

これらにより、教育体系が定めた“何を学ばせるか”が、教育基盤によって“どう実施し、どう評価し、どう改善するか”に接続される。最終的に、〈BASIS (到達目標・評価基準) → 教育体系 (内容設計) → 教育コンテンツ (学習実体) → 教育基盤 (蓄積・提供・評価・可視化)〉という一連の環が閉じることで、本教育プログラムは単なる教材群でなく、到達目標に対して測定可能で改善可能な教育システムとして成立する。

そして、この教育システムの外部に「人材×企業マッチング」「企業DXクリニック」等の**社会接続施策**が配置されることで、受講者が獲得したBASIS準拠の能力証拠が、就職・定着や企業課題の解決に接続される。

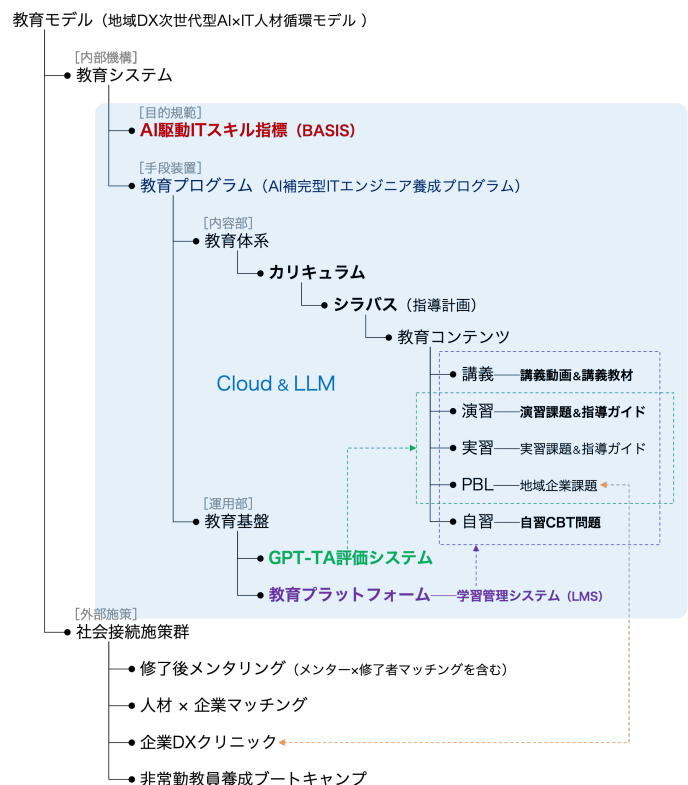
\*

以上のように、本事業で構築する教育モデルの構造は――

- ① 事業目的が規定する人材像 (AI補完型ITエンジニア) に基づき、教育モデルが構想される。
- ② 教育モデルは、内部機構 (教育システム) と外部施策 (社会接続施策) の二層から成る。
- ③ 教育システム内に、目的規範としてAI駆動ITスキル指標 (BASIS) が置かれ、その手段装置として教育プログラムが設定される。
- ④ BASIS到達のために、教育プログラムが教育体系と教育基盤に分解される。
- ⑤ 教育体系がカリキュラム→シラバス→教育コンテンツとして具現化され、教育基盤が教育プラットフォームとGPT-TA評価システムを通じて教育体系を運用する。
- ⑥ これにより、教育システムが末端の自習問題に到るまで到達評価を循環させる。

このような関係の連鎖によって説明できる

(右図参照)。



### 3.3 教育目標——AI駆動ITスキル指標 (BASIS)

本事業では、生成AIを駆使して情報システム開発の速度と質を飛躍的に向上させた次世代型IT人材（AI補完型ITエンジニア）を養成する教育プログラムを開発する。

そして、これに先立ち、この**AI補完型ITエンジニア**のスキル要件を定義し、その修得度を評価するための指標である**AI駆動ITスキル指標 (BASIS : Benchmark for AI-Supported IT Skills)**を策定する。

教育プログラムの開発に際し、拙速に教育コンテンツ（講義動画・教材等）の制作から始めても、到達すべき教育目標や評価基準が不明確なままでは学修成果の一貫性・再現性を担保できない。そのため、まず養成する人材像を定義し、その人材が0.5～1年で“要件定義→実装→運用自動化”を自走できるようにするための人材要件・スキル要件を明確にする必要がある。

BASISは、そのスキル要件を定義してその修得度を評価するための指標であり、教育プログラムの到達目標を可視化すると共に、GPT-TA評価システムの自動採点結果を“ドメイン別バッジ”として発行するための参照軸になる。

そのため、本年度事業における開発工程では、このBASISの策定を起点にし、GPT-TA評価システムと共に教育プラットフォームの設計、教育プログラムのカリキュラム・シラバスの開発、そして教育コンテンツ（講義動画・講義資料・演習課題・自習問題等）の制作へと段階的に展開する。

本節では、この後続工程に共通する装置として、BASISの枠組と評価観点を整理し、必要に応じて既存の公的スキル標準（iCD/DSS/ITSS等）との参照関係も点検することで、指標の妥当性と可搬性を確保する。

#### 3.3.1 BASIS設計過程

本事業の出発点には、人口減少の進行に伴って地方のIT人材の不足・流出が加速し、情報基盤更新の遅延が地域産業の技術停滞に直結し得るという構造的課題がある。それと共に、生成AIの高度化・商用化がシステム開発の速度と品質に“新基準”を齎し、地方企業が従来体制のまま追随することを難しくしている。他方で、クラウド環境や遠隔教育の普及によって“距離の壁”が低くなり、都市部の教育資源を地方に届ける基盤が整いつつあるが、AI支援コーディングは生産性向上と同時に欠陥・脆弱性混入のリスク増大も顕在化させてい、速度と品質のトレードオフを統制できなければAI導入自体が逆効果になり得る。

そのため、本事業では、地方の小規模チームが“少数精鋭+AI”で内製力を高められるよう、AIを統制的に使い熟せる実装者、すなわち**次世代型IT人材**を計画的に養成することが求められる。

この要請を受け、まず“いかなる人材を養成するか”を明確にし、その人材要件を教育目標・評価枠組に落とし込むための基準として「AI駆動ITスキル指標」(BASIS)を設計した。

その際、この養成対象人材（次世代型IT人材）は、本事業計画において次のように描出されている。

- ✓ 生成AIを武器にした次世代型IT人材
- ✓ AIコーディングアシスタントを駆使する人材
- ✓ AIが出力した成果物を精査して業務案件に落とし込めるエンジニア
- ✓ AIを有効に活用しながら情報システムを高速かつ高品質に内製できる人材
- ✓ 生成AIを駆使して情報システム開発の速度と質を飛躍的に向上させた次世代型IT人材
- ✓ 生成AIを駆使して情報システム開発のプロセスを高速化・高品質化できる「AI補完型ITエンジニア」
- ✓ 生成AIの力を実務に応用し、情報システムの要件定義から運用改善まで継続的に遂行・主導できる次世代型IT人材
- ✓ 「要件定義→実装→運用自動化」を自走できるエンジニア
- ✓ 生成AIとDevOpsの一体的活用によって要件定義から運用自動化まで自律的に遂行できる人材
- ✓ 地方で育って地方で働く次世代型IT人材
- ✓ 地方企業の内製力を底上げするDXの中核を担う人材
- ✓ 地方企業の実課題をAI駆動ITスキルによって解決し、継続的改善サイクルを組織内で主導できる人材

かような人材像記述を総合し、この次世代型IT人材の呼称を「AI補完型ITエンジニア」と確定した。

その上で、この人材の能力要件を、“開発工程における要件定義・実装・運用の各フェーズで生成AIをどう扱い、どう統制するか”という観点から、教育・評価に適用できる形として、次の4点に整理した。

#### I 生成AIを安全かつ適切に扱い、倫理的に運用できる力

モデルの限界・幻覚・バイアスを理解し、国際規格や組織ポリシーに準拠して生成物の事実性・一貫性を評価して扱える。

#### II 業務要件を設計・プロンプト・LLM連携に変換できる力

自然言語の要件をユーザーストーリー／受入基準と高精度プロンプトに落とし込み、RAG・API・マルチモーダル連携まで設計して実装可能にできる。

#### III 生成AI補完で短サイクルに実装し、テスト自動化で品質を保証できる力

Copilot等で実装を加速しつつ、静的解析・TDD/BDD・品質ゲートで“速度×品質”のトレードオフを統制できる。

#### IV 運用をセキュアに自動化し、本番移行・引継ぎまで遂行できる力

生成AI特有の脅威と法令に備え、CI/CD・IaC・Observability・AIOpsで運用を自動化し、“UAT→本番移行→運用手順化”を小規模チームで完遂できる。

この4能力は、開発する教育プログラムの教育内容を設計する際の原理であるのと同時に、受講者の到達度を判定する評価観点でもある。特に本事業では、生成AIを“代替者”でなく“補完的な道具”として制御し、生成物の妥当性を人が評価した上で、要件・品質・運用を統合して開発を推進する力を重視する。そのため、単なる知識量や作業手順の暗記でなく、AIを介した開発プロセスにおける判断・選択・統制を含む**実務能力(スキル)**を測定できる指標が求められる。

この要請に応えるべく設計したのが「AI駆動ITスキル指標」(BASIS)である。すなわち、BASISは、AI補完型ITエンジニアのスキル要件を定義し、その修得度を評価するための指標である。受講者が0.5～1年で“要件定義→実装→運用自動化”を自走できるよう、汎用ITスキルをAIで高速・高精度化する観点で再編し、「AIはツール、評価と最適化は人」という役割意識を前提にしてスキルと判断力を同時に測定する。そして、教育プログラムの到達目標を可視化し、GPT-TA評価システムによる自動採点結果をドメイン別バッジとして発行するための基準になる。

その際、BASISの枠組設計では、前述の4能力(I～IV)をその抽象的なまま適用するのではなく、現場の開発実務における“工程の連鎖”と“横断的な統制要件”を同時に表現できる構造にすることを重視した。それは、“要件定義→実装→運用自動化”という“価値実現の流れ”を軸にしつつ、生成AI時代に不可欠の倫理・規格・ガバナンス、セキュリティ、品質保証等の“横断要件”を、工程の(外部でなく)内部に組み込むことである。その結果として、能力I～IVは、次の5系の実務構造で捉え直された。

- i 生成AIを扱う前提になるリテラシー・倫理・規格
- ii LLMを使い熟するためのプロンプト設計と連携実装
- iii 要件定義から設計・実装・テストに進む中核工程
- iv 生成AI特有のリスクを包含するセキュリティ／ガバナンス
- v DevOps/AIOpsによる運用自動化と、本番移行・運用引継ぎまでの遂行

そうして再配置されたスキル要件群は最終的に、8つの領域(ドメイン)を擁するスキル指標(BASIS)の体系として設計された。

### 3.3.2 BASIS枠組

BASISの枠組設計では、3.3.1で見た能力I～IVを5系の実務構造(i～v)で捉え直し、それを教育・評価の設計単位として扱えるよう、8つの**ドメイン(大分類)**に再編した。そして、各ドメインには、教育内容を開発工程における〈理解・判断の前提〉〈設計〉〈実装〉〈運用・改善〉の粒度で配置できるよう、4つの**サブドメイン(中分類)**を設定した。すなわち、BASISは32のサブドメインで構成される。これにより、BASISは、単に単一の技能を羅列したものでなく、開発工程において実務の意思決定がどこで発生し、何を根拠にして統制されるべきかを評価可能な形で可視化した。

こうして設計された**BASISの枠組**(ドメイン設定と要素配置)を次頁に示す。

▼ AI駆動ITスキル指標 (BASIS) の枠組

大分類 (ドメイン)	中分類 (サブドメイン)
A 生成AIリテラシー／倫理・規格	A1 モデル原理・限界・幻覚・バイアスの理解
	A2 倫理・国際規格・ガバナンスの適用
	A3 組織内AIポリシーの理解・遵守
	A4 生成品質の基本評価観点 (事実性・一貫性・再現性)
B プロンプト設計／LLM活用	B1 プロンプト設計パターン (役割付与・CoT・分割統治)
	B2 出力評価と再プロンプトによる改善ループ
	B3 LLM API／RAGの実装
	B4 コスト／レイテンシ最適化とスロットリング設計
C 要件定義／設計変換	C1 ビジネス要件のユースケース／受入基準への変換
	C2 非機能要件 (性能・保守・運用) の定義
	C3 設計案の比較評価・採択 (AI提案の選択眼)
	C4 バックログ編成とスプリント計画
D AI補完型アジャイル実装	D1 生成補完によるCRUDの実装 (Copilot等の安全活用)
	D2 レビュー・リファクタの人間主導判断
	D3 API／DB連携とエラーハンドリング設計
	D4 チーム開発の運用 (Git・PR・課題トラッキング)
E テスト自動化／品質保証	E1 テスト戦略とTDD/BDD
	E2 LLMを用いたテスト生成とフェイルファスト
	E3 静的解析・SCA・SAST/DASTの統合
	E4 品質メトリクス統合と品質ゲート運用
F セキュリティ／法規・ガバナンス	F1 生成AI特有脅威 (プロンプトインジェクション等) の対策
	F2 データ保護 (PIIを含む) と著作権・ライセンス遵守
	F3 生成物の安全性・真正性チェック (SBOM等を含む実装)
	F4 監査ログ設計と利用履歴トレーサビリティ確保
G DevOps／AIOps／データ基盤	G1 CI/CDパイプラインの設計 (VCS～自動デプロイ)
	G2 IaC・コンテナ・Kubernetesによる環境標準化
	G3 観測可能性 (ログ・トレース・APM) の実装
	G4 AIOpsによる異常検知と改善提案の運用
H 実装移行／運用引継ぎ	H1 MVP実装とユーザー検証
	H2 UAT実施と「実装可」判定取得
	H3 本番移行計画・ドキュメント・運用手順の整備
	H4 改善スプリントでのSLO/KPIの達成

以上のように、BASISの枠組は、ドメイン（大分類）において、まづ“生成AIを扱う前提能力”として **A「生成AIリテラシー／倫理・規格」**と **B「プロンプト設計／LLM活用」**を置き、“開発工程の中核能力”として **C「要件定義／設計変換」**と **D「AI補完型アジャイル実装」**と **E「テスト自動化／品質保証」**を接続させ、“横断的統制”として **F「セキュリティ／法規・ガバナンス」**を組み込み、“運用自動化と移行・引継ぎの遂行能力”として終端に **G「DevOps/AIOps／データ基盤」**と **H「実装移行／運用引継ぎ」**を配置することで、能力 **I～IV**を開発工程上の連鎖として一貫表現できる枠組にした。

これにより、受講者は“どの工程で何を判断し、何を統制するか”をドメイン単位で自己点検でき、教育提供側も到達目標と評価軸を同じ構造で設計できる。

そして、サブドメイン（中分類）を含めて見ると、**能力I**（安全・倫理・品質統制）は **ドメインA**（モデル限界理解、規格・ガバナンス適用、組織内AIポリシー遵守、生成品質評価観点）、**能力II**（要件をAIに変換する力）は **ドメインB**（設計パターン、評価と再プロンプト、LLM API/RAG、コスト最適化）と **ドメインC**（ユーザーストーリー化、非機能要件、設計案の比較採択、スプリント計画）に置き、上流工程における設計判断を構造化した。

**能力III**（短サイクル実装×品質保証）は **ドメインD**（生成補完CRUD、人間主導のレビュー・リファクタ、API/DB連携とエラーハンドリング、チーム開発運用）と **ドメインE**（TDD/BDD、LLMテスト生成、静的解析統合、品質メトリクスと品質ゲート）に分割して置き、速度向上を品質統制と不可分の要件として定義した。

**能力IV**（セキュア運用自動化と移行）は、運用面の横断的統制として **ドメインF**（プロンプトインジェクション対策、PII・著作権遵守、安全性・真正性チェック、監査ログとトレーサビリティ）を設けた上で、実装～運用を成立させる仕組として **ドメインG**（CI/CD、IaC・Kubernetes、観測可能性、AIOps運用）と **ドメインH**（MVP検証、UATと「実装可」判定、本番移行手順整備、改善スプリントでのSLO/KPI達成）に配置した。

これにより、BASISは、《本事業で養成を目指すAI補完型ITエンジニアのスキル要件を定義し、その修得度を評価するための指標》として、教育目標と評価基準を一貫させる枠組になった。

また、BASISは、既存の公的なスキル標準との連続性と生成AI時代の新規要件の両立を図る観点から、IPA（独立行政法人情報処理推進機構）の **iCD・DSS・ITSS**等との比較対照（適合種・対応度・優先度）、そして国際的枠組（SFIA、e-CF等）も視野に入れた整合性確認を行った。次節（3.3.3）で詳説する。

尚、BASISは、本年度（令和7年度）事業では“設計”が取組の中心で、スキル項目（小分類）まで記述する“開発”や精緻化、そして評価・運用との統合は次年度以降に展開する計画である。そのため、本章で扱うBASISは、ドメイン-サブドメイン構造を設定・確定し、教育目標と評価の接続が可能になる骨格（設計書）を整える段階に位置付けられる。

### 3.3.3 BASISと既存標準の対照評価

BASIS（AI駆動ITスキル指標）は、3.3.1-3.3.2で見たように、本事業で開発する教育プログラムで養成するAI補完型ITエンジニアのスキル要件を、教育目標と評価項目に接続できる形で定義するための指標である。すなわち、教育プログラムの到達目標をドメイン構造として可視化し、評価結果をドメイン別バッジとして発行することを前提にした、教育と評価を同一の枠組で扱うための参照軸である。

BASISの枠組（3.3.2）は、その設計過程（3.3.1）で見たように、養成する人材像の定義からそのスキル要件の分類・配置まで、開発工程の実務構造と照合しながら、統合・分解・再編を幾度も繰り返して叩き上げられた指標体系である。

しかし、その設計過程がいかに精緻であっても、それが本事業の論理のみで閉じて完結したものであれば、BASISの枠組は外界には恣意的な独自定義としてしか映らない（客観性を欠くとの批判を免れない）。特にBASISは教育プログラムの仕様や評価に直結する中核要素であるため、外部展開・産学連携・継続改善を視野に入れた時、その可搬性や汎用性と共に、対外的な説明可能性が担保されていなければならない。

そこで、本項（3.3.3）では、BASISの枠組を既存の（権威性が高い）スキル標準に照合してその位置付けを確認し、対外的に通用する説明言語を確保する。

具体的には、独立行政法人情報処理推進機構（IPA：Information-technology Promotion Agency, Japan）が整備した【iCD】【DSS】【ITSS】の3標準（後段で詳述）を参照枠にし、BASISのドメイン・サブドメインについて、概念上の対応関係様相（適合種）、標準側での説明範囲（対応度）、本事業上の必須度（優先度）を整理した。

これにより、BASISが既存標準の汎用的ITスキルに根差す部分と、生成AI時代の実務で新たに前景化した論点として明示すべき部分を峻別し、今後の評価設計や対外的な説明可能性を高める。

尚、「AIはツール、評価と最適化は人」という役割意識の下で“要件定義→実装→運用自動化”を自走可能にするという観点からは、各項目の〈優先度〉判断においても、“品質・セキュリティ・運用に繋がる事故を未然に防ぐ基礎技能か否か”という判定軸を与える。

#### 参照枠にしたスキル標準——【iCD】【DSS】【ITSS】

本事業でBASISの参照枠にした（権威性が高い）スキル標準は、いずれもIPAが整備した次の3標準である。

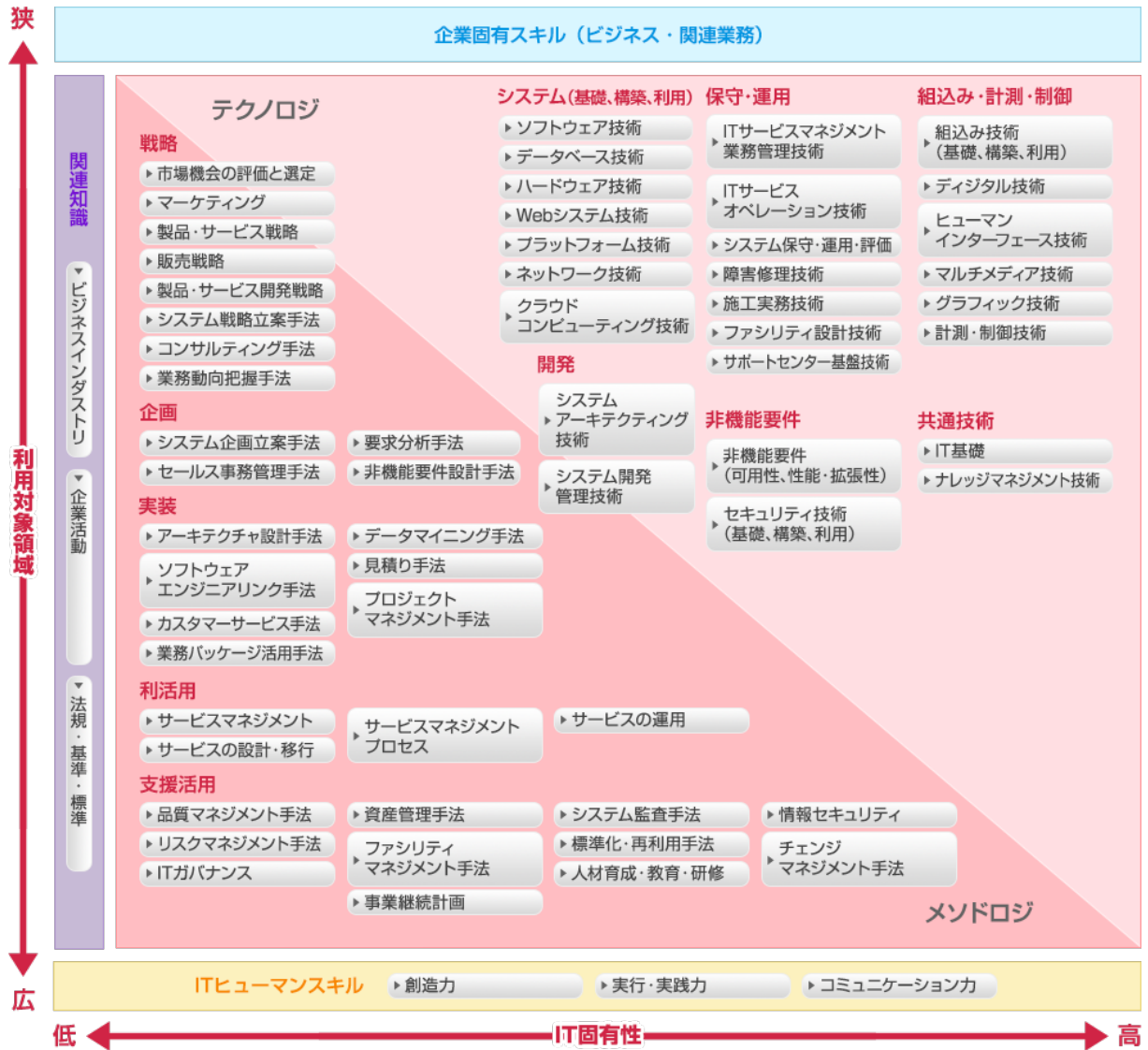
- ① i コンピテンシディクショナリ（iCD：i Competency Dictionary）
- ② デジタルスキル標準（DSS：Digital Skill Standards）
- ③ ITスキル標準（ITSS：IT skill standard）

この3標準の概要を各々、次頁以降に示す。

① 【iCD】 i コンピテンシーディクショナリ (2022)

組織でITを活用するビジネスに求められる業務（タスク）とそれを支えるIT人材の能力や素養（スキル）を「タスクディクショナリ」「スキルディクショナリ」として体系化したもので、組織は経営戦略等の目的に応じた人材育成に利用できる。

本対照では、BASISの各項目を例えば「(企画) 要求分析手法」「(システム) ソフトウェアの基礎技術」「(保守・運用) システム保守・運用・評価」「(支援活動) 品質マネジメント手法」等に対応付けた。



iCD 体系 (概要)

② 【DSS】 デジタルスキル標準 (2024)

企業等のDX推進に必要な人材のスキル（知識・技能）を体系化して定義した標準で、全従業員が習得すべき「DXリテラシー標準」（DSS-L）と、DXを推進する専門人材に求められる「DX推進スキル標準」（DSS-P）の2部構成である。カテゴリには「ビジネス変革」「データ活用」「テクノロジー」「セキュリティ」等があり、BASISとの対照では主に「テクノロジー：ソフトウェア開発」「セキュリティ：セキュリティ技術／セキュリティマネジメント」等のスキル項目を参照した。

カテゴリ	サブカテゴリ	スキル項目
ビジネス変革	戦略・マネジメント・システム	- ビジネス戦略策定・実行 - プロダクトマネジメント - 変革マネジメント - システムズエンジニアリング - エンタープライズアーキテクチャ - プロジェクトマネジメント
	ビジネスモデル・プロセス	- ビジネス調査 - ビジネスモデル設計 - ビジネスアナリシス - 検証（ビジネス視点） - マーケティング - ブランディング
	デザイン	- 顧客・ユーザー理解 - 価値発見・定義 - 設計 - 検証（顧客・ユーザー視点） - その他デザイン技術
データ活用	データ・AIの戦略的活用	- データ理解・活用 - データ・AI活用戦略 - データ・AI活用業務の設計・事業実装・評価
	AI・データサイエンス	- 数理統計・多変量解析・データ可視化 - 機械学習・深層学習
	データエンジニアリング	- データ活用基盤設計 - データ活用基盤実装・運用
テクノロジー	ソフトウェア開発	- コンピュータサイエンス - チーム開発 - ソフトウェア設計手法 - ソフトウェア開発プロセス - Webアプリケーション基本技術 - フロントエンドシステム開発 - バックエンドシステム開発 - クラウドインフラ活用 - SREプロセス - サービス活用
	デジタルテクノロジー	- フィジカルコンピューティング - その他先端技術 - テクノロジートレンド
セキュリティ	セキュリティマネジメント	- セキュリティ体制構築・運営 - セキュリティマネジメント - インシデント対応と事業継続 - プライバシー保護
	セキュリティ技術	- セキュア設計・開発・構築 - セキュリティ運用・保守・監視
パーソナルスキル	ヒューマンスキル	- リーダーシップ - コラボレーション
	コンセプチュアルスキル	- ゴール設定 - 創造的な問題解決 - 批判的思考 - 適応力

### DSS カテゴリ・サブカテゴリ・スキル項目（一部）

#### ③ 【ITSS】 ITスキル標準（2011）

個人のIT関連能力を職種毎や分野毎に体系化し、IT人材に必要なスキルやキャリア（職業）を示した指標で、ITサービスの分野11職種38分野毎に7段階のスキルレベルを設定し、各レベルで要求される業務経験や実務能力・知識を定義している。本対照では、「ITアーキテクト：設計技法／アーキテクチャ設計」

「ソフトウェア開発プロセス：分析・要求定義／開発方式設計／ソフトウェアエンジニアリング」  
「ITサービスマネジメント：ITサービスオペレーション／サービスサポート／セキュリティ管理」  
「ITスペシャリスト：セキュリティシステム構築／プラットフォームシステム構築」等を参照した。

職 種	スキルカテゴリ	スキル項目
ITアーキテクト	テクノロジー	テクノロジー
	ビジネス/インダストリ	インダストリ (ビジネス)
	メソドロジ	アプリケーションアーキテクチャ設計/アーキテクチャ設計/インテグレーションアーキテクチャ設計/インフラストラクチャアーキテクチャ設計/コンサルティング技法の活用/標準化と再利用/知的資産管理 (Knowledge Management) 活用/設計技法
ITサービス マネジメント	テクノロジー	ITサービスオペレーション/ITサービスマネジメント業務管理/アプリケーション管理/サポートセンタのインフラに関する知識/システムプラットフォーム管理/セキュリティ管理/テクノロジー/データベース管理/ナレッジマネジメント/ネットワークオペレーション/ネットワーク管理/プラットフォームオペレーション/個別業務
	パーソナル	顧客サポートスキル/顧客対応
	ビジネス/インダストリ	関連知識
ITスペシャリスト	メソドロジ	ITサービス管理/サービスサポート/サービスデスクの管理指標/サービスデリバリ/スタッフイング/ビジネスマネジメント/ファシリティマネジメント/情報資産管理/要員管理
	テクノロジー	アプリケーション共通基盤システム構築/アプリケーション共通基盤要素関連技術/システム管理システム構築/システム管理製品利用技術/システム管理要素関連技術/セキュリティシステム構築/セキュリティ要素関連技術/ソフトウェアエンジニアリング/テクノロジー/データベースサブシステム構築/データベースシステム構築/データベース製品利用技術/データベース要素関連技術/ネットワークシステム構築/ネットワーク製品利用技術/ネットワーク要素関連技術/プラットフォームシステム構築/プラットフォーム製品利用技術/プラットフォーム要素関連技術
	ビジネス/インダストリ	セキュリティ要素関連技術/テクノロジー/業務分析/知的資産管理 (Knowledge Management) 活用
アプリケーション スペシャリスト	メソドロジ	コンサルティング技法の活用/システム管理要素関連技術/セキュリティ要素関連技術/テクノロジー
	テクノロジー	テクノロジー
	ビジネス/インダストリ	業務システム構築 業務パッケージ適用 業務分析
エデュケーション	メソドロジ	コンサルティング技法の活用/ソフトウェアエンジニアリング/デザイン/業務分析/知的資産管理 (Knowledge Management) 活用
	テクノロジー	研修業界動向の把握
	メソドロジ	コースの開発/分析、設計、管理/担当専門分野における専門性/教授法
カスタマサービス	テクノロジー	ソフトウェア技術/テクノロジー/ハードウェア技術
	ビジネス/インダストリ	ファシリティ関連の法規・基準
	メソドロジ	カスタマサービスマネジメント/システム保守管理/ソフトウェア予防保守/ソフトウェア導入・配信/ハードウェア予防保守/ハードウェア導入/ファシリティ・マーケティング/ファシリティ保守運用/ファシリティ設計/一次対応・二次対応/施工実務/施工管理/障害修理
コンサルタント	テクノロジー	「経営におけるIT」の知見からの課題発見 (知見活用)
	ビジネス/インダストリ	インダストリの知見の活用/ビジネスファクションの知見の活用/ビジネス上の課題の特定と分析 (AS-IS分析) / 知的資産管理 (Knowledge Management) 活用/解決の方向性の提言 (TO-BE提言)
	メソドロジ	コンサルティングメソドロジの活用
セールス	テクノロジー	ITソリューション提案/特定製品・サービステクノロジー
	ビジネス/インダストリ	ITソリューション提案/顧客環境分析
	メソドロジ	ITソリューション提案/セールスメディアの活用/セールス事務管理/ビジネス戦略/顧客リレーションシップ構築/顧客満足度管理/顧客環境分析
ソフトウェア 開発プロセス	テクノロジー	テクノロジー
	メソドロジ	ソフトウェアエンジニアリング/デザイン/ハードウェア適合設計/プラットフォーム非依存設計/分析・要求定義/業務適用設計/開発方式設計
	テクノロジー	ITソリューション設計・開発管理/情報システム管理/新規ソフトウェア開発・既存ソフトウェア改良/業務分析/通信環境設計・運用管理
プロジェクト マネジメント	ビジネス/インダストリ	ITソリューション設計・開発管理 業務分析
	プロジェクトマネジメント	プロジェクト・コスト・マネジメント/プロジェクト・コミュニケーション・マネジメント/プロジェクト・スコープ・マネジメント/プロジェクト・タイム・マネジメント/プロジェクト・リスク・マネジメント/プロジェクト人的資源マネジメント/プロジェクト品質マネジメント/プロジェクト統合マネジメント/プロジェクト調達マネジメント
	メソドロジ	ITソリューション設計・開発管理/コンサルティング技法の活用/情報システム管理/新規ソフトウェア開発・既存ソフトウェア改良/業務分析/知的資産管理 (Knowledge Management) 活用
マーケティング	ビジネス/インダストリ	市場機会の評価と選定
	メソドロジ	コンプライアンス/マーケットコミュニケーション戦略/マーケティング戦略/マーケティング環境分析/マーケティング統括/市場機会の評価と選定/販売チャネル戦略
全職種共通	パーソナル	コミュニケーション/ネゴシエーション/リーダーシップ
	プロジェクトマネジメント	プロジェクトマネジメント

### ITSS 職種・スキルカテゴリ・スキル項目 (一部)

以上のように、【ICD】は組織のITビジネスにおける業務・活動 (実務の局面) で求められる能力・素養から、

【DSS】は企業等のDX推進に求められるDX人材のデジタルスキルから、【ITSS】は職種（役割）毎・分野毎のIT関連スキルから技能を整理する。整理軸が各々異なるからこそ、BASISの各項目にこの3標準を並置して対照することで、“本事業独自の枠組”を既存標準の交差点に据え直し、その結果としてBASISの可搬性・汎用性・客観性を高めた。

## 対照評価の観点——適合種・対応度・優先度

本対照では、BASISの項目と3標準の項目の関係を次の3軸で評価した。

① 適合種： そのBASIS項目と標準項目がいかなる関係にあるかを示す。

（“近い／遠い”だけでなく“含む／含まれない”も考慮）

評価	評価意味
=	「同等」。ほぼ同じ意味で置き換えられる。説明・運用上のズレが小さい。
c	「BASIS c 標準」。BASISが示す内容は標準の内容に含まれる。ただし、標準の方が範囲が広い。
⊃	「BASIS ⊃ 標準」。BASISが示す内容は標準の内容を含む。つまり、BASISの方が範囲が広い。
△	「部分重複」。共通部分はあるが、どちらにも固有内容があり、包含関係が成立しない。「近いが同じでない」。
×	「非該当」。実務的に対応付ける意味がない。
—	「範囲外」。関連はあり得るが、本事業の対象（地方中小IT企業のエンジニア）外のため、対応付けを扱わない。

② 対応度： その標準項目がBASIS項目をどの程度カバーできるかを示す。

評価	評価意味
A	「ほぼ全面对応」。ほぼそのまま説明できる。差は例示や用語程度である。
B	「大部分対応」。核は一致するが、重要な下位要素が一部弱い。補足が必要である。
C	「部分対応」。共通核はあるが、欠けが複数ある。標準だけでは説明が足りない。
D	「周辺のみ対応」。交点はあるが、中心が異なる。参照としては有用であるが、主対応としては弱い。

③ 優先度： そのBASIS項目が対象人材にとってどの程度に必須か（学ぶ重み）を示す。

評価	評価意味
高	「ほぼ全ての現場で必要で、欠けると事故（品質・セキュリティ・運用）に繋がり易い」。基礎スキル。
中	「有用であるが、案件や業種で変動する。必修でなく選択・発展に回す余地がある」。
低	「特定場面に限られる。参照として扱う」。

その際、BASISは「AIはツール、評価と最適化は人」という役割意識を前提にした“要件定義→実装→運用自動化”の自走を射程に置くため、〈優先度〉の評価は、単なる知識の有無でなく、実務事故の回避と自律遂行に直結するかどうか（現場で“欠けると破綻する”か）を重く見ている

**BASIS × iCD/DSS/ITSS 対照評価**

以上を踏まえ、BASISのサブドメイン（中分類）32項（A1~H4）について、【iCD】【DSS】【ITSS】の該当項目と対照し、それを ① 適合種・② 対応度・③ 優先度の観点で評価した結果を以下に示す。

大分類	BASIS		iCD			DSS			ITSS					
	中分類	スキル分類	適合種	対応度	優先度	カテゴリ・サブカテゴリ・スキル項目	適合種	対応度	優先度	職種・スキル項目	適合種	対応度	優先度	
A 生成AIリテラシー/ 倫理・規格	A1	モデル原理・境界・幻覚・バイアスの理解	(システム) ソフトウェアの基礎技術	△	D	中	テクノロジー: デジタルテクノロジー: テクノロジートレンド	△	C	中	ソフトウェア開発: 分析・要求定義	△	D	中
	A2	倫理・国際規格・ガバナンスの適用	(関連知識) 法規・基準・標準	△	C	高	セキュリティ: セキュリティマネジメント: プライバシー保護	△	C	高	ソフトウェア開発: 分析・要求定義	△	C	高
	A3	組織内AIポリシーの理解・遵守	(支援活動) ITガバナンス	c	C	高	セキュリティ: セキュリティマネジメント: セキュリティマネジメント	△	C	高	ITアーキテクト: 標準化と再利用	△	C	高
	A4	生成品質の基本評価観点 (事実性・一貫性・再現性)	(支援活動) 品質マネジメント手法	c	B	高	テクノロジー: ソフトウェア開発: ソフトウェア開発プロセス	△	C	高	ソフトウェア開発: 開発方式設計	c	B	高
B プロンプト設計/ LLM活用	B1	フロント設計パターン (役割付与・CoT・分割統治)	(企画) 要求分析手法	△	C	高	テクノロジー: ソフトウェア開発: ソフトウェア設計手法	△	C	高	ITアーキテクト: 設計技法	△	C	高
	B2	出力評価と再プロンプトによる改善ループ	(支援活動) 品質マネジメント手法	c	B	高	テクノロジー: ソフトウェア開発: ソフトウェア開発プロセス	c	B	高	ソフトウェア開発: 開発方式設計	△	C	高
	B3	LLM API/RAGの実装	(システム) Webシステムの構築技術	c	B	高	テクノロジー: ソフトウェア開発: バックエンドシステム開発	△	C	高	ITアーキテクト: アーキテクトチャ設計	△	C	高
	B4	コスト/レイテンシ最適化とスロットリング設計	(非機能要件) 非機能要件 (可用性・性能・拡張性)	c	B	高	テクノロジー: ソフトウェア開発: SREプロセス	△	C	高	ITアーキテクト: 設計技法	△	C	高
C 要件定義/ 設計変換	C1	ビジネス要件のユーザーストーリー/受入基準への変換	(企画) 要求分析手法	c	B	高	テクノロジー: ソフトウェア開発: ソフトウェア開発プロセス	△	C	高	ソフトウェア開発: 分析・要求定義	c	B	高
	C2	非機能要件 (性能・保守・運用) の定義	(企画) 非機能要件設計手法	c	B	高	テクノロジー: ソフトウェア開発: SREプロセス	△	C	高	ITアーキテクト: 設計技法	c	B	高
	C3	設計案の比較評価・採択 (AI提案の選択)	(実装) アーキテクトチャ設計手法	△	C	高	テクノロジー: ソフトウェア開発: ソフトウェア設計手法	△	C	高	ITアーキテクト: アーキテクトチャ設計	△	C	高
	C4	バックログ編成とスプリント計画	(実装) プロジェクトマネジメント手法	△	C	中	テクノロジー: ソフトウェア開発: チーム開発	△	C	中	ソフトウェア開発: 開発方式設計	△	C	中
D AI補完型 アジャイル実装	D1	生成補完によるCRUDの実装 (Copilot等の安全活用)	(システム) ソフトウェアの構築技術	c	B	高	テクノロジー: ソフトウェア開発: バックエンドシステム開発	c	B	高	ソフトウェア開発: ソフトウェアエンジニアリング	c	B	高
	D2	レビュー・リファクタの人間主導判断	(実装) ソフトウェアエンジニアリング手法	c	B	高	テクノロジー: ソフトウェア開発: ソフトウェア開発プロセス	c	B	高	ソフトウェア開発: ソフトウェアエンジニアリング	c	B	高
	D3	API/DB連携とエラーハンドリング設計	(システム) ソフトウェアの構築技術	c	B	高	テクノロジー: ソフトウェア開発: Webアプリケーション基本技術	c	B	高	ソフトウェア開発: ソフトウェアエンジニアリング	c	B	高
	D4	チーム開発の運用 (Git・PR・課題トラッキング)	(支援活動) 標準化・再利用手法	△	C	高	テクノロジー: ソフトウェア開発: チーム開発	c	B	高	ソフトウェア開発: ソフトウェアエンジニアリング	c	B	高
E テスト自動化/ 品質保証	E1	テスト戦略とTDD/BDD	(支援活動) 品質マネジメント手法	c	B	高	テクノロジー: ソフトウェア開発: ソフトウェア開発プロセス	c	B	高	ソフトウェア開発: 開発方式設計	c	B	高
	E2	LLMを用いたテスト生成とフェイルファスト	(支援活動) 品質マネジメント手法	△	C	高	テクノロジー: ソフトウェア開発: ソフトウェア開発プロセス	△	C	高	ソフトウェア開発: ソフトウェアエンジニアリング	c	B	高
	E3	静的解析・SCA・SAST/DASTの統合	(支援活動) 品質マネジメント手法	△	C	高	セキュリティ: セキュリティ技術: セキュア設計・開発・構築	c	B	高	ITスペシャリスト: セキュリティシステム構築	c	B	高
	E4	品質メトリクス統合と品質ゲート運用	(支援活動) 品質マネジメント手法	c	B	高	テクノロジー: ソフトウェア開発: ソフトウェア開発プロセス	△	C	高	ソフトウェア開発: 開発方式設計	△	C	高
F セキュリティ/ 法規・ガバナンス	F1	生成AI特有脅威 (プロンプトインジェクション等) の対策	(非機能要件) セキュリティの構築技術	△	C	高	セキュリティ: セキュリティ技術: セキュア設計・開発・構築	△	C	高	ITスペシャリスト: セキュリティ要素関連技術	△	C	高
	F2	データ保護 (PIIを含む) と著作権・ライセンス遵守	(関連知識) 法規・基準・標準	△	C	高	セキュリティ: セキュリティマネジメント: プライバシー保護	△	C	高	ソフトウェア開発: 分析・要求定義	△	C	高
	F3	生成物の安全性・真正性チェック (SBOM等を含む実装)	(非機能要件) セキュリティの構築技術	△	D	高	セキュリティ: セキュリティ技術: セキュア設計・開発・構築	△	C	高	ITスペシャリスト: セキュリティシステム構築	△	C	高
	F4	監査ログ設計と利用履歴トレーサビリティ確保	(支援活動) システム監査手法	c	B	高	セキュリティ: セキュリティマネジメント: セキュリティ体制構築・運営	△	C	高	ITサービスマネジメント: セキュリティ管理	c	B	高
G DevOps/AIOps/ データ基礎	G1	CI/CDパイプラインの設計 (VCS-自動デプロイ)	(開発) システム開発管理技術	c	B	高	テクノロジー: ソフトウェア開発: SREプロセス	c	B	高	ソフトウェア開発: ソフトウェアエンジニアリング	c	B	高
	G2	IaC・コンテナ・Kubernetesによる環境標準化	(システム) プラットフォームの構築技術	c	C	高	テクノロジー: ソフトウェア開発: クラウドインフラ活用	c	B	高	ITスペシャリスト: プラットフォームシステム構築	c	B	高
	G3	観測可能性 (ログ・トレース・APM) の実装	(保守・運用) システム保守・運用・評価	c	B	高	テクノロジー: ソフトウェア開発: SREプロセス	c	B	高	ITサービスマネジメント: ITサービスオペレーション	c	B	高
	G4	AIOpsによる異常検知と改善提案の運用	(保守・運用) システム保守・運用・評価	△	C	中	テクノロジー: ソフトウェア開発: SREプロセス	△	C	中	ITサービスマネジメント: ITサービスマネジメント業務管理	△	C	中
H 実装移行/ 運用引継ぎ	H1	MVP実装とユーザー検証	(利活用) サービスの設計・移行	c	B	高	テクノロジー: ソフトウェア開発: ソフトウェア開発プロセス	△	C	高	ITサービスマネジメント: サービスデリバリー	△	C	高
	H2	UAT実施と「実装可」判定取得	(利活用) サービスの設計・移行	△	C	高	テクノロジー: ソフトウェア開発: ソフトウェア開発プロセス	△	C	高	ITサービスマネジメント: サービスデリバリー	△	C	高
	H3	本番移行計画・ドキュメント・運用手順の整備	(利活用) サービスの設計・移行	c	B	高	テクノロジー: ソフトウェア開発: SREプロセス	△	C	高	ITサービスマネジメント: サービスサポート	c	B	高
	H4	改善スプリントでのSLO/KPIの達成	(保守・運用) システム保守・運用・評価	c	B	高	テクノロジー: ソフトウェア開発: SREプロセス	c	B	高	ITスペシャリスト: システム管理要素関連技術	c	B	高

以上の対照結果から次のようなことが判った。

● BASISの中核は、既存標準の“ソフトウェア開発～運用”を確実に根を持つ

BASISのドメインC（要件定義／基本設計）・D（実装）・E（テスト／品質保証）・G（DevOps／AIOps）・H（移行／引継ぎ）の多くが、【iCD】【DSS】【ITSS】のいずれか、または複数の標準において「**c**／**B**／**高**」を中心にして対応付けられている。これは、BASISが“生成AI固有の技能を集めた奇矯な枠組”でなく、既存の汎用ITスキル（ソフトウェア工学・SRE・サービスマネジメント等）を基盤にし、その遂行速度と精度をAIで引き上げるという観点で設計されていることを、既存標準側から裏付ける結果である。

すなわち、BASISの“中心線”は、要件定義・設計・実装・テスト・デプロイ・運用という従来の開発ライフサイクルに基づいて、生成AIはそれを置換するのではなく、補完して高速化するための“道具”として位置付けられる。この構造自体が、本事業が目標とする「0.5～1年で“要件定義→実装→運用自動化”を自走」という到達像に整合的である。

● A（生成AIリテラシー）と B（プロンプト設計）は、既存標準に「部分重複」として顕れる

BASISのドメインA（生成AIリテラシー／倫理・規格）と B（プロンプトエンジニアリング）は、〈適合種〉が「△」（部分重複）になる項目が多く、〈対応度〉も「**C**」（部分対応）や「**D**」（周辺のみ対応）が目立つ。これは、既存標準の多くが**A1**（AIモデル固有の限界（幻覚・バイアス））・**B1**（プロンプト設計パターン）・**B2**（出力評価と再プロンプトによる改善ループ）等の生成AI特有の実務を、（少なくとも現行の分類上は）専用の中心項目として持ち難いことを示す。そのため、【DSS】では「テクノロジートレンド」や「ソフトウェア開発プロセス」、【iCD】では「要求分析」や「品質マネジメント」、【ITSS】では「分析・要求定義」や「設計技法」に“寄せて”説明する構図になる。

しかし、ここで重要なことは、“一致しにくい＝不要”でないということである。**A2**（倫理・国際規格・ガバナンス）や**A4**（生成品質評価）は〈優先度〉が「**高**」で、**B2**（出力評価と再プロンプト）も「**高**」である。すなわち、既存標準上は“周辺”に見えても、生成AIを実務導入した途端に品質事故・セキュリティ事故・運用破綻の起点になり得る領域として、BASISでは基礎スキル「**高**」に格上げして扱う必要があるという判断が本対照に顕れている。

● F（セキュリティ）は、既存標準に接続しつつ、生成AI特有の“追加説明”を要する

BASISのドメインF（セキュリティ／ガバナンス）は、**F4**（監査ログ設計・トレーサビリティ）が【iCD】と【ITSS】の双方で「**c**／**B**／**高**」に位置付けられる一方で、**F1**（プロンプトインジェクション／データ漏洩）や**F3**（SBOM等を含む真正性チェック）は「△／**C**～**D**／**高**」が中心になる。これは、監査・管理・運用等の“枠組”は既に既存標準に厚みがある（そのため「**c**」に整理できる）が、生成AIの入力面（プロンプト）や生成物のサプライチェーン面（SBOM等）に固有の具体論は、既存標準上では一般的なセキュア設計・開発・構築の一部として扱われ易く、BASISではそれを“生成AI前提で具体化した項目”として再定義している。そのため、本事業では、既存標準で説明可能な骨格（監査・統制・セキュア開発）を踏まえつつ、生成AI固有の攻撃ベクトル・漏洩パターン・真正性確保を教育・評価の観点に落とし込む必要がある。

### ● 3標準の“軸の違い”が、BASISの外部展開性をむしろ補強する

【iCD】【DSS】【ITSS】は同じ“スキル標準”であっても整理軸（業務局面／DX推進／職種）が異なるため、対照評価の結果も一律でない。例えば、G2（環境標準化）は【iCD】では「**c**／**C**」でやや薄い一方で、【DSS】【ITSS】では「**c**／**B**」で厚く出る。これは、当該技能を“業務スキル”として説明するか、“DXスキル”として説明するか、“職種スキル”として説明するかで、既存標準の“見え方”が変わるためである。

しかし、この“揺らぎ”は欠点でない。むしろ、BASISが、i) 組織内の人材育成・配置（iCD）、ii) DX文脈での対外説明（DSS）、iii) 職種（役割）の言語での合意形成（ITSS）という複数の“外部言語”に翻訳可能であることを意味する。すなわち、BASISは、本事業内の教育設計のためだけでなく、地方中小企業とのコミュニケーションや修了者のスキル可視化（対外提示）にも耐える枠組として、実務的な可搬性を得ている。

\*

本節（3.3）では、AI補完型ITエンジニアの到達像を、“要件定義→実装→運用自動化”を自走し得る人材と捉え直し、そのために必要な汎用ITスキルを“AIで高速化・高精度化する”という観点から、BASIS（AI駆動ITスキル指標）を設計した。BASISは、教育プログラムの到達目標を可視化し、評価結果をバッジとして提示するための参照軸であり、教育と評価を循環させる本事業全体の中核に位置付けられるものである。

そして、BASISを32サブドメイン（4サブドメイン×8ドメイン）に整理し、生成AI活用の局面を開発工程に基づいて配置することで、単なる知識の列挙でなく、実務遂行の判断力（「AIはツール、評価と最適化は人」）を含む体系として構造化した。

最後に、3.3.3では、【iCD】【DSS】【ITSS】との対照評価により、BASISの中心が既存標準の汎用ITスキルに確実に根差していること、それと同時に生成AI特有の論点（幻覚・バイアス、プロンプト、生成物の真正性等）が既存標準上は「部分重複」として現れ易く、そうであればこそ本事業では〈優先度〉が高い基礎項目として明示的に立てる必要があることが判った。

いづれにせよ、BASISは、“本事業内で閉じた独自枠”でなく、既存標準の交差点に位置付けられる形で設計されている。そのため、今後のカリキュラム具体化・評価設計・産学連携においても、BASISは対外的に通用する説明言語を保持したまま、地方中小企業の実務課題に即して深化・改善してゆくことが可能である。

## 3.4 教育体系——教育プログラム

前節(3.3)では、AI補完型ITエンジニアに必要なスキル群をAI駆動ITスキル指標(BASIS)の枠組として整理し、教育と評価を接続できる参照軸として骨格化した。その際、本年度(令和7年度)事業では“設計”を取組の中心に据え、スキル項目(小分類)の整備・精緻化や評価・運用との統合は次年度(令和8年度)事業で展開する。そのため、本年度に策定したのは“何を到達点に措定し、どの構造で可視化・評価するか”という枠組であり、教育モデルを実装可能なものとして成立させるためには、この参照軸を“教育の順序・量・型”に翻訳し、授業設計として確定させる工程が不可欠である。

そこで、本節(3.4)では、教育プログラムの体系としてカリキュラムとシラバスを整備する。すなわち、BASISが“測るべき到達点”を与えるのに対し、カリキュラムは“到達のための全体設計”を、シラバスは“各コマの実行計画”を与える役割を担う。これらは、後続の教育コンテンツ開発および教育基盤・評価システムの構築に先行して確定されるべき、教育設計の中核仕様である。

具体的には、教育プログラム全体を見通した計60コマのカリキュラムを提示し、各コマを教育形態(講義・演習・実習・PBL・自習)としても把握できるように構造化することで、教育活動の配列と配分を可視化する。シラバスは、教育プログラムの設計思想を担保しつつ、授業運用と評価連動に整合する記述単位としてフォーム(項立て・記述規約)を定めた上で、本年度事業の先行開発範囲である【科目1】【科目2前半】(12コマ)について、各コマのシラバスを整備する。

尚、各コマで扱う映像・教材・課題等の教育素材は、次節(3.5)で「教育コンテンツ」として詳述する。

### 3.4.1 設計方針——BASISからカリキュラムへ

前節(3.3)で整理したBASIS(AI駆動ITスキル指標)は、AI補完型ITエンジニアに求められるスキル群を教育の到達水準として可視化するための枠組である。ただし、その指標がいかに精緻であっても、それ自体は“測るための座標系”でしかなく、受講者を現実的に到達点に導くためには、その座標系を教育の“実装仕様”に翻訳する工程が不可欠である。すなわち、BASISで示された到達点を“教育の順序(何から何へ)・量(どの程度)・型(どう学ばせるか)”に変換し、授業設計として確定させる必要がある。

そこで、本項では、教育プログラムの体系として、まづ**(全体)カリキュラム**を定め、その次に**シラバス**を整備するという設計方針を明確にする。

本事業で構築を企図する教育モデルは、3.2で見たように、BASISに基づいて“AI×DevOps”の教育内容を体系化した上で、(GPT-TA評価システムを駆使した)“測定可能な循環型教育”と“地方企業課題解決PBL”を融合し、人口減少地域でも企業DXを推進・持続できる職業人材を0.5~1年間で養成するという基本構造を持つ。

そのため、教育体系の設計では、単に科目を並べるのではなく、i)到達点(BASIS)と教育活動(授業)を対応付け、ii)教育の成果が測定され、iii)測定結果が次の指導・次の改善に循環し得るよう、授業単位の計画・仕様(シラバス)まで一貫して定義しておくことが肝要である。これは、後続の教育素材(コンテンツ)や

教育基盤（プラットフォーム・評価システム）を、指導者の属人的な工夫でなく、再現可能な“設計された教育”として成立させるための前提条件でもある。

この変換（指標→授業設計）を行う際、本事業では、教育体系を〈カリキュラム〉と〈シラバス〉に二層化して扱う。カリキュラムは、教育プログラム全体を俯瞰し、科目の配列と配分、そして各科目内の教育段階（講義・演習・実習）を一つの体系として定める“全体設計”である。これに対してシラバスは、各コマを実行可能な指導計画に落とし込み、受講者に求める活動、達成すべき到達点、評価・フィードバックの取り回しを、運用可能な最小単位で規定する“実装仕様”である。両者の役割差を明確にしておくことで、全体最適（体系）と局所最適（各コマ）の混同を避け、後続工程（映像・教材・課題等の作成や基盤・評価システムの構築）を同一の設計意図の下に整合的に展開できる。

教育体系の骨格は、7科目（S1～S6+PBL）から成る全体構成と、その計60コマ（講義23コマ・演習21コマ・実習16コマ）という教育量の設定である。

ここで重要なことは、各科目が単なるテーマ別の寄せ集めでなく、全科目が〈講義（L：Lecture）→演習（E：Exercise）→実習（P：Practical）〉という段階教育で構成される点である。すなわち、講義で概念・枠組・判断基準を与え、演習で手順化・技能化し、実習で現実の制約を伴う実装・運用に接続するという教育課程の基本線を、全科目に共通する設計原理として採用している。

さらに、授業外学習としての**自習**は、講義に附属して配置し、受講者が各自で取り組む前提であるため、コマ数に計上しない（ただし、設計上は“授業の外に置かれた教育活動”として、講義とセットで位置付けられる）。

また、BASISとの接続は、抽象的に“関連がある”と述べるのではなく、科目修了時に「BASIS該当ドメインのバッジLv.2以上（全4Lv.）」が取得可能になる設計思想として、教育体系に埋め込まれている。すなわち、各科目は、BASISの特定ドメイン群に対して一定の到達を齎す“まとめり”として編成され、受講者には学習の進捗と達成が可視化され、教育提供側には教育目標・評価・改善の単位が確保される。このように、カリキュラムはBASISを教育活動に変換する一次仕様であり、シラバスはその一次仕様を授業運用に変換する二次仕様という関係で整理される。

教育体系における**科目**（S：Subject）の配列は、“生成AIの基礎理解と倫理”（S1）から始め、“LLMの活用方法”（S2）に進み、“AI補完型の開発・品質・統制・運用”（S3～S6）に射程を段階的に広げ、最終的に地域企業の実課題を扱う**PBL**（Project-Based Learning「プロジェクト型学習」）で統合するという構造を採る。この配列は、個別技能の獲得に留まらず、人口減少地域でDXを推進・持続させるという本事業の課題設定に基づいて“技術理解→実装能力→品質・セキュリティ・ガバナンス→運用・改善→現場課題の解決”という実務の連鎖を教育の連鎖として再現することを意図する。

以上の方針に基づき、本事業では、教育プログラム全体を俯瞰するカリキュラム（60コマ）を開発した。次項（3.4.2）では、そのカリキュラムの構造を示し、科目配列と段階教育（L→E→P）の設計意図を明確にする。

### 3.4.2 カリキュラム—AI補完型ITエンジニア養成プログラム

本事業で開発するAI補完型ITエンジニア養成プログラムの教育体系は、7科目（S1～S6+PBL）から成る。この全体を俯瞰し、各科目を教育の“順序と量”の両面で整列させた“教育プログラムの全体設計図”がカリキュラムである。その際、ここで提示するのは、単なる科目名の列挙でなく、i) 全体としての教育配分、ii) 各科目の目的と教育段階（教育形態）、iii) 最終統合としてのPBLの位置付け という3点を一体として読める構造である。

尚、この科目配列と配分が拠って立つ教育上の前提（教育目的・ポリシー・教育目標・対象・修了判定等）は、後掲の「教育プログラム概要」に整理する。

#### ▼ AI補完型ITエンジニア養成プログラム カリキュラム（教育体系）

科目名	科目目的 [主要単元]	コマ	L	E	P
<b>[S1] 生成AIリテラシーとAI倫理</b>	生成AIの基礎構造・限界・社会的影響を理解し、その利用のリスクと影響を評価しながら適切な利用行動を判断した上で、組織内AIポリシーを理解・遵守できる素養を形成する。 [生成AI概論、AI倫理とガバナンス、モデル限界・バイアス、標準化・量子化、国際AI倫理規格、等]	6	4	2	0
<b>[S2] プロンプト設計とLLM活用</b>	LLMの内部機構とAPI実装を理解し、出力評価と再プロンプトによる改善ループを通じて高品質の出力を得るプロンプト設計とRAG（文書検索で根拠を附与して回答精度を高める手法）構築を習得する。 [LLMアーキテクチャ、プロンプト設計手法、LLM API実装、RAG応用、マルチモーダルプロンプト、生成品質評価、等]	12	6	4	2
<b>[S3] AI補完型アジャイル開発</b>	Copilot等の生成AIツールを組み込んだアジャイル開発手法を学び、受入基準に基づいて短サイクルで機能実装・リリースできるチーム開発力を涵養する。 [アジャイル概論、バックログ管理、Copilot CRUD、レビュー・リファクタ、リリース計画・KPI、等]	9	3	4	2
<b>[S4] テスト自動化と品質保証</b>	Mutation TestやLLMを用いたテスト生成で品質を測定・維持する技術を学び、CIと品質ゲートを設計できる力を涵養する。 [テスト自動化戦略、カバレッジ最適化・TDD、品質メトリクス統合、品質ゲートライン設計、等]	8	3	3	2
<b>[S5] セキュリティとガバナンス</b>	生成AI特有の脅威を理解し、安全な利用ポリシーと（データ保護および著作権・ライセンス遵守を含む）法的コンプライアンス対応策を実践できる力を涵養する。 [生成AIセキュリティ脅威、プロンプトインジェクション対策、法規制・コンプライアンス、等]	4	2	1	1
<b>[S6] DevOps/AIOps/データ基盤</b>	IaC・Kubernetes・AIOpsを組み合わせ、CI/CDによるデプロイと運用自動化を実装できるプラットフォームエンジニアリングを習得する。 [DevOps文化・CI/CD、GitHub Actions 演習、IaC・K8s、AIOps・Observability、データパイプライン設計、等]	9	2	4	3
<b>[PBL] 地域DXプロジェクト</b>	地方企業のDX課題をチームで解決し、生成AIとDevOpsの実務適用経験を通じてBASISバッジ上位レベルを取得する。 [前期：要件定義→MVP、後期：改善スプリント→KPI達成・納品]	12	3	3	6
(コマ数) 合計		<b>60</b>	<b>23</b>	<b>21</b>	<b>6</b>

- ※ 各科目（S）は、講義（L）・演習（E）・実習（P）の段階教育で構成
- ※ 自習は講義に附属して配置（授業外で受講者が各自で取り組むため、コマ数に計上しない）
- ※ 科目修了時にBASIS該当ドメインのバッジLv.2以上（全4Lv.）が取得可能

上表は、本教育プログラムを“7科目・60コマ”の体系として編成する際の骨格（科目配列とコマ配分）を示したものである。これと共に、その運用設計として、本教育プログラムの目的・ポリシー・修了判定等の基本仕様を明確にするべく、**教育プログラム概要**を以下に示す。

▼ **教育プログラム（AI補完型ITエンジニア養成プログラム）概要**

名 称	<b>AI補完型ITエンジニア養成プログラム</b>	
概 要	人口減少地域でも企業DXを推進・持続させるべく、AI駆動ITスキル指標（BASIS）に基づいてAI×DevOpsの教育内容を体系化した上で、測定可能な循環型教育と地方企業課題解決PBLを融合し、AI補完型ITエンジニアを0.5～1年間で養成する専修学校発の職業教育プログラム。	
ポリシー	<ul style="list-style-type: none"> <li>○ 教育目標をBASISレベルに直結し、科目内容・評価をBASISに対応させる。</li> <li>○ 全体の2/3をハンズオン型教育（演習・実習・PBL）にして実践型教育を実現する。</li> <li>○ GPT-TAを活用したCBT解答・提出物等の採点・講評によって学修成果を評価する。</li> <li>○ GPT-TA集計データの分析によって教育プログラムの質保証と継続的改善を実施する。</li> </ul>	
教育目標	<ul style="list-style-type: none"> <li>○ 生成AIとDevOpsの一体的活用によって要件定義から運用自動化まで自律的に遂行できる能力を習得させる。</li> <li>○ 地方企業の実課題をAI駆動ITスキルによって解決し、継続的改善サイクルを組織内で主導できる人材にする。</li> </ul>	
教育対象	<ul style="list-style-type: none"> <li>○ IT専門学校（情報処理系学科）卒業生・在校生（2年生以上推奨）</li> <li>○ 地方企業所属の現役エンジニア</li> <li>○ リスキリング目的の非IT社会人 等</li> </ul>	
教育形態	講義（Lecture）	教員主導で理論・枠組等を体系的に解説する授業 [33.3%]
	自習（Self-Study）	各講義後20分で実施するCBT（正答率80%未満は自動再受験） [授業外]
	演習（Exercise）	講義内容を踏まえて教員が工程・手順を提示しながら小規模タスクを実装するラボ授業 [30.0%]
	実習（Practical Training）	受講者が半自律的にサブシステムや機能を実装して講評を受ける開発実務型授業 [16.7%]
	PBL（Project-Based Learning）	地方企業の実課題をチームで分析・設計・実装・検証して成果発表する課題解決型授業 [20.0%]
修了判定	<ul style="list-style-type: none"> <li>○ GPT-TAによるスコア算出方法と判定ロジックに基づき、全科目で65点以上（100点中）を得点</li> <li>○ BASISの全ドメインでバッジLv.1以上を取得</li> <li>○ PBL成果物に対して企業が「実装可」と承認</li> </ul>	
学修成果 （目標）	<ul style="list-style-type: none"> <li>○ BASISバッジ全ドメイン取得率90%</li> <li>○ PBL成果物 企業導入率70%</li> <li>○ 修了後1年定着率85%</li> </ul>	

## 教育プログラムの科目構成と段階設計——基礎から統合へ

科目構成は、まず【S1】「生成AIリテラシーとAI倫理」（6コマ：L4・E2・P0）により、生成AIの基礎構造・限界・社会的影響を理解し、リスク評価と倫理的利用判断の素養を形成する所から開始する。人口減少地域でのDX推進では、限られた人員でAIを扱う局面が多く、誤用や過信が直接的に業務リスクに転化し得るため、初期段階で“技術を利用する以前の判断力”を獲得させる設計である。

続く【S2】「プロンプト設計とLLM活用」（12コマ：L6・E4・P2）は、LLMの内部機構とAPI実装を理解し、プロンプト設計とRAG構築を習得する科目である。【S1】で形成した倫理・ガバナンスの観点を前提にし、生成AIを“道具として使い熟す”ための基礎技能を“講義→演習→実習”という段階を踏んで定着させる。ここで言う実習は、単なる操作練習でなく、実装・検証という工学的プロセスに接続するための足場であり、次の開発プロセス教育（【S3】以降）に橋を架ける役割を担う。

【S3】「AI補完型アジャイル開発」（9コマ：L3・E4・P2）は、Copilot等の生成AI支援ツールを組み込んだアジャイル開発手法を学び、短サイクルで機能実装・リリースできるチーム開発力を涵養する科目である。【S1】【S2】の狙いを“AIを扱う個の技能”の形成と捉えれば、【S3】はそれを“開発プロセスに組み込んだ集団技能”に引き上げる段である。生成AIは開発速度を上げるのと同時に、レビュー・リファクタ等の判断を曖昧にし得るため、講義でプロセス原理を与え、演習で手順化し、実習で実装と講評を通じて“人間主導の意思決定”を鍛えるように設計している。

【S4】「テスト自動化と品質保証」（8コマ：L3・E3・P2）は、Mutation TestやLLMを用いたテスト生成等で品質を測定・維持する技術を学び、CI/品質ゲートを設計できる力を涵養する科目である。生成AIの導入局面では“早く作れる”ことが強調されがちであるが、人口減少地域の現場では運用品質の低下が直ちに事業継続リスクになる。【S4】は、【S3】までの実装を“継続的に安全にリリースできる状態”に収束させるための品質工学を扱い、後続のセキュリティ・運用に自然に接続する。

【S5】「セキュリティとガバナンス」（4コマ：L2・E1・P1）は、生成AI特有の脅威（プロンプトインジェクション等）を理解し、安全な利用ポリシーと法的コンプライアンス対応策を実践できる力を涵養する科目である。ここでの狙いは、セキュリティや法規を“知識として知る”だけでなく、設計・実装・運用の各局面に落とし込んだ判断と手当て（ポリシー策定や対策の適用）を、演習と実習で具体化させる点にある。

【S6】「DevOps/AIOps/データ基盤」（9コマ：L2・E4・P3）は、IaC・Kubernetes・AIOpsを組み合せ、デプロイと運用自動化を実装できるプラットフォームエンジニアリングを習得する科目である。人口減少地域でDXを持続させるためには、個別システムを作るだけでなく、運用・監視・改善の仕組みを最小人数で回す必要がある。【S6】はそのための技術的基盤（CI/CD・環境標準化・観測可能性・データパイプライン等）について、講義で概念を押さえつつ、演習と実習で実装に接続する。

以上の【S1】～【S6】が“AI×DevOps”の中核技能を段階的に形成する“科目列”であるのに対し、【PBL】「地域DXプロジェクト」（12コマ：L3・E3・P6）は、地方企業のDX課題をチームで解決し、生成AIとDevOpsの実務適用経験を通じてBASISバッジ上位レベルを取得することを狙う統合科目である。その際、

前期（要件定義→MVP）と後期（改善スプリント→KPI達成・納品）という二段構えを明示している点に、本教育プログラムが“作って終り”でなく、改善と定着までを射程に入れていることが顕れている。PBLは、受講者の学習の集大成であるのと同時に、地域企業の課題そのものを教材化し得るため、人口減少地域における産業・自治体の実装課題と教育を接合する“結節点”として配置している。

### コマ配分と教育形態——教育体系としてのバランス

本教育プログラムのコマ総数は60であり、その内訳は講義23・演習21・実習16である。

講義が占める割合（38.3%）は、理論・枠組・判断基準を体系的に理解させるために必要な最小限の厚みとして確保されている一方で、演習・実習が全体の過半（61.7%）を占める。これは、AI補完型ITエンジニアに求められる能力が、知識の保持でなく、実際に手を動かして“実装し、運用し、改善する”一連の技能として発現することを前提にした配分である。

また、本教育プログラムでは、講義に附属して自習を配置するが、自習は授業外学習として各自で取り組む前提であるため、コマ数に計上しない。すなわち、カリキュラムの「60コマ」は授業配置の設計であり、授業外に置かれる自習は、講義で得た理解を短サイクルで確認・補強するための装置として位置付けられる。教育形態の定義においても、「講義＝体系的理解」「演習＝手順化・技能化」「実習＝実装・講評」という役割分担が与えられ、授業外の自習は講義後の短時間CBTとして設計されている。

このように、授業内外を含めて教育活動を設計することで、受講者の到達状況を“測定可能な循環”に接続し得る下地を整える。

科目の配分は、【S1】～【S6】が計48コマ、【PBL】が12コマである。

その際、【PBL】が全体の2割を占め、このうち実習が6コマで厚く配されている点は、このPBLが“発表会型イベント”でなく、要件定義からMVP・改善スプリント・納品までを通して“現場の制約下で成果を出す”訓練として位置付けられていることを示す。人口減少地域でDXを持続させるためには、技術の適用が単発のPoCで終らず、運用に引き継がれ、改善され、定着する必要がある。【PBL】の二段構成は、その要請を教育設計に写像したものである。

また、全科目を通じて“講義→演習→実習”という段階教育の設計を採ることで、受講者は各科目内でも“理解→手順化→適用”という同型の学習経験を反復することになる。この反復は、受講者には“学び方の型”を獲得する効果を持ち、教育提供側にはシラバス記述の規約化を容易にする。その結果として、科目横断での到達度把握や教育プログラムの改善（循環）を、個別授業の偶然の成功に依存せずに行えるようになる。

さらに、科目修了時にBASIS該当ドメインのバッジLv.2以上を取得できるという設計思想は、カリキュラムを“修了単位の集合”として閉じるのではなく、BASISを介して達成状況を可視化し、次の教育・次の課題に接続することを意図している。その意味で、カリキュラムは、教育プログラムの内容表であるの同時に、到達指標（BASIS）と評価・改善の仕組を載せるための“台座”でもある。

\*

本年度事業では、カリキュラムの確定を前提に、その【S1】（6コマ）＋【S2】前半（6コマ）について

先行的にシラバスを整備し、先行実証講座（2週間）と統合実証講座（6週間）に接続することで、体系設計の妥当性を現場で点検しながら次年度以降に拡張する、段階的な開発計画を採る。この段階化は、教育体系を“初めから完成品として作る”のではなく、測定と改善を繰り返しながら完成度を高めるという本教育モデルの基本思想にも整合している。

以上が、本教育プログラムの教育体系=カリキュラム（60コマ）の構造である。

次項（3.4.3）では、この全体設計を各コマに落とし込むべく、シラバスのフォーム（項立て・記述規約）を定め、先行開発範囲である【S1】と【S2】前半の各コマについて、授業運用と評価連動に整合する指導計画として具体化する。

### 3.4.3 シラバス——各コマ指導計画

前項（3.4.2）では、AI補完型ITエンジニア養成プログラムを“7科目・60コマ”の教育体系として示し、その科目配列とコマ配分を以て教育プログラムの全体設計とした。しかし、カリキュラムが提示するのは、あくまで“教える順序と量”である。教育モデルを実装可能な仕様として成立させ、教材・指導・評価・運用を一貫させるためには、授業の最小単位である各コマについて、指導内容・進行・評価接続を確定させねばならない。本教育プログラムにおけるシラバスは、正にこの“コマ単位の教育仕様”を記述したものであり、授業担当者の属人的判断に依存しない再現性の確保を主眼にして整備した。

また、本事業のテーマでもある生成AIは、出力の揺れや誤りを内在し、誤用がそのまま業務リスクに転化し得る技術である。そのため、どのコマで何を根拠にして〈到達〉と判定するか、また次コマに接続するために何を提出物・記録として残させるかを、指導計画の段階で明示することが不可欠である。シラバスは、BASISで定義した到達目標を、授業運用と評価連動に整合する記述単位に変換し、教育コンテンツ開発（3.5）および教育基盤構築（3.6）に橋渡しする中核仕様として位置付けられる。

#### シラバス様式（フォーム）の設計

本教育プログラムでは、まづシラバスを**全コマ共通の様式（フォーム）**に定め、記述粒度と用語を統一した。フォームは、講義・演習・実習・PBLのどの教育形態にも適用できる共通骨格を持たせつつ、教育形態毎の運用上の差異（提示すべき前提、提出物、評価観点、記録様式等）が漏れなく記載されるように構成した。

その**シラバスの要項**（主要項目と記載内容）をフォームと共に次頁に示す。

## ▼ シラバスの様式&要項

コマ名	当該コマの指導射程を1フレーズで設定する。
コマ概要	指導対象の知識・判断・操作の範囲を要約し、授業運用上の“外せない線”を提示する。
科目内での当該コマの位置付け	前後のコマとの依存関係を明示し、〈導入〉〈定着〉〈応用〉〈統合〉等の役割を設定する。 ※ 当該コマがその前後コマといかなる概念的・技能的連関を有し、科目全体の到達目標のどの部分を担うかについて、前コマからの接続と次コマへの展開を明示する。これにより、当該授業が単発の話題提示でなく、到達目標に向けて構造的に積み上げられる設計単位であることを自覚し、科目全体の論理的整合と教育責任を可視化する。
コマ形態	〈講義（オンデマンド視聴）〉〈演習（ライブ配信）〉〈実習（ライブ配信／対面リアル）〉等を明示し、その指導の型と実施条件を設定する。
コマ内容	指導者が提示する論点と手順を列举し、講義台本・教材・演習シナリオ等に直結する仕様にする。コマ内テーマを4項で設定する。
キーワード	当該コマ内容に準拠しながら科目内で共有すべき重要概念・重要用語等を（8点以上）提示する。これにより、教材における表記の揺れや理解のズレも抑える
到達目標	BASISに基づき、当該コマ終了時点で到達させる状態を行動記述（“CanDo”式）で提示する。
理解度確認テスト	自習（CBT）における出題観点（1コマ5点）を列举し、指導内容と評価の整合性を担保する。

このように、フォーム自体が〈指導内容〉〈評価観点〉〈記録・提出〉の3点を同一の枠内で結節させる構造になっている点が、本シラバスの設計の要点である。これにより、教材や課題を個別に作り込む前に、教育提供側としての意思決定（何をどう教え、何を以て〈到達〉と判定するか）をコマ単位で設定できる。

そして、次節以降で扱う教育コンテンツ（講義映像・講義資料・演習課題・自習問題：3.5）および教育基盤（教育プラットフォーム・GPT-TA評価システム：3.6）は、このフォームを参照項にして整合的に設計・開発される。

### 本年度に整備したシラバス（【S1】＋【S2】前半：計12コマ）

教育体系（カリキュラム）のうち本年度の先行開発範囲は【S1】「生成AIリテラシーとAI倫理」（講義4コマ・演習2コマ）と【S2】「プロンプト設計とLLM活用」前半（講義3コマ・演習3コマ）で、この計12コマについてシラバスを具体化した。

これらは、次年度に実施する「先行実証講座」（2週間）で実際に運用する範囲であるとの同時に、以降の科目（【S2】後半～【S6】＋PBL）に展開する際の記述規約と運用基準を確立するための先行設計でもある。

**【S1】と【S2】前半のシラバスを各々、次頁以降に示す。**

▼【S1】「生成AIリテラシーとAI倫理」シラバス（6コマ）

コマ名	<b>01 生成AIの基礎と安全利用の前提</b>
コマ概要	生成AIが何を行う技術かを整理し、入力と出力の関係、プロンプトの基本形、目的の提示の仕方等を学習する。検索との違いや限界を把握し、業務利用時の注意点や避けるべき事例を共有する。安全利用のための最小限のルールを明確にする。
科目内での当該コマの位置付け	S1全体の土台として、共通語彙と前提理解を確立するコマである。この後に学ぶ倫理・品質の問題（03・04）や演習（05・06）で扱う用語と適切利用の判断基準を先に捉え、生成AIに関する誤解を減じて以降の学習を進め易くする。
コマ形態	講義（オンデマンド視聴）
コマ内容	<ul style="list-style-type: none"> <li>○ 生成AIは何か／LLMの概観</li> <li>○ 検索と生成の違いと使い分け</li> <li>○ プロンプトの基本形と目的の提示の仕方</li> <li>○ 初期設定と安全利用の基本</li> </ul>
キーワード	生成AI LLM（大規模言語モデル） 検索 プロンプト 目的定義 出力形式 入力禁止 安全利用
到達目標	<ul style="list-style-type: none"> <li>◇ 生成AIの出力が確率的で条件によって揺れることを理解し、検索等の決定論的処理との違いを説明できる。</li> <li>◇ 生成AIの特性と限界を踏まえ、業務等で適切利用の判断基準が必要になる理由を説明できる。</li> </ul>
理解度確認テスト	<ul style="list-style-type: none"> <li>① 生成AIと検索の差異識別</li> <li>② プロンプト必須要素の選別</li> <li>③ 目的適合プロンプト例の選択</li> <li>④ 入力禁止情報の判別</li> <li>⑤ 出力変動要因と抑制策の選択</li> </ul>
コマ名	<b>02 誤りの起こり方と確かめ方</b>
コマ概要	出力に誤りが混入する仕組を理解し、事実でない文が尤もらしく出る現象の見分け方を学習する。信頼できる情報源の参照、複数資料の照合、言い切り表現の点検等、簡単に試せる確かめ方を習得し、過剰な自動化を避ける姿勢を確認する。
科目内での当該コマの位置付け	01を踏まえ、誤りの混入を前提にして安全・保守的に確かめる視点を獲得するコマである。この後に学ぶ倫理・品質の問題（03・04）や演習（05・06）の検証手順に接続するため、そのチェック観点と再確認の流れを整備する。
コマ形態	講義（オンデマンド視聴）
コマ内容	<ul style="list-style-type: none"> <li>○ 誤りの型の基本（混同・作り話・過度な一般化等）</li> <li>○ 確かめ方の基本（出典確認・複数照合等）</li> <li>○ 言い切り表現の点検と書き直し</li> <li>○ 自動化の限界と利用者による最終確認</li> </ul>
キーワード	誤り 幻覚 検証 出典確認 複数照合 断定回避 フォールバック 最終確認
到達目標	<ul style="list-style-type: none"> <li>◇ 生成AIにおいて誤りや幻覚が生ずる仕組を理解し、それがいかなる場面で問題化するかを説明できる。</li> <li>◇ 出力を無条件に信頼できない理由を理解し、確認が必要になる典型例を挙げて説明できる。</li> </ul>
理解度確認テスト	<ul style="list-style-type: none"> <li>① 幻覚（事実性欠如）例の識別</li> <li>② 一次検証行動の選択</li> <li>③ 断定回避表現の選択</li> <li>④ 検証後対応手順の選択</li> <li>⑤ 最終確認必須業務の判別</li> </ul>

コマ名	<b>03 倫理・権利・プライバシー</b>
コマ概要	入力情報と出力結果の関係を理解し、個人情報の扱い、著作権やライセンスへの配慮、引用と帰属の基本等を学習する。業務で避けるべき入力、持出禁止の情報、共有時の注意等を具体的に整理し、迷った時に（上長等の）責任者に確認する手順を把握する。
科目内での当該コマの位置付け	安全利用の基礎（01）を倫理・権利の観点で補強するコマである。この後に学ぶ組織内ポリシー（04）や権利表示演習（06）に備え、守るべき事項を最小限の条項に整理し、実務に使える言い回しと連絡の流れを整備する。
コマ形態	講義（オンデマンド視聴）
コマ内容	<ul style="list-style-type: none"> <li>○ 個人情報と機微情報の取扱い</li> <li>○ 著作権・ライセンスの基本と引用</li> <li>○ 共有時の注意と社内外の線引き</li> <li>○ 迷った時の相談と記録の残し方</li> </ul>
キーワード	個人情報 機微情報 入力可否 著作権 ライセンス 引用 帰属 相談
到達目標	<ul style="list-style-type: none"> <li>◇ 個人情報・著作権・ライセンス等の基本的な考え方を理解し、生成AI利用時に注意すべき点を説明できる。</li> <li>◇ 引用や帰属が必要になる理由を理解し、適切でない利用例を説明できる。</li> </ul>
理解度確認テスト	<ul style="list-style-type: none"> <li>① 個人情報該当性の判断</li> <li>② 入力不適切情報の判別</li> <li>③ 権利侵害リスク利用例の識別</li> <li>④ 適切な引用要件の選択</li> <li>⑤ 相談・記録・承認手順の選択</li> </ul>
コマ名	<b>04 組織内AIポリシーと品質評価観点</b>
コマ概要	組織内AIポリシーの最小構成を知り、守るべき事項を憶える。事実性・一貫性・再現性等の品質観点を把握し、簡易チェック表の活用法を学習する。作業記録の残し方、承認の流れ、見直しの頻度等を整理し、実務に繋げる形を理解する。
科目内での当該コマの位置付け	倫理・権利の課題（03）を実務運用に落とし込む橋渡しのコマである。演習（05）で使うチェック表と記録様式の雛形を整備し、提出基準を充して評価対象になり得る最小限の品質確認の手順を共有する。
コマ形態	講義（オンデマンド視聴）
コマ内容	<ul style="list-style-type: none"> <li>○ 組織内AIポリシーの最小構成</li> <li>○ 品質観点の基本（事実性・一貫性・再現性）</li> <li>○ 作業記録と承認の流れ</li> <li>○ 見直しと改善の小さな循環</li> </ul>
キーワード	AIポリシー 遵守 事実性 一貫性 再現性 チェックリスト 作業記録 承認フロー
到達目標	<ul style="list-style-type: none"> <li>◇ 組織内AIポリシーと品質観点（事実性・一貫性・再現性）の位置付けを理解し、その目的を説明できる。</li> <li>◇ 作業記録や承認フローが必要になる理由を、運用上のリスクに関連付けて説明できる。</li> </ul>
理解度確認テスト	<ul style="list-style-type: none"> <li>① AIポリシー必須要素の選別</li> <li>② 事実性・一貫性・再現性の対応付け</li> <li>③ 作業記録の必須要素の選択</li> <li>④ 品質問題時の一次対応の選択</li> <li>⑤ 例外利用時の承認要件の選択</li> </ul>

コマ名	<b>05 プロンプトの基本練習と出力の見直し</b>
コマ概要	架空会議のメモを社内共有文に整形し、その要約文と作業項目を作成する。事実性・一貫性・再現性で点検し、根拠資料に照合して誤りと断定過多を是正する。再実行し、形式の揺れも確認する。不明箇所は「要確認」と明示し、差分と判断根拠と停止基準を記録して提出する。
科目内での当該コマの位置付け	講義で学んだ基礎を自分で実践して確認するコマである。設計演習(06)に進む前に、目的定義→作成→確認→修正→提出の最小ループを体験し、記録と差分の残し方も習得する。
コマ形態	演習(ライブ配信)
コマ内容	<ul style="list-style-type: none"> <li>○ 目的の1文化と基本プロンプト作成</li> <li>○ 出力の読み取りと書き直し</li> <li>○ 出典確認と記録</li> <li>○ 直前直後の差分と提出物の整え方</li> </ul>
キーワード	目的定義 プロンプト 要約 箇条書き 検証優先 修正ループ 差分 停止基準
到達目標	<ul style="list-style-type: none"> <li>◇ 目的と条件を明示したプロンプトを作成し、出力を品質観点で点検して是正を実施できる。</li> <li>◇ 改善前後の差分と判断根拠を作業記録として整理し、停止基準を明示できる。</li> </ul>
理解度確認テスト	<ul style="list-style-type: none"> <li>① 目的適合プロンプトの選択</li> <li>② 検証優先情報(数値・固有名詞等)の特定</li> <li>③ 修正ループ手順順序の選択</li> <li>④ 提出・共有に必要な証跡項目の選別</li> <li>⑤ 改善終了(停止)基準の選択</li> </ul>
コマ名	<b>06 データ保護と権利表示のミニ設計</b>
コマ概要	架空案件を想定し、取扱いデータを「公開/社内/機密/個人情報」に区分して入力可否と条件を定める。「保存・共有」も明示する。引用と出典表示の書式、作業記録の必須項目、迷う場合の確認・承認手順を整理し、運用雛形として提出する。
科目内での当該コマの位置付け	S1の総括として、安全利用と品質確認を小さな運用に落とし込むコマである。以降の科目やPBLで再利用できる最小の様式と手順を用意し、初学者も迷わずに作業できる状態の達成を目指す。
コマ形態	演習(ライブ配信)
コマ内容	<ul style="list-style-type: none"> <li>○ データ分類と持出禁止の線引き</li> <li>○ 引用と帰属の書式</li> <li>○ 作業記録の必須事項</li> <li>○ 相談先と承認の流れ</li> </ul>
キーワード	データ分類 公開/社内/機密 PII(個人識別可能情報) 疑似化 引用表示 帰属 記録事項 エスカレーション
到達目標	<ul style="list-style-type: none"> <li>◇ 取扱いデータを区分し、生成AIへの入力可否と条件を判断して整理できる。</li> <li>◇ 引用・帰属・相談先を含む最小限の運用ルールを作成し、記録項目を明示できる。</li> </ul>
理解度確認テスト	<ul style="list-style-type: none"> <li>① データ分類(公開/社内/機密)の判別</li> <li>② PII取扱い安全措置の選択</li> <li>③ 引用・帰属表示の適切性判断</li> <li>④ 監査対応の記録項目の選別</li> <li>⑤ エスカレーション先と手順の選択</li> </ul>

▼【S2】「プロンプト設計とLLM活用」前半 シラバス（6コマ）

コマ名	<b>01 LLMの仕組みとプロンプトの基本形</b>
コマ概要	LLMは、入力文から次の語を予測して文章を生成する。トークン（文章を分割した単位）・入力長・推論設定が出力の揺れに影響する点を把握し、目的・条件・出力形式を明示した基本プロンプトを作成した上で、入力制約を守った入力・設定・出力の対応関係を理解する。
科目内での当該コマの位置付け	S2の導入として、設計パターン・改善ループ・API・RAGに共通する語彙と記録作法を共通化するコマである。演習（02）で出力比較と差分を扱う前提を整え、揺れを前提にして判断する基礎を早期に確立する。
コマ形態	講義（オンデマンド視聴）
コマ内容	<ul style="list-style-type: none"> <li>○ LLMの生成の流れ（確率生成・トークン）</li> <li>○ 入力長と設定が出力に与える影響</li> <li>○ 基本プロンプト（目的・条件・出力形式）</li> <li>○ 入力制約と記録（入力・設定・出力）</li> </ul>
キーワード	生成AI LLM トークン 入力長 推論設定 プロンプト 出力形式 記録
到達目標	<ul style="list-style-type: none"> <li>◇ LLMの内部構造と入力・設定が出力に与える影響を理解し、設計時に留意すべき点を説明できる。</li> <li>◇ 出力の揺れが生ずる理由を理解し、安定化が必要になる背景を説明できる。</li> </ul>
理解度確認テスト	<ul style="list-style-type: none"> <li>① トークン概念の識別</li> <li>② 入力長と出力揺れの関係の把握</li> <li>③ 基本プロンプト要素の判定</li> <li>④ 推論設定変更の影響の推定</li> <li>⑤ 記録事項と入力制約の選択</li> </ul>
コマ名	<b>02 基本プロンプトでの結果の揃え方</b>
コマ概要	架空会議のメモを題材にし、目的別の要約等を生成するプロンプトを複数案作成する。複数回実行で揺れを観察し、条件追加と形式固定で狙い通りの安定出力に整える。入力・推論設定・出力と差分、改善理由を作業記録に残して所定形式で提出する。
科目内での当該コマの位置付け	01で学んだ基本プロンプトを実際に記述して試す最初の演習コマである。この後に学ぶ改善ループやAPI実装で必要になる、出力形式の固定、再現性の確認、差分記録の作法等を、提出まで含めて先に確立する。
コマ形態	演習（ライブ配信）
コマ内容	<ul style="list-style-type: none"> <li>○ テンプレートに基づくプロンプト作成</li> <li>○ 反復実行による揺れの観察</li> <li>○ 条件追加・形式固定による調整</li> <li>○ 差分記録の作成と提出</li> </ul>
キーワード	テンプレート 目的 条件 出力形式 反復実行 再現性 チェックリスト 差分
到達目標	<ul style="list-style-type: none"> <li>◇ 同一入力に対する出力の揺れを観察し、条件追加や形式固定を適用して再現性を高められる。</li> <li>◇ 複数回実行の結果を比較し、その差分と改善理由を記録できる。</li> </ul>
理解度確認テスト	<ul style="list-style-type: none"> <li>① テンプレート必須要素の判定</li> <li>② 出力形式指定の適否の判断</li> <li>③ 揺れ低減の修正方針の選択</li> <li>④ 差分記録事項の選択</li> <li>⑤ 入力制約遵守の判定</li> </ul>

コマ名	<b>03 出力評価の基本——よい答えの見分け方</b>
コマ概要	事実性・一貫性・再現性の3観点で出力を点検する。数値や固有名詞等の検証優先箇所を見つけ、誤りの型を分類して出典確認と断定回避を行う。品質ゲート最低水準と停止基準を定め、判断根拠を作業記録に残す。次の演習で改善に繋げる。
科目内での当該コマの位置付け	02で作成した出力を良否判定できる状態にするコマである。演習(04)で再プロンプト改善を回すため、評価観点・検証優先・停止基準・記録事項を統一する。この後に学ぶRAGやAPIでも同じ軸で点検する。
コマ形態	講義(オンデマンド視聴)
コマ内容	<ul style="list-style-type: none"> <li>○ 3観点(事実性・一貫性・再現性)の点検</li> <li>○ 検証優先箇所(数値・固有名詞等)の特定</li> <li>○ 品質ゲート最低水準と停止基準の設定</li> <li>○ 判断根拠の作業記録(共有前提)</li> </ul>
キーワード	事実性 一貫性 再現性 検証優先 品質ゲート 停止基準 断定回避 作業記録
到達目標	<ul style="list-style-type: none"> <li>◇ 出力評価の観点(事実性・一貫性・再現性)を理解し、各々が必要になる理由を説明できる。</li> <li>◇ 品質ゲートと停止基準の考え方を理解し、演習での適用方法を説明できる。</li> </ul>
理解度確認テスト	<ol style="list-style-type: none"> <li>① 3観点の対応付け</li> <li>② 検証優先箇所の選別</li> <li>③ 品質ゲート最低水準の判定</li> <li>④ 停止基準設定方針の選択</li> <li>⑤ 判断根拠記録項目の選択</li> </ol>
コマ名	<b>04 再プロンプトによる改善ループ</b>
コマ概要	誤りを含む出力を業務利用に堪える水準に引き上げることを目的とする。評価表で問題点を分類し、一次情報で照合して断定回避も含めて再プロンプトで是正する。差分と根拠を記録し、品質ゲートと停止基準を充した時点で提出する。
科目内での当該コマの位置付け	03の評価軸を用いて改善ループを手順として回す演習コマである。この後に学ぶ設計パターン活用やAPI実装でも同じ改善作法を使うため、差分と根拠の記録まで含めて型を確実に固定する。
コマ形態	演習(ライブ配信)
コマ内容	<ul style="list-style-type: none"> <li>○ 問題点抽出と分類(誤り・不足・冗長等)</li> <li>○ 再プロンプトの修正方針と適用</li> <li>○ 一次情報照合と断定回避への修正</li> <li>○ 差分・根拠記録と提出判定</li> </ul>
キーワード	改善ループ 再プロンプト 評価表 出典照合 断定回避 品質ゲート 停止基準 差分記録
到達目標	<ul style="list-style-type: none"> <li>◇ 出力の問題点を把握・分類し、出典確認を行った上で再プロンプトによる是正を実施できる。</li> <li>◇ 品質ゲートと停止基準に基づいて改善を終了する判断を行い、その改善判断の根拠と差分を記録できる。</li> </ul>
理解度確認テスト	<ol style="list-style-type: none"> <li>① 問題点分類の選択</li> <li>② 再プロンプト修正方針の選択</li> <li>③ 出典照合要否の判定</li> <li>④ 停止基準到達の判定</li> <li>⑤ 差分・根拠記録の要点の選択</li> </ol>

コマ名	<b>05 プロンプト設計パターン</b>
コマ概要	役割付与、分割統治、手順の段階化、出力例提示等の基本パターンを学習する。曖昧な要求を分解して追加質問を引き出す方法も扱い、誤誘導を避ける制約条件の書き方を習得する。再利用テンプレートとして整える観点も確認し、次の演習で効果を比較する。
科目内での当該コマの位置付け	改善ループ (04) で見えた課題に対し、修正の引出しを増やすコマである。次の演習 (06) でパターンを組み合せ、品質と再現性が上がる構造を比較して選ぶため、型と注意点を本コマで整理する。
コマ形態	講義 (オンデマンド視聴)
コマ内容	<ul style="list-style-type: none"> <li>○ 役割付与と前提条件の与え方</li> <li>○ 出力例提示と出力テンプレート化</li> <li>○ 段階化 (手順化) と分割統治の考え方</li> <li>○ 追加質問と制約条件 (誤誘導回避)</li> </ul>
キーワード	役割付与 出力例 テンプレート 段階化 分割統治 追加質問 制約条件 誤誘導
到達目標	<ul style="list-style-type: none"> <li>◇ プロンプト設計パターン (役割付与・分割統治・段階化等) を理解し、使い分けの考え方を説明できる。</li> <li>◇ 誤誘導を避けるための制約条件や追加質問の役割を説明できる。</li> </ul>
理解度確認テスト	<ol style="list-style-type: none"> <li>① 役割付与の適用場面の選択</li> <li>② 出力例提示効果の判定</li> <li>③ 段階化・分割統治の使い分け</li> <li>④ 追加質問設計意図の判定</li> <li>⑤ 制約条件記述の適否の判断</li> </ol>

コマ名	<b>06 パターン組合せによる業務プロンプトの作成</b>
コマ概要	問合せ対応を題材にし、役割付与や分割統治等を組み合わせて業務プロンプトを設計する。二案を作成して品質と再現性を比較し、採択理由と注意点を短文で整理する。改善ループを一度回し、出力形式を固定して差分と根拠を記録し、提出する。
科目内での当該コマの位置付け	05の設計パターンを実際の課題に適用して定着させる演習コマである。作成したプロンプトは実習 (08) のAPI呼出しで流用するため、出力形式の固定と記録様式を本コマで確実に統一する。
コマ形態	演習 (ライブ配信)
コマ内容	<ul style="list-style-type: none"> <li>○ パターン選択と組合せ設計</li> <li>○ 複数案の比較評価 (品質・再現性)</li> <li>○ 改善ループ併用と最終案決定</li> <li>○ 設計意図整理と提出 (記録添付)</li> </ul>
キーワード	パターン組合せ 業務課題 出力形式 再現性 改善ループ 比較評価 設計意図 記録
到達目標	<ul style="list-style-type: none"> <li>◇ 設計パターンを組み合わせて業務想定に適合するプロンプトを作成し、出力形式を固定できる。</li> <li>◇ 複数案を比較評価し、その採択理由と設計意図を記録して整理できる。</li> </ul>
理解度確認テスト	<ol style="list-style-type: none"> <li>① パターン組合せ順序の選択</li> <li>② 出力形式統一要点の判定</li> <li>③ 比較評価観点の選択</li> <li>④ 改善ループ適用の判断</li> <li>⑤ 提出記録の充足の判定</li> </ol>

尚、演習コマは、指導の再現性を担保するべく、“導入→作業→提出→講評”という標準フローを定め、課題趣旨・前提・禁止事項・提出物・評価観点等を示した上で、作業記録（操作ログ・判断根拠）の提出を必須要件として組み込んだ。

講義コマも、オンデマンド視聴後に自習CBT（理解度確認テスト）を課し、到達確認を経て次コマに接続する運用を前提にし、出題観点をシラバス段階で確定している。

以上のように、カリキュラムで定めた教育の“順序と量”は、シラバスによって授業運用可能な“型”と“評価接続”に落とし込まれた。

次節（3.5）では、このシラバスを一次仕様として整備した教育素材、すなわち講義・演習・実習・PBL・自習に対応する教育コンテンツの開発内容について詳説する。

#### 演習の標準フロー

- ① [導入] 指導者が課題趣旨・前提・禁止事項・提出物・評価観点を明示する。
- ② [作業] 受講者がテンプレート（チェックリスト・記録様式）に基づいて作業し、操作ログ・判断根拠を作成する。
- ③ [提出] 受講者がLMSに成果物を提出し、必要に応じてGitに履歴・差分を残す。
- ④ [講評] GPT-TAの自動評価（事実性・一貫性・再現性）  
+ 指導者レビューで総評・是正指示を行う。

## 3.5 教育素材——教育コンテンツ

前節(3.4)では、BASISで定義した到達目標を教育体系、すなわちカリキュラム(教育の“順序と量”)とシラバス(各コマの指導計画)に変換し、教育プログラムを運用可能な設計として確定した。

しかし、設計が確定しても、それだけで教育は実施できない。指導者が同一水準の指導を再現し、受講者に一定の成果物品質を要求し、それを評価に接続するためには、授業を成立させる**教育素材**、すなわち講義動画・講義資料・演習課題・実習シナリオ・PBL教材・自習問題等の**教育コンテンツ**群を、仕様として整備しておく必要がある。

本教育プログラムが企図する“生成AIを道具として使い熟し、要件定義から運用自動化までを短期間で自走できるIT人材”の養成は、知識伝達だけで達成できない。**講義**で必要最小限の理論と判断基準を与え、**演習・実習**で手順と制約を与えた上で反復させ、最終的に**PBL**で実務適用として統合する——この**教育形態**の役割分担を、具体的な教材・課題として実装して初めて、教育プログラムは機能する。

そこで、本節(3.5)では、教育5形態に対応する教育コンテンツの構造を整理した上で、本年度(令和7年度)事業の先行開発範囲である【S1】と【S2】前半について、講義動画・講義資料(3.5.2)、演習課題(3.5.3)、そして自習CBT問題(3.5.4)の整備内容を報告する。併せ、次節(3.6)で扱う教育基盤(教育プラットフォーム・GPT-TA評価システム)との接続を見据え、提出物・記録様式・評価観点を後続工程で一貫して扱えるよう、コンテンツの単位と体裁を揃える方針を示す。

### 3.5.1 開発方針

前節(3.4)では、教育プログラム(AI補完型ITエンジニア養成プログラム)を“7科目・60コマ”の教育体系(カリキュラム)として確定し、その各コマを授業運用可能な粒度の指導計画(シラバス)に落とし込んだ。ただし、カリキュラムとシラバスはあくまで“何を、どの順序と量で、いかなる形態で教えるか”を規定した設計情報であり、それ自体が教育行為を代替するものでない。教育プログラムを現場で再現可能なものとして成立させるためには、指導者が一定の品質と手順で授業を実施できるよう、教材・課題・評価接続情報を“教えるための実装仕様”として整備する工程が不可欠である。

本事業の対象エリアである人口減少地域では、教員不足や地理的制約が常態であり、同質の教育を各地で反復可能にするためには、指導内容を口頭伝達や属人的裁量に依存させず、コンテンツ自体に指導の骨格を埋め込む必要がある。すなわち、講義は体系的解説の標準化、演習は手順と評価観点の固定化、実習は環境・シナリオ・採点フローの整備、PBLは成果物要件と審査観点の規定というように、教育形態毎に“指導の設計”を教材に移植することが、本事業における教育コンテンツ開発の基本方針である。

また、教育コンテンツは独立した成果物でなく、“BASIS(教育目標)→カリキュラム(教育体系)→シラバス(指導計画・授業仕様)→教材(提示物)→受講生解答・提出物(記録)→評価(講評)”という一連の鎖の中で

機能する。そのため、本年度事業における開発・整備では、次節（3.6）で扱う教育基盤（学習管理システム（LMS）・クラウド統合開発環境（IDE）・GPT-IA評価システム）に接続可能な形式を前提にし、配信単位・命名規則・メタデータ（範囲・キーワード等）を予め整流化した。

尚、本年度事業では、次年度に実施予定の先行実証講座（2週間）の対象範囲に合せ、【S1】 + 【S2】 前半（計12コマ）を教育コンテンツ開発の起点にした。具体的には、**講義コンテンツ**として〈講義映像〉〈講義資料〉を7コマ、**演習コンテンツ**として〈演習課題〉を5コマ、**自習コンテンツ**として〈理解度確認CBT問題〉を12コマ（60問）分作成した。その際、次年度以降に拡張される実習とPBLは、カリキュラム上の位置付けと接続条件を明確にした上で、そのコンテンツ化は次年度開発範囲として切り分けている。

教育形態	本年度対象コマ	成果物	数量	備考
講義	【S1】 01～04 【S2】 01, 03, 05	講義動画（mp4） 講義資料（スライド等）	7コマ	1コマ10～20分、AIアバター教員、 自習CBT連携メタデータ付与
演習	【S1】 05, 06 【S2】 02, 04, 06	演習課題 （課題・シナリオ・指導者ガイド）	5題	テンプレートに基づく作業とログ管理が 必須
自習	【S1】 01～06 【S2】 01～06	理解度確認CBT問題 （講義準拠）	60問 （12コマ）	1コマ5問、正答率80%未満は再受験
実習	— （次年度以降開発）	実習計画 （開発案件・シナリオ・評価フロー）	— （次年度検討）	カリキュラム上の位置付けを確定して 次年度に設計
PBL	— （次年度以降開発）	PBL計画 （企業課題・成果物要件・評価フロー）	— （次年度検討）	企業課題との接続を踏まえて次年度に設計

#### 本年度に開発した教育コンテンツ（対象・数量等：教育形態別）

以上の開発方針と開発範囲を踏まえ、教育コンテンツは、カリキュラムとシラバスで策定した授業仕様を起点にし、講義（動画・資料）・演習（課題）・自習（CBT問題）が順次具体化され、各コマで求める成果物と評価観点が整合する形で制作された。次項以降（3.5.2～3.5.3）、教育形態に対応して本年度に開発した教育コンテンツ（講義コンテンツ・演習コンテンツ・自習コンテンツ）を順に示す。

### 3.5.2 講義コンテンツ——講義動画・講義資料

**講義**は、教育内容の理論・枠組・判断基準を、指導者主導で体系的に提示する教育形態である。

本教育プログラムでは、講義を単なる説明時間として扱わず、後続の演習・実習・PBLで常に参照され続ける“判断の根拠”を確立する教育要素として位置付ける。そのため、**講義コンテンツ**は、理解のための提示物（動画・スライド）に留めず、参照・再確認・評価に連動する設計で制作する必要がある。

本年度事業では、【S1】 01～04および【S2】 01・03・05の講義（7コマ）を対象にし、その**講義動画**と**講義資料**を“対”で作成した。

講義動画は、各コマ10～20分を目安にして収録し、オンデマンド視聴を前提にした配信単位に統一することで、地域差・担当者差を極小化し、反復可能な指導を担保する。各動画には、チャプター・台本・理解度確認CBT連携メタデータを付与し、単に視聴させるだけでなく“どの論点をどの順序で理解させるか”“理解度確認をどの範囲で実施させるか”をコンテンツ側で制御できるようにした。

その際、講義動画は、eラーニングの展開と共にその形態が更新されてきた。その変遷を機能面に限定して最小限に整理した上で、本事業で制作した講義動画の様式を位置付ける。

1970年代頃	授業のテレビジョン放送 放送・収録型の遠隔教育で、教員が教室で講義する様子を撮影・放送する“授業の映像化”が成立（1971年にOpen University放送講義（BBC）が開始）。
1990年代～2000年代前半頃	スライドの普及と収録・編集環境の整備（講義収録配信の拡大） プレゼンテーション資料（スライド）の一般化と収録・編集環境の整備により、講義の収録配信（lecture capture）やスライド中心の講義映像が拡大。
2000年代前半～2008年頃	lecture captureの制度・装置化 動画圧縮技術や配信環境の整備により、大学等で講義を収録して配信する“lecture capture”が制度・装置として普及。
2000年代前半頃	〈スライド“紙芝居”＋音声〉形式の一般化 画面収録（screencast）型の制作ツールが普及し、〈スライド“紙芝居”＋教員ナレーション〉（教員像の非表示）の制作が容易化したことで、教室収録に依存しない“画面中心”講義映像の制作が拡張。
2005年以降 & 2008年頃以降	Web動画視聴環境の一般化 2005年（YouTube開始）以降、Web動画の視聴環境が一般化。 2008年頃以降、（Khan Academyに代表される）〈スライドor板書画面＋音声〉の短尺説明スタイルが拡大。
2010年代前半頃	スタジオ型〈スライド＋教員像〉の標準化と短い動画単位の運用の一般化 2012年頃（MOOC普及期）、オンライン講座の大規模展開に伴い、スタジオ収録や〈スライド＋教員像〉を同時提示する制作形態が標準化。これに伴い、“短い動画単位で学習を進める”運用が一般化。その過程で、教員の収録負担と視認性（見易さ）の最適化が課題化。  アニメ教員スタイルの拡張 アニメーションによる“説明役”（アニメ教員）を用いた教育コンテンツが普及し、教員の出演負担を抑えつつ視覚提示を制御するスタイルが拡張。
2010年代後半 & 2016年頃以降	ニューラル音声合成の高度化による〈台本→自然音声〉化 2010年代後半、ニューラル音声合成（TTS）が進展し、台本テキストから自然な音声を生成するTTSの品質が向上。 2016年以降、TTSの実用化・高度化により、〈スライド＋機械“読み上げ”〉の制作が加速。
2017年頃以降	AIアバター話者の商用化 生成AI・合成映像技術により、実在人物でないが人間に近い外観の“AIアバター話者”が講義映像に登場するスタイルが商用化・拡大。教員非表示時代を経て、教員の出演負担を抑えながら“話者の存在”を担保するスタイルとして浮上。
2024年頃以降 & 2025年頃以降	表現力・リップシンク・動作の向上 2024年頃以降、顔の表情や口調・抑揚、口形の整合性（リップシンク）を含む表現力が向上し、より自然な話者表現を伴うAIアバター映像が実装段階に移行。 2025年頃以降、AIアバターの全身動作やリップシンク精度が更に向上する局面に進展。

### 講義映像変遷略史

本事業で制作した講義動画は、上記の展開における「短尺スタイル」と「AIアバター＋リップシンク」を組み合わせ、オンデマンド視聴教材として可読性・明瞭性・反復視聴性を高める設計にした。具体的には――

- 各コマを短尺講義に設定し、受講者が“理解の単位”を自覚しながら学習できるようにした。
- ナレーションは、機械“読み上げ”に留めず、リップシンク適用のAIアバターを用いて表情・口調・抑揚を調整可能な形で提示することで、情報伝達の単調化を回避し、理解の保持を支援するようにした。その際、表情・口調には複数のバリエーションを設定し得るが、本教育コンテンツでは聴き取り易さを重視し、

抑揚を少し持ちながら落ち着いて要点を明確に示す“セミナー講師風”の話し方を基調としている。これにより、重要語・重要文の強調、情報の区切り、受講者の注意の向け先を動画内で明示し易くすることを企図した。

また、各講義動画は、シラバスに基づいて次のような手順で制作した。

- ① コマ毎の講義台本（説明順序・例示・用語）の作成
- ② スライド構成（視覚提示・キーワード）への落とし込み
- ③ 収録・編集（音声・画面提示の整合、冗長部の削減）
- ④ 最終レビュー（シラバスとの整合、日本語表現の平易化、誤解誘発表現の修正）

特に④ 最終レビューでは、内容の正確性に加え、受講者が後続の演習や講義で何を学ばばよいかを講義末尾で見通せることと、自習問題（理解度確認テスト）に対応する要点が講義内で明確に扱われていることを確認した。

学校法人電子学園 日本電子専門学校  
AI補完型ITエンジニア養成プログラム

【科目S1】生成AIリテラシーとAI倫理  
01 生成AIの基礎と安全利用の前提

### 要素1 & 2: 目的と条件

**目的 (役割とタスクの定義)**

「あなたはシニアJavaエンジニアです」

役割を与えることで、  
専門的な視点や用語選びを引き出す。

**条件 (制約事項とコンテキスト)**

「Java 17, Spring Bootを使用」  
「初心者向けに」

技術スタックやトーンを指定し、  
回答の精度を高める。



学校法人電子学園 日本電子専門学校  
AI補完型ITエンジニア養成プログラム

【科目S2】プロンプト設計とLLM活用  
01 LLMの仕組みとプロンプトの基本形

### ② LLMの生成の仕組み (トークンの予測と自己回帰)

LLMの生成は、驚くほど地道な「繰り返し作業」です。

▶ステップ1 (トークン化)    ▶ステップ2 (確率分布の計算)



▶ステップ3 (トークンの選択)    ▶ステップ4 (自己回帰)

→次にくるトークンが「名前はまだない」の確率は90%、

→次にくるトークンが「吾輩は猫である。名前はまだない」の確率は5%……



講義動画 画面例 (【S1】01)

コマ名	再生時間	備考（仕様等）
【S1】 01 生成AIの基礎と安全利用の前提	14分26秒	
【S1】 02 誤りの起こり方と確かめ方	13分37秒	
【S1】 03 倫理・権利・プライバシー	16分14秒	ファイル形式：mp4
【S1】 04 組織内AIポリシーと品質評価観点	15分49秒	コーデック：MPEG-4 AAC、H.264 画面サイズ：1920×1080
【S2】 01 LLMの仕組とプロンプトの基本形	16分03秒	カラープロファイル：HD (1-1-1) オーディオチャンネル：ステレオ
【S2】 03 出力評価の基本——よい答えの見分け方	10分53秒	
【S2】 05 プロンプト設計パターン	19分21秒	

### 本年度事業で制作した講義動画一覧

他方で、講義資料は、動画の同内容の論理展開を保持しつつ、参照性と再利用性を優先した構成にした。具体的には、講義スライドに加えて講義ノート（台本+補足説明・注意点等）や用語集（キーワード解説）を附属し、オンライン授業・オフライン参照のいずれでも指導の一貫性を保てるように整備した。その際、講義資料は“配付教材”であるのと同時に、演習課題における禁止事項や判断基準、実習・PBLにおける品質基準の根拠としても機能するため、表現の曖昧さや例外条件の抜けを許容しない方針で記述規約を統一している。

以上のような講義動画と講義資料は、教育プラットフォーム上のLMS（Learning Management System「学習管理システム」）での提示・視聴履歴管理・自習導線と不可分であることから、講義1コマを基本単位として“動画・資料・連携情報”を紐付け、運用管理上の迷いが生じない形で整備した。これにより、指導内容の標準化と後続形態（演習・自習）への滑らかな接続が可能になる。

特に本事業では、生成AIの誤用・過信が業務リスクに直結し得ることを踏まえ、講義段階で判断基準を確立し、それを教育コンテンツとして固定化することで、教育品質を“指導者の属人的な判断・実践”から“教材仕様”に移管することを狙っている。

### 3.5.3 演習コンテンツ——演習課題セット

**演習**は、講義で与えた概念・判断基準を具体的な操作・手順に変換し、一定の再現性を持って遂行させる教育形態である。

本教育プログラムにおける演習は、自由制作や偶発的試行錯誤に委ねるのではなく、指導者が工程・前提・評価観点を明示し、それに基づいた作業を課すことで、基礎技能を確実に定着させることを狙う。生成AIを“道具として使い熟す”ためには、出力の妥当性判断、再現手順の保持、改善ループの運用等の一連の実務作法が必要であり、演習はこれらを“型”として教え込む段階に相当する。

本年度事業では、【S1】 05・06および【S2】 02・04・06の演習（5コマ）を対象にし、“課題・指導ガイド・シナリオ”から成る演習課題セットを整備した。この課題セットは、① 課題趣旨と作業範囲、② 前提・禁止事項、③ 作業手順（工程分解）、④ テンプレート（チェックリスト・記録様式）、⑤ 提出物要件、

⑥ 評価観点（自動評価／指導者講評の双方）を構成要素として持ち、いづれの実施環境においても同一の指導が可能になるように形式を統一した。

その際、本教育プログラムの演習では、成果物そのものに加え、操作ログや判断根拠の記録を“提出物の一部”として扱う。生成AIを介した作業は、外形的成果物だけでその妥当性を検証するのが困難で、再現性や一貫性の担保も難しい。そのため、演習段階から、i) 作業過程を記録させ、ii) 根拠を明示させ、iii) 再現可能な形で提出させるという手順を標準化し、評価対象として固定した。

演習の運用フローは、指導者が課題の枠組を設定し、受講者に統一テンプレートで作業を行わせ、LMS上で成果物とログを回収し、GPT-TA評価システムによる自動評価と指導者講評を組み合わせる是正指示を与えるという4段階（指導→作業→提出→講評）で構成する（3.4.3 p.123）。本年度事業で作成した演習課題セットも、この標準フローを前提にして整備してい、次年度に設計する実習・PBLに移行する際にも同一の評価観点と提出規程を共有できるように設計している。

以上のように、演習は“講義で与えた概念・理論を、指導者が提示する手順に翻訳し、作業と記録を一体で評価可能にする”段階として機能する。これにより、次年度以降に拡張される実習（半）自律的実装）やPBL（企業課題解決）でも、作業規律と品質基準が連続する。

**前提条件（全演習共通）**

**2 共通テンプレート**

2-2 品質チェック表 (A4) テンプレート (必須)

- 事実性（一次情報との照合）
- 数値／日付／固有名詞／決定事項：一致しているか
- 推測を断定していないか（「～である」等）
- 一貫性（文内整合）
- 決定事項とToDoが矛盾していないか
- 用語（表記）が統一されているか
- 再現性（形式・構造）
- 指定フォーマットが守られているか
- 同条件で再実行しても要点が維持されるか

2-3 改善ループ評価表 (S2-04用 必須)

問題点タイプ	例	修正方針
誤り（事実不一致）	一次情報と異なる数値	根拠箇所を指定して再生成
不足	必要項目が欠落	必須項目を列挙して補充
冗長	無関係な説明が多い	出力長・構造を制約
断定過多	不明点を断言	「要確認」に置換+確認先
形式揺れ	見出し類が変る	出力テンプレート固定

AI補完型ITエンジニア養成プログラム【演習課題】 3

**S1-05 プロンプトの基本練習と出力の見直し**

**a 課題（受講者向け）**

**課題趣旨** ▶ 会議メモを社内共有文に整形し、要約文と作業項目（ToDo）を作成する。出力を事実性・一貫性・再現性で点検し、根拠資料に照合して誤りと断定過多を是正する。差分・判断根拠・停止基準を作業記録として残す。

**配付資料** ▶

- ◇【資料A】 会議メモ（ドラフト）
  - 【会議名】 北星工業 受発注システム刷新 キックオフ
  - 【日時】 2025/09/18 14:00-15:00
  - 【参加】 北星工業（業務・技術、情報、田中）/ベンダ（PM:鈴木、SE:小林）
  - 【目的】 現行Excel受発注をWeb化し、入力二重化と属人化を解消する
  - 【メモ】
    - まずMVPは「受注登録・検索・一覧・CSV出力」まで。請求は第2段階。
    - 所属部門は営業/開発、参加4名。参加期は入力が基本。
    - 要件定義は10月まで、開発は11月～。テストは12月前半、1月から段階導入。
    - 次回は10/15（水）10:00。議題：画面項目確定、権限、CSV形式。
    - 既存マスタは商品約1200件、得意先約300社。移行は担当前に表記揺れあり。
    - 懸念：受注番号採番ルールが部門で異なる。入力ミス根拠が弱い。
    - 決定：二要素認証は必須（?）。操作ログは残す。
    - 宿題：北星工業は現行Excelを共有、ベンダは簡易画面案を作る。
- ◇【資料B】 決定事項（確定版：一次情報）
  - MVP範囲：受注登録/検索/一覧/CSV出力（請求は第2フェーズ）
  - 要件定義完了目標：2025/10/31
  - テスト期間：2025/12/01～12/15
  - 段階導入開始：2026/01/05
  - 次回会議：2025/10/16（水）10:00（オンライン）
  - セキュリティ：二要素認証は必須でなく、初期は「盗難なパスワード+IP制限」を優先。一要素は次フェーズで再検討。
  - ログ：監査目的で「誰が・いつ・何を変更したか」を必須で記録。
- ◇【資料C】 用語（本演習での表記統一）
  - 「北星工業」= 株式会社北星工業
  - 「刷新P」= 受発注システム刷新
  - 「MVP」= Minimum Viable Product（実用最小限の製品）

AI補完型ITエンジニア養成プログラム【演習課題】 4

**演習課題セット例（全コマ共通 & 【S1】 05）**

### 3.5.4 自習コンテンツ——理解度確認CBT問題

**自習**は、講義内容の理解度を点検し、必要な再教育（再視聴・再説明・再演習）を促すための“即時の理解確認”機能として設計した。

本教育プログラムでは、講義の直後に短時間で取り組める**CBT**（Computer-Based Testing）を課し、正答率が80%に達しない場合は再受験を要件にすることで、知識の脱落や誤解を放置しない運用を基本とする。自習は受講者の任意努力に委ねる領域でなく、教育提供側が理解到達を確認し、補強を強制できる機構として位置付けた。

本年度事業では、【S1】01～06および【S2】01～06の自習（12コマ）を対象にし、各コマ5問、計60問のCBT問題を作成した。問題は講義内容に準拠し、重要概念・用語定義・判断基準・典型的誤解（失敗モード）等を中心にして出題する。出題形式は全問択一式で空欄補充・正誤判定・事象理解の3種を基本にし、短時間で取り組める一方で、単純な暗記に偏らず、定義の正誤や判断基準の適否を問えるように設計した。

その際、この3種の出題形式を採用したのは、受講者が何をどう理解すべきかを、出題の仕方からも捉え易くするためである。この3形式は各々、測定し易い能力要素が異なるため、併用することで内容妥当性の欠落（特定形式でしか拾えない理解の偏り）を回避し、誤解の所在を把握し易くする。各形式には次のような特徴がある。

#### ▼ 空欄補充

講義内容におけるキーセンテンス（理解の要点になる説明文・定義文・規範文等）を提示してその中核を成す概念・用語等を空欄にし、適切な語を選択させる問題である。この形式が測定する中心は、“文脈において概念と語を対応付けられているか”という理解である。単なる用語暗記でなく、キーセンテンスが示す関係（役割・条件・線引き・手順の要点等）を踏まえて語を充当できるかを点検できるため、当該講義で扱う核（講義の結論や判断基準）を直接的に測定し易い。また、誤答肢を近接概念や混同し易い用語で構成することで、受講者がどの概念境界で誤解しているかを示し易く、識別力（理解の差を分ける力）を確保し易い利点がある。

#### ▼ 正誤判定

講義内容におけるキーセンテンス（またはそれに近い文）を提示し、その記述が適切か、あるいは一部に誤りを含むかを判断させる問題である。この形式が測定する中心は、概念・規範・条件の理解を前提にした“記述内容の評価（正しい／誤りを見抜く力）”である。受講者は、単に語を知っているだけでなく、当該講義で示された基準・条件・線引きに照して文の妥当性を判定する必要があるため、誤概念の固定化を抑止する点検として機能する。また、正誤判定は、学習者が陥り易い“尤もらしい誤り”（条件の欠落、適用範囲の誤り、例外の無視等）を扱い易く、講義の要点を“批判的に読み取る”観点から確認できる利点がある。他方で、曖昧性が混入すると測定誤差が増えるため、設計上は“誤りが生ずる箇所を限定し、判定根拠が講義内容に一意に回収できる”記述を基本にする。

#### ▼ 事象理解

「～は何か」「～を何と言うか」「～はどれか」「～はどの程度か」等の短い問いに対する最も適切な回答を選択させる問題である。この形式が測定する中心は、講義で扱った事象に関する即時的な理解（定義・分類・対応関係の把握）であり、当該講義の中核事項を“迷わずに特定できる”水準に到達しているかを点検するのに適している。特に、講義の学習成果として最低限に求める“概念の輪郭の把握”を確認する役割を担い、講義後点検としての即時性（短時間で到達状況を確認できる）に優れる。設計上、正答が一義に定まる問いを中心にして誤答肢を典型的誤解（類似語、条件の読み落とし、範囲の取り違い等）に対応させることで、理解の浅さを可視化し易くする。

このように、空欄補充問題はキーセンテンスの核を再構成できるか、正誤判定問題は基準に照して誤りを識別できるか、事象理解問題は中核事項を即時に特定できるかを各々点検する。この3形式を組み合わせることで、講義内容・到達目標に対する点検が“語の知識”に偏らず“文脈での理解”“誤りの識別”“即時の把握”という複数の観点で成立するように設定した。

#### ▼ 空欄補充 (【S1】 01)

プロンプト設計の説明として「目的・条件・( ) (字数・箇条書き等) を明示する」がある。空欄に入る最も適切なものは次のうちどれか。

- ① 出力形式      ② 入力制約      ③ 対象読者      ④ 参照資料

#### ▼ 正誤判定 (【S1】 01)

「生成AIの出力変動を抑えるには、出力形式と条件を固定することが有効である」という文の内容は正しいか、誤っているか。

- ① 正しい      ② 誤っている

#### ▼ 事象理解 (【S2】 01)

システム開発で用いる基本プロンプト要素として最も適切なものは次のうちどれか。

- ① 目的・条件・出力形式      ② 目的・装飾・語尾  
③ 条件・余白・色      ④ 形式・比喩・絵文字

#### 自習 (理解度確認CBT) 問題例 (【S1】 01 & 【S2】 01、空欄補充・正誤判定・事象理解)

運用面では、正答率80%未満の場合に再受験を課すことを前提に、問題セット (5問) の粒度をコマ単位で統一し、講義動画のチャプターやキーワードと照合可能な形でメタデータを整備する。これにより、誤答が生じた論点に即時に戻らせ、再提示 (再視聴・再確認) を指示できる。さらに、CBT解答の履歴は教育基盤に記録され、次節 (3.6) で扱うGPT-TA評価システムに接続されることで、弱点傾向の把握と指導改善に資する情報になる。

以上のように、自習 (理解度確認CBT) は“講義内容を理解したつもり”を許容せず、理解の不足を早期に顕在化させ、再教育に接続するための統制機構として機能する。講義・演習・実習・PBLの各段階 (教育形態) はいずれも、前段の理解を前提にするため、基礎段階での理解確認を教育コンテンツとして制度化しておくことは、教育モデル全体の安定運用に直結する。

\*

本節 (3.5) では、カリキュラムとシラバスで策定した授業仕様を運用可能な教材仕様に変換するものとして、教育コンテンツを整備した。特に本年度事業では、次年度に実施する先行実証講座 (2週間) の対象である**【S1】 + 【S2】 前半**を起点にし、講義 (7コマ) の動画・資料、演習 (5コマ) の課題セット、自習 (12コマ) のCBT問題 (60問) を作成し、教育内容の提示・理解確認・作業規律・評価接続を指導者主導の枠組としてコンテンツ側に固定化した。

これにより、本教育プログラムは“設計 (目標・体系) → 実装 (授業・教材) → 運用 (履歴・提出) → 評価 (講評) → 改善 (次年度反映)”という循環の入口に到達した。

次年度以降の取組では、本年度事業での整備方式を基準にし、対象科目を**【S2】 後半 ~ 【S4】**に拡張すると共に、実習計画・PBL設計 (教材化) の具体化を進め、教育形態の後段 (実習・PBL) に連続的に接続する教材体系として完成度を高める。

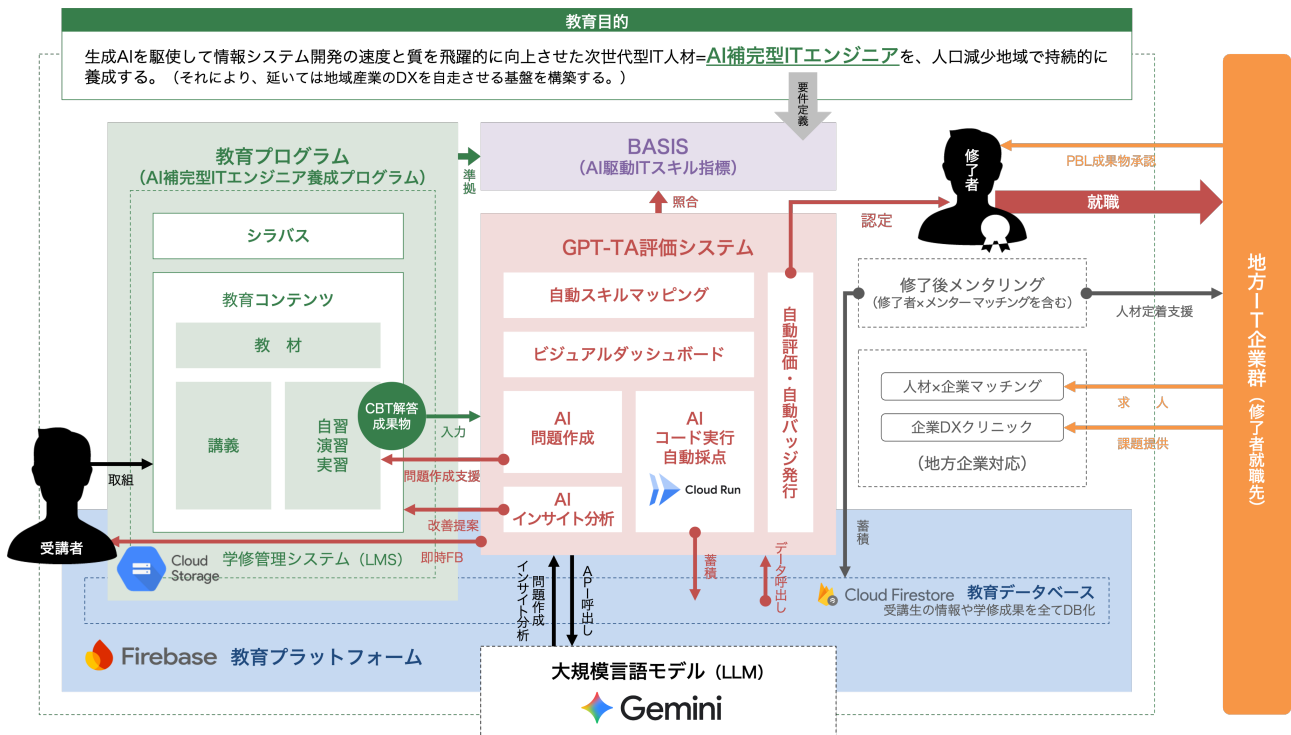
### 3.6 教育基盤——教育プログラムの運用

前節 (3.5) では、カリキュラムとシラバスで策定した授業仕様を、各種の教育素材（講義動画・講義資料・演習課題・自習CBT問題等）として具体化する過程を見た。しかし、教育コンテンツが整備されただけでは、人口減少地域における複数拠点で同一の教育を再現し、教育の到達度を測定し、次の指導設計に還流させることはできない。特に本事業は、BASIS（AI駆動ITスキル指標）を教育目標に掲げる以上、“教えた内容”を“測定可能な記録”として残し、評価と改善を循環させる運用機構を教育プログラム内に組み込む必要がある。

そこで、本節 (3.6) では、教育プログラムの運用を担う**教育プラットフォーム**と、指導・評価・可視化・改善の支援を担う**GPT-TA評価システム**について、本年度事業で取り組んだ要件定義と設計の内容を概観する。

その際、教育プラットフォームは、学習管理システム（LMS）を中核に据え、コンテンツ提示・履歴管理・提出物回収・学習環境（クラウドIDE等）提供等を通じてオンライン教育を実現する共通基盤である。他方で、GPT-TA評価システムは、CBTの自動採点と誤答分析に留まらず、提出物の評価と自然言語フィードバック、到達度の可視化とバッジ発行、作問支援や教材改善提案等、評価を単なる“採点作業”にせず“教育品質を向上させる統制機構”である。

本年度（令和7年度）事業では、次年度の本格構築に先立ち、運用像の具体化を優先して設計を進めた結果、教育プラットフォームとGPT-TA評価システムを一体の**教育基盤**として統合する構成に帰着した。以下では、技術仕様の子細に深入りせず、主に教育的意義と運用上の要請に即し、その位置付けと設計上の要点を整理する。



教育プラットフォームを基盤にしてGPT-TA評価システムを中心に据えた教育システム全体構成

### 3.6.1 設計方針と取組範囲

本事業で開発する教育プログラムは、教育体系として【S1】～【S6】+ PBLの連続構造として設計され、そこに講義・演習・実習・PBLの各教育形態が段階的に配置される。

他方で、教育の提供・管理・評価を地域横断で成立させるためには、単に“授業を実施した”事実だけでは不十分で、教育提供側がその実施内容と成果を同一の枠組で把握し、是正と改善を繰り返せる仕組が要請される。すなわち、本**教育基盤**は、単に教育コンテンツを配信する器でなく、教育を統制・再現するための装置である。

この要請に応えるべく、本事業では教育基盤を**教育プラットフォーム**と**GPT-TA評価システム**の二層で捉える。ただし、この両者は、運用上のデータ（履歴・採点結果・提出物・到達度等）を共有し、教材プログラムの改善に還流させる点では不可分である。そのため、設計上は、計画当初の“別システム並立”でなく、教育プラットフォームを中核とし、その内部機能（サブシステム）としてGPT-TA評価システムを統合する構成に整理した。これにより、コンテンツ配信・学習管理・評価・分析が分断されず、次年度以降の拡張においても、同一の履歴基盤の上で機能追加が可能になる。

その際、本年度事業では、教育プログラムの先行実証範囲（【S1】+【S2】前半）にわせ、講義視聴・演習提出・CBT等の基礎段階のログ取得と評価処理を優先して設計した。実習・PBLは、次年度以降の教材化と運用設計に接続できるよう、提出物管理・評価観点・記録要件が拡張可能な形で、基盤側の受け皿を先に定義している。

### 3.6.2 教育プラットフォーム——提供・管理・記録の統合基盤

**教育プラットフォーム**は、教育提供側が“何を・いつ・誰に・どの順序で・どの条件で教えたか”を管理し、それと同時に“どの取組が行われ、いかなる成果が出たか”を履歴として蓄積するための統合基盤である。ここで言う「統合」は、単に機能を並置することでない。教育の運用をコンテンツ提示・提出物受領・採点・講評・改善という一連の流れとして通貫し、属人的運用を介さずに再現できる状態に整えることを意味する。

本教育プラットフォームはまず、LMS（学習管理機能）をその中核に据える。具体的には、教育コンテンツ（講義動画・講義資料・演習課題・自習CBT問題）提示、受講者管理、進捗管理、履歴（視聴・提出・解答）記録、連絡・周知等、教育運用の基本動作を一つの枠組で統制する。人口減少地域では、指導者の配置や時間割が各地域に揺れ易い。そのため、コンテンツ提示の単位、提出物の形式、記録の粒度を基盤側で規格化し、“運用の揺れ”を最小化することが要点になる。

また、本教育プログラムでは、演習・実習を通じて生成AIを業務の道具として扱う技能を定着させるため、クラウドIDE（統合開発環境）等の実行環境要件が不可欠である。この点は、**調査③**（2.4「生成AIコーディングアシスタント事例調査」）で把握した導入形態や機能特性を踏まえ、演習環境を教育プラットフォームの外に附属させるのではなく、教育運用として制御可能な構成要素として統合する方針を採った。すなわち、コンテンツ提示と実行環境を分離せず、“課題提示→作業→提出→評価”の導線を教育提供側が確実に通せる設計にした。

以上のような運用統制は、機能名の列挙だけで成立しない。教育提供側が日常的に操作する導線として“どの画面で何を設定し、確認し、是正するか”が画面単位で連続している必要がある。そこで、本年度の

設計では、主要機能のUI（ユーザインターフェース）を作成し、コンテンツ提示から評価・改善に到る操作の流れを具体化した。その例を以下に示す。



教育プラットフォームUI例① ダッシュボード



教育プラットフォームUI例② スキル評価基準の作成・編集

運用の入口になるのが「ダッシュボード」である。コース進捗やテストの実施状況・合格率等の運用指標を集約し、教育提供側が介入の優先順位を判断できるようにする。そして、ここで把握した状況を評価基準の設定に直結させるのが「スキル評価基準の作成・編集」である。BASISに基づく評価観点を定義・更新し、CBTや課題の採点・講評が同一基準で運用されるように評価の根拠を固定する。

さらに、その評価結果を単なる点数に留めず、どの領域が伸び、どこに停滞があるかを把握するために「スキルマップの表示」を用いる。テスト結果や提出物評価を各スキル領域に集約して可視化することで、再教育指示や教材補強の判断材料を提供する。



教育プラットフォームUI例③ スキルマップの表示

こうした運用を支える教材・テストの拡充は、「AIによるプログラミング問題の生成」によって教員が狙い（領域・難度・出題形式等）を指定した上で問題案を生成し、作問負荷の低減と品質の平準化を図る。



教育プラットフォームUI例④ AIによるプログラミング問題の生成

最後に、「受講者の取組からの教育コンテンツの分析と改善提案」では、視聴・解答・提出のログを素材にして誤答傾向や理解停滞点を抽出し、教材の補足・修正候補を自然言語で提示する（図は次頁）。これにより、“運用→評価→改善”が循環し、教育内容が更新され続ける状態を設計として担保する。

かくして、このような教育プラットフォームに関する本年度の成果物は、以上の運用要件を充すための設計書である。ここでは、対象範囲（先行実証講座の運用に必要な機能群）を明示し、次年度におけるβ構築で実装すべき機能の優先順位と、拡張時に不整合が生じないデータ設計・画面設計の骨格を確定した。

その設計書の骨子と、本教育プラットフォームの仕様を次頁以降に示す（設計書の全部は5 別掲資料に示す）。

## 講義資料（Pythonエントリーレベルテキスト）に対する改善提案

📄 提案をコピー

- [誤答パターンの分析]: 問題1（不正解率100%）は、2つの整数を受け取り、4つの算術演算（和、差、積、整数除算）の結果をタプルで返す関数を作成することを求めています。この問題で全問不正解ということは、学生が単なる四則演算だけでなく、以下の点を混同または理解不足している可能性が考えられます。

- \* \*\*関数定義と引数\*\*: `def` キーワード、関数名、引数の指定方法。
- \* \*\*算術演算子\*\*: `+`, `-`, `\*` は理解している可能性が高いですが、整数除算 (`//`) の概念と、結果が整数になるという仕様を理解していない可能性があります。
- \* \*\*タプル\*\*: 複数の値をまとめて返すデータ構造であるタプルを作成する（括弧 `()` で囲む）方法。
- \* \*\*戻り値\*\*: `return` キーワードを用いて関数の結果を返す概念。

- [講義資料との関連性]: 誤答率の高い問題1は、提示された講義資料の「第1章: 基本文法とデータ構造」の「1.1 初級レベル」に部分的に関連しています。特に「基本的な整数の四則演算」については触れられていますが、以下の点が不足しています。

- \* \*\*整数除算 (`//`) の説明\*\*: 資料では `a/b` という浮動小数点除算の例のみが示されており、整数除算 (`//`) については言及されていません。
- \* \*\*関数定義と戻り値\*\*: 資料には関数定義や `return` 文に関する具体的な説明やコード例が一切含まれていません。
- \* \*\*タプルの利用\*\*: 資料ではタプルを作成して返すという概念に触れていません。

- [具体的な改善アクション 1]:

- \*\*「1.1 初級レベル」セクションへの「関数と戻り値」の追加\*\*:
  - 関数の定義方法（`def` 関数名(引数1, 引数2, ...):）を明確に説明する。
  - `return` キーワードを用いて、関数から値を返す方法を解説する。
  - \*\*タプルの利用例\*\*: 複数の値をまとめて `return` する方法として、タプル（`('値1', '値2', ...)`）を使用する例を具体的に示す。
  - \*\*整数除算 (`//`) の説明\*\*: 浮動小数点除算 (`/`) と整数除算 (`//`) の違いを明記し、問題1の要件に合致する整数除算のコード例を含める。

## 教育プラットフォームUI例⑤ 受講者の取組からの教育コンテンツの分析と改善提案

項目	概要
コーディング規約（命名規則を含む）	ソースコードの記述ルール
機能一覧	システム上の全ての機能の詳細 ▶ <a href="#">次頁に掲載</a>
処理機能記述	各機能の処理ロジック
画面一覧	全ての画面の名称と概要
画面仕様	各画面のデザインや操作方法等
画面遷移図	各画面への移動経路
項目ラベル名一覧	入力欄や表示項目の一覧
エラーメッセージ	表示されるエラーメッセージと対処法
ファイル一覧	入出力されるファイルの一覧
ファイル仕様	入出力されるファイルのデータ構造
外部インターフェース一覧	外部システムと連携する際の通信方法・API等の一覧
外部インターフェース仕様	外部システムとのデータの送受信形式・項目や、APIの呼び出し方法
バッチ処理一覧	定期的に自動実行されるバッチ処理の名称や実行タイミング
バッチ処理仕様書	各バッチ処理の具体的な実行内容・処理ロジックや、エラー時の動作等の定義
データモデル（コレクション/ドキュメント）一覧	データベースに作成されるテーブルの一覧
データモデル定義	各テーブルの項目（フィールド）名、データ型の定義
セキュリティ仕様	セキュリティ上の脆弱性対策（XSS・CSRF等）やアクセス制御等のルール ▶ <a href="#">次頁に掲載</a>
アプリケーション仕様	ログの出力形式・文字コード・実行環境（OS・ミドルウェア）等、アプリケーション全体の共通的な技術要件

## 教育プラットフォーム設計書 骨子


受講者用画面		教員用管理画面	
ダッシュボード	公開中のコース・テスト・結果・スキルを表示	ダッシュボード	システム全体の統計情報を可視化
コース一覧	公開中の全てのコースを一覧表示	受講者管理	受講者データのCRUD (作成・読取・更新・削除)
コース受講	選択したコースを受講する	教材バンク	コース (eラーニング) 用教材のCRUD
テスト一覧	受験可能な全てのテストを一覧表示	コース管理	コース (eラーニング) のCRUD
テスト結果	全てのテスト提出履歴を一覧表示	コースプロGRESS	受講者のコース進捗状況の一覧表示
スキルマップ	スキル習熟度を可視化	スキル管理	スキル評価基準のCRUD
		問題作成	AIと手動による問題作成 (プログラミング+通常) プログラミング問題を採点サーバで検証
		問題バンク	Firestoreに保存された問題の表示・CRUD
		テストセット管理	テストセットのCRUD+公開・非公開設定
		テスト結果	全受講者の全テスト提出結果を一覧表示 テスト結果詳細とスキルマップの表示
		分析	テスト結果・講義資料をAIで分析 → 改善提案 (インサイト) を生成


### 教育プラットフォーム設計書 骨子 ▶ 機能一覧

項目	機能概要
認証 (ログイン) ルール	Googleが提供するFirebase Authenticationを使用する。 メールアドレスとパスワードによりログインを行う。
認可 (権限付与) ルール	firestore.rulesに記述したルールに基づき、サーバーサイドでのアクセス制御を実行する。 CRUD権限のうち、Create (登録)・Update (更新)・Delete (削除) は、原則として教員ロール (teacher) を持つユーザーだけに許可される (一部を除く)。 教員専用API (Cloud Functions (Callable)) でのonCall関数 (createUser (受講者登録)・generateTest (問題作成等)) は、処理の冒頭でrequest.auth.uidを使用し、呼び出し元が教員ロール (teacher) であることをfirestoreのusersコレクションに問合せ検証する。
コードのサンドボックス実行	受講者が提出したPythonコードは、executorサーバー (Google Cloud Run) 内のサンドボックス環境で実行される。child_process.exec (コマンド) は、タイムアウト (10秒) 付き、uuidで生成された一時ディレクトリ内 (隔離環境) で実行され、実行後に即時削除される。
XSS対策	フロントエンドはReactを採用しており、JSXによるDOMレンダリング時の標準的なXSSエスケープが適用される。つまり、クロスサイトスクリプティング (XSS) によるセキュリティ攻撃に対し、Reactがデフォルトで提供する自動的な防御機能が有効である。
APIキー保護	GEMINI_KEY等の機密情報は、firebase-functions/paramsとGoogle Cloud Secret Managerを使用して管理され、フロントエンドには公開されない。

### 教育プラットフォーム設計書 骨子 ▶ セキュリティ仕様

## ▼ 教育プラットフォーム仕様

アプリケーションアーキテクチャ	<b>フロントエンド (React)</b> コンポーネントベース (独立したUI部品) を組合せてUIを構築しているため、拡張性が高い。 JSX (JavaScript XML) + CSS	▶ <b>指導者:</b> ダッシュボード、受講者管理、スキル評価基準作成、問題作成、テストセット作成、教材管理、コース作成 (eラーニング)、進捗管理、テスト結果管理、インサイト分析 ▶ <b>受講者:</b> テスト受験、コース受講、テスト結果確認、スキル分析、AIフィードバック
	<b>バックエンド実行環境 (司令塔)</b>  Cloud Firestore	▶ Firebaseのデータベース機能 (Cloud Firestore) との連携  Gemini ▶ 生成AI (Gemini) とのAPI連携 (AIによる問題生成の指示、インサイト分析の指示) ▶ Cloud Run (次項) への指示 (プログラミングコードの実行と採点)
	<b>採点サーバ (Cloud Run)</b>  Cloud Run	▶ Python実行環境 (標準ライブラリ (ファイル操作・ネットワーク通信・計算等のタスク用) を含む) を利用 ▶ pandas (統計)・numpy (多次元配列) をインストール ▶ scikit-learn (機械学習)・matplotlib (グラフ描画) をインストール
プラットフォームインフラストラクチャ	<b>データベース (NoSQL) 機能</b> 非構造化データ (テキスト・マルチメディアデータ) や大量のデータ等のリアルタイム高速処理が可能。	全てのアプリケーションデータの格納 ▶ 受講者データ (ユーザー情報と役割 (教員/受講者)) ▶ スキル評価基準 (BASIS) ▶ 問題 (AI作成/手動作成)、テストセット、採点結果 ▶ コース、コース進捗データ ☞ 以上のデータをフロントエンドから呼び出す。
	<b>ユーザー認証機能</b> Googleが提供するセキュアな環境で運用。	認証データの格納 ▶ 教員と受講者のデータ (氏名・メールアドレス・役割) ☞ メールアドレス等を識別子として認証し、氏名・役割をプロフィール情報として保持する。
	<b>クラウドストレージ機能</b>  Cloud Storage あらゆるファイル形式に対応。	教材 (テキスト+マルチメディアファイル) の格納 ▶ テキスト・PDF・Office系ファイル等 ▶ 画像・音声・動画等 ☞ 格納されたデータを使用してeラーニングのコース作成に利用する。

※ Google Cloud上で  **Firebase** (認証・DB・ストレージ) 等のマネージド基盤を中核として採用

### 3.6.3 GPT-TA評価システム——生成AIを活用した教育支援機構

**GPT-TA評価システム**は、教育プラットフォーム上で発生する解答・提出物・履歴データを対象にして採点・分析・フィードバック等を行い、教育提供側の評価業務を支援する教育支援機構である。特に本教育プログラムでは、AI活用技能を“できた/できない”の二値で捉えるのではなく、BASISに基づく到達度として把握し、教育プログラムの改善へ接続することを重視する。そのため、GPT-TA評価システムは、採点処理に留まらず、評価結果をBASISに照合し、到達度の可視化と指導上の介入点の提示まで担う設計にした。

本年度事業は、【S1】 + 【S2】 前半の先行実証範囲に合せ、主にCBTの自動採点・誤答分析・到達度可視化の設計を優先した。CBTは、単に正誤を判定するだけでは、理解不足の所在 (どの概念を取り違えたか、どの判断基準が抜けたか) を教育提供側が把握し難い。そこで、GPT-TA評価システムでは、設問単位の正誤に加え、誤答の型や論点の偏りを抽出し、指導側の補強指示に直結できる情報として整形する方針を採った。これにより、教育提供側は、再視聴・再演習・再受験等の再教育を、経験則でなくデータに基づいて指示できる。

また、演習・実習・PBLの提出物評価は、次年度以降の教材拡張を見据え、提出物 (成果物) だけでなく、

作業過程の記録や根拠の記述を評価対象に含める設計にした。生成AIを介した制作は、外形的成果物だけで品質を判断し難く、再現性や妥当性の検証が難しい。そのため、提出物要件と評価観点を教育提供側が規定し、評価が属人的な印象論に流れないように、評価の骨格をシステム側で支える必要がある。GPT-TA評価システムはこの点を担保し、次年度の実装段階では、演習・実習・PBLに評価対象を段階的に拡張する。

さらに、本評価システムは、教育提供側の作問・採点負荷を低減し、同時に問題品質の均質化を図るため、作問支援機能を備える設計にした。本教育プログラムが全国の人口減少地域に展開される時、コンテンツ配信の標準化だけで教育品質は揃わない。測定・評価のための問題の品質と難易度が揺れば、到達度の比較可能性が崩れ、改善も困難になる。そこで、指導者が指定した論点やキーワード・出題形式・設問数等に基づいて問題案を生成し、指導者が最終確定する運用を想定する。確定した問題は、プリセット問題と別に指導者オリジナル問題として蓄積でき、必要に応じて試験セットに組み込めるように設計している。

ここで、本教育基盤が外部LLM (Large Language Model「大規模言語モデル」)「Gemini」(Google)と連携する意義は、単なる“便利機能の追加”でない。

第一に、自然言語による講評・フィードバックを、一定の品質と速度で供給できることである。教育提供側の講評は本来、誤りの所在と是正の方向を明示して次の取組に接続させるべきであるが、個別対応の負荷が高く、地域展開時に最も品質が崩れ易い。Gemini連携を前提にすることで、誤答・提出物・履歴データを根拠に、論点の指摘、典型的誤解の是正、次に実施させるべき再教育の提案等を一定の形式で生成できる。これは、教育プログラムの個別最適化を“受講者任せ”にするためでなく、教育提供側が必要な介入を漏れなく実施するための補助線として機能する。

第二に、作問支援における反復生成とバリエーション確保の効果がある。生成AI活用技能の評価では、同一問題の反復は安易な学習戦略を誘発し易く、問題の使い回しは短期に限界に達する。Gemini連携により、同一の評価観点を維持したまま、表現や題材を変えた問題案を生成し、指導者が採否を判断する運用が可能になる。これにより、問題品質の統制と運用負荷低減を同時に達成し、地域展開に耐える評価運用に近づく。

第三に、教育コンテンツ・シラバス改善への還流において、Gemini連携は有効である。履歴・解答・提出物のデータを集計しても、そこから改善の論点を抽出し、次年度のコンテンツ改修案に翻訳する作業は、教育提供側にとって重い。Geminiと連携することで、弱点傾向や誤答パターンを言語化し、どのコマのどの説明、課題要件や評価観点を修正すべきか、改善仮説を整理する支援が可能になる。すなわち、ログを“蓄積して終り”にせず、改善の意思決定に接続するための言語化装置として外部LLM連携が位置付けられる。

尚、Gemini連携は技術的に外部API連携であるが、ここではその詳細仕様に踏み込まない(要件定義だけ次頁に示す)。本節で重要なことは、外部LLM連携が教育提供側の評価・講評・改善という中核業務を補助し、地域展開時の品質低下を抑制する設計思想として組み込まれている点である。

## ▼ GPT-TA評価システム要件定義

1 目的	1.01 学習効果の最大化	▶ AIによる自動採点と即時フィードバックにより、受講者が自立的に弱点を克服し、効率的にスキルを習得できる環境を提供する。
	1.02 採点・評価・フィードバックの効率化	▶ AIによる採点→評価→フィードバックプロセスの自動化と効率化を図る。
	1.03 データ駆動型の教材改善	▶ 蓄積された学習データ (CBT・課題の結果) を分析し、改善が必要なシラバス・教材を特定する。 ▶ 具体的な修正案を教育提供者に提案することで、教育プログラムの継続的な改善に繋げ、受講者・教育提供者の循環的成長を有機的に促すPDCAサイクルを実現する。
	1.04 スキルの可視化	▶ スキル評価基準に基づき、個々の受講者のスキル到達度を可視化し、ポートフォリオに反映させてキャリアパス・プラン設計を支援する。
2 システム利用者	2.01 教育提供者	▶ CBT・課題等の作成と配布、受講者の学習進捗管理を行う。
	2.02 受講者	▶ CBT受験・課題提出・フィードバック受領、課題の修正と再提出を行う。
	2.03 システム管理者	▶ システムのユーザー管理・データ管理等、システム全体の運用・保守を行う。
3 システム化の 範囲	3.01 CBT・課題等実施機能	▶ CBTやプログラミング課題等を実施する機能。
	3.02 自動採点・評価・フィードバック機能	▶ AIによるCBTやプログラミング課題の自動採点・評価・フィードバックを行い、採点結果に基づいてバッジ (デジタル修了証) を発行する機能。
	3.03 スキル評価基準管理機能	▶ スキル評価基準の登録・更新・管理を行う機能。
	3.04 学習進捗管理機能 (スキルマッピング機能)	▶ 受講者の学習進捗およびスキル到達度の可視化 (ダッシュボード) やポートフォリオを作成する機能。
	3.05 教育プログラム改善提案機能	▶ 学習データを蓄積・分析し、教育提供者に対して教育プログラムの改善提案レポートを作成する機能。
4 機能要件	4.01 スキル評価基準マトリクス管理	▶ ドメイン別×段階別のスキル評価基準 (リテラシー/スキル) を定義・登録・更新できる。
	4.02 教材管理：シラバス管理	▶ スキル評価基準に紐付いたシラバスを作成・管理できる。
	4.03 教材管理：試験課題管理	▶ CBTやプログラミング課題を登録・管理できる。各教材とスキル評価基準を紐付けられる。
	4.05 学習機能：演習課題提出	▶ CBT解答やプログラミング課題のコードをシステム上で提出できる。
	4.06 学習機能：ダッシュボード (受講者用)	▶ 学習進捗・採点結果・フィードバックを確認できる。自身のスキルマップ (マトリクスの到達度) を可視化できる。
	4.07 評価・分析機能：AI自動採点	▶ プログラミング課題 (静的解析・単体テスト実行等) の他、生成AIがコードの品質 (可読性・設計・効率性) を評価し、具体的な改善点をコメントとして生成する。 ▶ CBTで正誤判定を自動で行い、採点・評価を行う。
	4.08 評価・分析機能：スキルマッピング	▶ 採点結果を自動的にスキル評価基準マトリクスにマッピングし、各受講者の到達度を更新する。
	4.09 評価・分析機能：ダッシュボード (開発者用)	▶ 全受講者の進捗状況、課題毎の平均点、スキル項目毎の到達率を分析・可視化できる。
	4.10 改善提案機能：低到達率項目抽出	▶ プログラム終盤で、全体の到達率が設定した閾値 (例：80%) 未満のスキル項目を自動で抽出する。
	4.11 改善提案機能：改善提案レポート生成	▶ 抽出された項目に対し、関連する教材・課題・受講者フィードバックのに基づき、生成AIが教材のどの部分をどう修正すべきかの提案レポートを自動生成する。
	4.12 共通機能：ユーザー管理	▶ 受講者・開発者・管理者のアカウントを管理する。
	5 非機能要件	5.1 性能・可用性
5.2 セキュリティ		▶ 個人情報保護法および関連ガイドラインを遵守する。OWASP-Top10に挙げられる脆弱性への対策を講ずる。
5.3 UI/UX		▶ 全ての利用者が直感的に操作できる、シンプルで一貫性のあるデザインを採用する。 ▶ スキルマップや分析ダッシュボードは、視覚的に判り易く、重要な情報を一目で把握できるように設計する。
5.4 拡張性・保守性		▶ 将来的に新たな教材形式 (例：対話型AI等) を追加し易いアーキテクチャにする。コードをモジュール化し、保守・改修を容易にする。
5.5 外部連携		▶ 外部LMS (学習管理システム) との連携を可能にする。

\*

本節(3.6)では、教育プログラムの地域横断での運用を成立させる条件として、教育プラットフォームとGPT-TA評価システムを教育基盤として位置付け、本年度の取組範囲(設計)とその基本方針を整理した。教育提供側がコンテンツ提示・課題運用・履歴蓄積・評価・改善等を統制できる状態を基盤側で実装できる仕様として設定したことで、次年度の $\beta$ 構築は、機能の寄せ集めでなく、教育運用の鎖を一貫させる工程として着手できる。

次年度以降の取組では、本年度に確定した設計に基づき、先行実証講座の運用を通じて実装仕様の妥当性を検証し、評価結果とログを教育コンテンツ・シラバス改善に還流させる循環を実装として成立させる。

\*

本章では、教育モデルの設計を「構造」(3.2)・「教育目標」(3.3)・「教育体系」(3.4)・「教育素材」(3.5)・「教育基盤」(3.6)の順に、上位概念から実装に向けて段階的に確定してきた。

本年度事業では、BASISによって養成人材像をスキル要件に落とし込み、教育目標として形式化した上で、7科目・60コマの教育体系としてカリキュラムを策定し、授業運用可能な粒度でシラバスを整備した。これにより、“何を、どの順序と量で、どの形態で教えるか”という教育設計の骨格は、設計情報として揃った。

その上で、先行実証範囲(【S1】+【S2】前半)について、講義・演習・自習の主な教育コンテンツを整備し、教育を属人的な実施に留めず、一定の品質で再現できる“教材としての仕様”に近付けた。さらに、教育プラットフォームとGPT-TA評価システムを設計成果として確定し、教材提示・履歴蓄積・評価・改善等を一体で運用できる教育基盤像を示したことで、次年度の $\beta$ 構築と実証は、教育モデル統合の実装段階として具体化された。

かくして、本年度の開発は“教育目標→教育体系→教育素材→教育基盤”という連鎖を切らずに遂行し、人口減少地域に展開し得る教育モデルとしての輪郭を成立させたことが成果である。

次年度事業では、本年度の設計成果を土台にし、対象科目と教材の拡張、教育基盤の $\beta$ 構築運用、実証講座を通じた評価データの蓄積と改善還流を進め、教育モデルの実装完成度を高めてゆく。



## 事業総括

### 4.1 総括の型——CIPP+OR

本章では、本書「事業概要」(1)で整理した事業の目的と前提、「調査報告」(2)における調査結果、「開発報告」(3)における開発成果の各内容を踏まえ、本年度(令和7年度)事業における取組を総括する。

その際、教育・人材養成事業の評価枠組として広く用いられる**CIPP**(Context/Input/Process/Product)の観点で総括する。CIPPは、プロジェクトを“好かった/悪かった”の印象論で整理するのではなく——

- ① (Context) いかなる環境・課題の下で……
- ② (Input) いかなる設計と資源投入を行い……
- ③ (Process) いかなる手順で実行し……
- ④ (Product) 何を成果物として生み、いかなる効果が観察されたか

という4点を因果の筋道として整理するための枠組である。

したがって、本事業の総括では——

- ① (Context) 地域課題の把握と論点の設定 → 4.2
- ② (Input) 教育モデルとしての設計投入 → 4.3
- ③ (Process) 調査と開発の接続プロセス → 4.4
- ④ (Product) 成果物と到達点 → 4.5

という4点の後に、⑤ (Outreach) 普及・共有に向けた外部発信 (→ 4.6)、⑥ (Roadmap) 次年度以降の実装と成熟化 (→ 4.7) という2観点も加え、本事業を一貫した構造として捉える。

人口減少地域における人材不足は、単に人数の問題でない。更新が滞り易い情報基盤、担い手の流出、属人的な運用の固定化等が複合し、その結果として地域企業のDXが“回らない状態”で定着していることが本質である。

この状況に対し、本事業で構築する教育モデルでは、生成AIの活用を単なる技能として教えるのではなく、教育目標・教育体系・教育素材・教育基盤を一体化し、さらに地域課題の解決にも接続することで、人口減少地域でも反復可能な人材養成の仕組みを成立させることを企図した。

## 4.2 Context——論点と課題

本事業が対象にする人口減少地域では、IT人材の不足・流出と情報基盤更新の遅滞が同時に進行し、地域産業を支えるサプライチェーン全体が技術的停滞の連鎖に陥り易い。これに加え、生成AIの急速な商用化が、開発の速度と品質の新基準を生み出したことで、地方企業は対応余力を失いつつある。かような環境下では、従来のIT教育の延長線上で“技能を積み上げる”だけでは不十分で、生成AIを実務の中で運用し、要件定義から実装・運用改善までを継続的に遂行・主導できる次世代型IT人材の育成が求められる。

しかし、人口減少地域で教育を成立させるためには、教育の内容以上に、教育の実施条件の不安定さが大きな制約になる。すなわち、指導者の確保と配置、授業実施の継続性、演習・実習環境の整備、評価の実施と記録の維持が地域毎に揺れ易く、教育の品質が属人的判断に左右され易い。そのため、本事業における論点は、次の4点を一体として設計できるか否かにある。

- 目標（到達点）を測定可能な形で定義すること
- その目標に基づいて教育の順序・量・形態を規定すること
- 指導品質を教材側で制御すること
- 運用履歴を基盤側に蓄積し、評価と改善に接続すること

そして、地域企業のDX推進は教育で完結しない。地域の実務課題を教育に取り込み、成果物によって企業課題の解決に接続する仕組みを持たねば、教育成果が地域の改善循環に結び付かず、人材が域外に流出する圧力も弱まらない。そのため、本事業では、PBLを通じて地域課題を扱い、修了後の支援も含めて地域定着を後押しする教育モデルを構想した。

かくして、本事業の“Context”は、生成AI時代における人材像を再定義し、人口減少地域の制約条件下でも教育を反復可能にする**教育モデル**を構築する必要性として整理できる。

## 4.3 Input——施策と投入

本事業では、養成を目指す人材像を**AI補完型ITエンジニア**と定め、生成AIを駆使して情報システム開発の速度と質を高めつつ、要件定義から運用改善までを自走できる能力を養成対象にした。この人材像を起点にし、本年度事業では教育モデルを実装可能な水準まで落とし込むための投入を、次の順序で設計した。

- 1) **教育目標**を抽象的理念に留めず、測定可能な形に整形するべく、**AI駆動ITスキル指標（BASIS）**を策定対象に据えた。教育目標は、教材の良し悪し以前に、何を到達点として定め、どの水準を充したと

判断するかが確定していなければ、**教育プログラム**を設計できない。そのため、本年度事業は、教育目標の骨格をBASISとして構造化し、既存の（権威性が高い）スキル標準との整合性も踏まえ、客観性と可搬性を担保する方向で投入を行った。

- 2) BASISで定めた到達点を、教育の順序・量と**教育形態**に翻訳するべく、**教育体系**として**カリキュラム**（7科目・60コマ）を策定し、授業運用可能な粒度として**シラバス**を整備する方針を採った。ここでの投入は、科目の設定に留まらず、講義・自習・演習・実習・PBLという教育5形態を明確に配列し、最終統合としてPBLが位置付けられる構造を与えた点にある。
- 3) 教育体系を現場で再現可能なものとして成立させるべく、**教育素材**（講義動画・講義資料、演習課題セット、理解度確認CBT）を整備し、指導者が一定の品質で指導できる条件を教材側に埋め込む投入を行った。ここで重要なことは、受講者の努力に依存して成果を期待するのではなく、指導者側が提示・統制・確認・是正までを運用設計として組み込む点にある。
- 4) 教育の運用を継続可能にし、履歴を蓄積し、評価と改善に接続するべく、**教育プラットフォーム**と**GPT-TA評価システム**の設計を投入対象にした。本年度事業では、開発・構築そのものでなく、次年度のβ構築に向けて設計書群を成果物として確定させることを目的にしたが、設計が机上整理に堕しないよう、運用導線を起点にして必要機能とデータの骨格を整理する投入を行った。

以上の“Input”は、教育モデルを“方針”でなく“運用できる仕様”に落とし込むための必須要素であり、単発の教材開発や単一のシステム導入では代替できない構成として整理できる。

## 4.4 Process——調査・設計・開発の連結

本年度の事業プロセスの特徴は、調査と開発が並列に存在するのではなく、調査結果を教育モデルの設計入力に連結し、その設計結果を次年度の開発・構築に連結する“翻訳工程”を明確に置いた点にある。

本年度に実施した調査群は、地域企業における生成AI活用の実態（2.2）、IT“学び直し”ニーズ（2.3）、生成AIコーディングアシスタントの事例（2.4）、AI活用エンジニアリング教育の事例（2.5）という4領域を対象にし、教育プログラムと**教育基盤**の設計に必要な前提条件を抽出する役割を担った。これにより、生成AI活用の阻碍要因や導入上の論点、実務で求められる技能の範囲、教育形態として有効な指導手順や評価の置き方が、設計上の判断材料として整理された。

この調査入力を受け、開発工程では、教育モデルを“目標→体系→素材→基盤”という順序で設計・整備した。すなわち、まづ教育目標をBASISとして構造化し、次にその目標を授業設計として確定するべくカリキュラムとシラバスを整備し、さらに授業を再現可能にするための**教育コンテンツ**（教材）群を整備し、最後に履歴化と

評価・改善を担う教育基盤を設計した。

この順序を採ったのは、教育コンテンツから作成し始めると、教育内容の粒度や難易度が属人的に決り、教育体系としての整合が後追いになり易いからである。本事業が狙うのは、人口減少地域でも同質の教育を反復可能にすることであり、“上手く教えられる人”に依存した属人的な教育では、普及と継続が成立しない。そのため、本年度事業では、設計要素間の依存関係を明示し、後段工程が前段仕様に従う形で整流化した。

また、本年度は基盤構築を完了させる年度でないが、教育プラットフォームとGPT-TA評価システムは、教育運用の導線を成立させる観点から試作と検討を行い、その結果として両者を分離した仕立てより、統合基盤として設計した方が合理的であることが確認された。この判断は、次年度のβ構築における実装範囲の整理と、ログ・評価データの一貫性確保に直接に資するものである。

以上から、本年度事業の“Process”は、調査から得た外部条件を教育設計に翻訳し、設計成果を次年度構築に接続する一連の工程として総括できる。

## 4.5 Product——成果と評価

本年度事業の成果は、システムやコンテンツを“作った”という事実のみで評価されるべきでない。本事業の本質は、人口減少地域でも反復可能な教育モデルを成立させるために、教育を仕様化し、運用と改善の循環に接続する点にある。そのため、本年度事業の“Product”は、次年度以降の構築と実証を可能にする設計成果の確定として評価する必要がある。

### 1) 教育目標に関する成果

BASISの枠組を策定し、養成対象である「AI補完型ITエンジニア」に必要な能力を体系化した。既存のスキル標準との整合性を検討することで、内部設計に閉じない客観性と可搬性を担保する方向性を確立した。これにより、教育成果の判定が恣意的評価に寄らず、外部説明可能な形式で整理される基盤が整った。

### 2) 教育体系に関する成果

7科目・60コマのカリキュラムを確定し、教育形態の配列と配分を明示した。カリキュラムは単なる科目表でなく、基礎から統合に向けた**教育段階**を規定し、最終統合としてPBLが位置付けられる構造を与えた。そして、シラバスは、授業運用と評価連動に整合する記述単位としてフォームを整備し、本年度の先行範囲について各コマの指導計画を提示できる水準まで整備した。

### 3) 教育素材に関する成果

先行範囲に対応する各種の教育素材（講義動画・講義資料、演習課題セット、理解度確認CBT問題）を整備した。

ここでの要点は、教育コンテンツを単なる配付物として扱わず、指導者が一定品質で授業を実施できるよう、“提示→作業→提出→講評”の手順をコンテンツ側に組み込む方針を採ったことである。特にCBTは、理解の不足を可視化し、再教育に接続する統制機構として位置付けたことで、基礎段階の品質保証に資する成果になった。

#### 4) 教育基盤に関する成果

教育プラットフォームとGPT-TA評価システムの設計書群を成果物にした。本年度は構築年度でないため、提供機能の実装を網羅的に語るのではなく、教育運用を成立させるための機能骨格、データの粒度、画面・導線の設計方針を整理し、次年度β構築の実装優先順位を確定した。

以上を総合すると、本年度、教育モデルを“理念”から“運用できる仕様”に転換し、部分範囲ではあるが教育コンテンツ整備まで到達した年度であったと評価できる。他方で、次年度以降の拡張と実証に向け、評価指標の精緻化、教材範囲の拡張、基盤実装とログ活用の具体化が、引き続き主要課題として残る。

## 4.6 Outreach——普及・共有に向けた外部発信

本事業は、人口減少地域の制約条件を前提にする以上、単年度の開発成果を内部に留めず、普及・共有の設計を併走させる必要がある。構築する教育モデルは、特定の現場でのみ成立する“特注品”でなく、複数地域に移植可能な構造として設計されねば、地域課題への実効的な解になり得ない。

本年度は、『事業報告書』として、調査結果(2.2~2.5)と開発成果(3.3~3.6)を章立てで体系化した上で、教育目標(3.3)・教育体系(3.4)・教育素材(3.5)・教育基盤(3.6)の関係を追跡可能な形で整理した。これは、次年度以降の実証結果を追記するための基盤になるのと同時に、他校・他地域に移植する際の参照枠組としても機能する。そして、成果報告会の開催や本報告書の配布、ウェブサイトでの公開を行うことで、成果の外部共有を計画に組み込んだ。

また、BASISの整備やその既存スキル標準との照合は、教育モデルの普及にとって重要な意味を持つ。教育モデルが各校の独自構想に閉じてしまうと、他機関には再利用し難く、説明責任も果し難い。他方で、既存標準との接続を意識して教育目標と評価基準を整備すれば、教育成果の説明可能性が高まり、外部との合意形成も進め易くなる。

このように、本年度事業の“Outreach”は、外部発信自体を目的化するのではなく、教育モデルの可搬性を高めるための整理・標準接続・公開計画として位置付けられる。

## 4.7 Roadmap——次年度以降の実装と成熟化

本事業は、本年度の成果によって完結する構造でない。本年度は寧ろ、教育プログラムを仕様化し、教育コンテンツを部分範囲で整備し、次年度のβ構築と実証に移行するための“踏み台”を築き固めた年度である。その上で、次年度以降の重点は次の3点に集約される。

- 1) 教育目標と評価指標の完成度を高める。BASISは枠組として成立したが、実証の深化に応えるべく、各要素の定義粒度・到達水準・評価基準の整備が必要である。その際、評価に資する（教育コンテンツの問題・課題の品質と難易度の管理は、教育品質保証の中核である。CBTや演習課題では、設問（評価の対象になる問題）の品質が不安定であれば、到達度の判定が揺れ、指導の是正が機能しない。そのため、次年度は、設問の妥当性・難易度の校正、BASISへの紐付け、誤答分析に基づく改善等の編集工程を制度化し、評価を“仕組”として安定させる必要がある。
- 2) 教育基盤のβ構築と実証運用を進める。本年度の成果物は設計書群であり、次年度に実装・構築を行うことで、教材提示・課題提示・提出・採点・講評・改善の導線が実運用として成立する。ここで重要なことは、教育基盤を単なるITシステムとして構築するのではなく、教育運用の統制機構として設計意図を保ったまま実装することである。ログの蓄積は目的でなく、指導の是正、コンテンツ改善、評価の精緻化に接続して初めて意味を持つ。
- 3) 生成AI連携の教育的意義を、実証データに基づいて具体化する。外部LLM（Gemini）との連携は、単に自動化のためにあるのではない。評価結果の自然言語フィードバック、誤答・誤解の傾向分析、問題・課題の作成支援、改善提案の提示等、教育提供側の意思決定を支える“教育支援機構”として位置付けられる点に意義がある。換言すれば、ここで想定する「個別最適化」は、受講者の任意性に委ねた学び方の自由でなく、教育提供側がログと評価データに基づき、再教育の条件分岐、提示順序、課題の再提示、講評の重点化を調整し、指導の確実性を高めることを意味する。次年度以降の事業では、この調整が実運用としてどの程度有効に機能するかを検証し、設計に反映する循環を確立せねばならない。

以上のロードマップにより、本事業は、本年度の設計成果を足場に、次年度のβ構築・実証、次々年度の拡張と普及に段階的に移行してゆく。人口減少地域における教育の成立条件は厳しいが、教育を仕様化し、運用を履歴化し、評価と改善に接続することで、地域制約を超えて同質の教育を反復可能にする道筋は明確になりつつある。本年度の成果を確実に引き継ぎ、次年度は実装と実証の段階に進め、教育モデルの完成度を高めてゆく方針である。

\*

本章では、本年度事業の取組を整理し、初年度成果の中核が“設計の確定”と“次年度実装への接続条件の明文化”にあることを確認した。人口減少地域で教育を展開する場合、教材作成や授業実施の工夫のみで品質と

再現性を担保することは難しく、教育提供側が“提示・回収・評価・改善”を一連の運用として統制できるよう、到達目標と運用仕様を先に整流化しておく必要がある。本章は、この要請に対し、BASIS・教育プログラム・教育基盤という複数層の設計成果を、相互の関係性も含めて総括したものである。

1 「事業概要」で見たように、本事業が直面する環境は、“人口減少による担い手の縮小”と“生成AI普及による速度・品質の新基準”という二重の圧力である。

2 「調査報告」では、地域における生成AI活用の現状、IT“学び直し”の需要、生成AIコーディング支援の導入形態、AI活用エンジニアリング教育の事例を把握し、AI時代の職業教育で重視すべき中核が“生成の巧さ”でなく、選定・統制・検証を反復できる作法、そして根拠・差分・ログ等を“証拠”として残して説明責任を支える設計であることを明示した。

3 「開発報告」では、これらの知見を教育設計に翻訳し、BASISの枠組、7科目・60コマのカリキュラム、授業運用可能な粒度のシラバス、先行実証範囲の各種教育コンテンツ、さらに教育プラットフォームとGPT-TA評価システムの設計書群を整備した。

これにより、教育提供側が「何を、どの順序と量で、どの形態で教え、何を以て“できた”と見做すか」を基準化し、教育プログラムを属人的運用に依存させずに地域に展開し得る輪郭が成立した。

次年度事業では、本年度に確定した設計成果を土台にし、対象科目・教材の拡張と教育基盤のβ構築を並行して進め、実証講座を通じてログと評価データを蓄積し、BASIS・シラバス・教材に改善を還流させる循環を実装として成立させる段階に入る。特に外部LLM（Gemini）連携を含むGPT-TA評価システムは、CBT自動採点や提出物講評の自動化に留まらず、作問支援・フィードバック生成・到達度可視化を通じ、教育提供側の設計・運用・改善を支える〈教育支援〉として機能させることが重要である。設計から実装へ、仕様から運用へと歩を進め、人口減少地域でも“同質の教育”を継続的に提供し得る教育プログラムとして完成度を高め、地域産業のDXを担う人材が循環的に生れ、地域に定着してゆく状態に確実に接続する。



# 別掲資料

## 5.1 教育コンテンツ——講義資料

【科目S1】生成AIリテラシーとAI倫理  
01 生成AIの基礎と安全利用の前提

### 生成AIは何か／LLMの概観

生成AIの基礎と安全利用の前提

#### 2つのAIの違い

- ✓ 識別系AI (従来) データが何かを判定する「審判」。
- ✓ 生成AI (現在) 新しいデータを生み出す「クリエイター」。



#### LLM (大規模言語モデル) の何が「大規模」?



データ

ネット上の膨大なテキスト、書籍、コード。



パラメータ

数千億個以上の結合により複雑な文脈を理解できる。



計算資源

数千台のGPUと膨大な電力が必要。

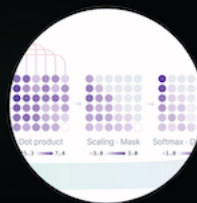
#### LLMの仕組み

##### 次のトークン予測する

- ✓ 文章を「トークン」に分解
- ✓ 統計的確率で「次に来る言葉」を計算
- ✓ 例: 「吾輩は猫である。名前は○○。」  
→「まだ無い。」



#### トランスフォーマー



2017年に登場した  
深層学習モデルの革新的技術。

入力文中のどの単語が重要かを動的に判断し、学習する。  
長い文章の複雑な関係性を効率的に捉えて処理できる。

#### 知るべきリスク

LLMは真実ではなく  
統計的に次に来る確率が高い言葉  
を出力する

そのため、ハルシネーション (もっともらしい嘘) が必ず発生してしまう。

### 検索と生成の違いと使い分け

生成AIの基礎と安全利用の前提

## 検索と生成AIの根本的な違い

 <b>検索</b>	 <b>生成AI</b>
図書館型	料理人型
既存のページ (URL) を探し出す。	知識という材料から新しい回答を作る。

## 検索と生成AIの使い分け

<b>1. 正確性</b> 検索: 高い (一次情報) 生成AI: 不確実 (嘘をつく)	<b>2. 情報の鮮度</b> 検索: リアルタイム 生成AI: 過去の学習データ
<b>3. 得意分野</b> 検索: 事実確認 生成AI: 要約、コード生成	<b>4. 出力</b> 検索: リンク集 生成AI: 答えそのもの

## 現場での使い分け

<b>検索を使う時</b>  最新ライブラリの仕様 エラーコードの原因特定  通信障害などの最新情報	<b>AIを使う時</b>  コードの要約・変換  アイデアの壁打ち  中学生にもわかるように説明して
--	--

## 「検索と生成」のハイブリッド活用術

- 01 生成AIで「全体像」**  
構造を提案させ、思考をショートカット。
- 02 検索で「裏付け」**  
最新の公式ドキュメントで仕様を確認。
- 03 生成AIで「実装」**  
確認した仕様に基づきコードを書かせる。

AIは自信满满に  
**嘘**をつくことがある。

「AIが言ったから」は通用しません。

鉄則: 必ず検索でファクトチェック!

## プロンプトの基本形と 目的の提示の仕方

## 大原則: GIGO

Garbage In,  
Garbage Out

「ゴミ」を入力すれば、「ゴミ」が出てくる。  
指示 (プロンプト) の質が、回答の質を決める。

## 「構造化」された指示へ

単なる「質問」ではなく、3つの要素で定義する。




<b>1. 目的</b> 役割とタスクの定義 「誰として」振る舞うか	<b>2. 条件</b> 制約とコンテキスト 範囲やトーンを限定する	<b>3. 形式</b> アウトプットの型 Markdown, JSON等
--	--	---

## 要素1 & 2: 目的と条件

<b>目的 (役割とタスクの定義)</b> 「あなたはシニアJavaエンジニアです」  役割を与えることで、 専門的な視点や用語選びを引き出す。	<b>条件 (制約事項とコンテキスト)</b> 「Java 17, Spring Bootを使用」 「初心者向けに」  技術スタックやトーンを指定し、 回答の精度を高める。
--	---

## 要素3: 出力形式

**出力形式 (アウトプットの型)**  
回答を「どこで使うか」に合わせて型を指定する。

 Markdown形式の表で	 プログラムで読み込めるよう JSON形式で など	 コピーしやすいように コードブロックで囲んで など
--	---	--

## 大切なマインドセット



### プロンプトは微調整で改善する！

一発で完璧な回答を出すことを目指すのではなく、対話を通じて「微調整」を繰り返して改善するのがプロのコツ。

## AIを使いこなす側へ

「目的・条件・形式」の型を持ち、エンジニアとしての武器にしよう。

## 初期設定と安全利用の基本

生成AIの基礎と安全利用の前提

## オプトアウトの設定

- ✔ デフォルトは「AIの学習に再利用」
- ✔ 秘密のコードやアイデアが、将来「他人の回答」として流出するリスクがある。
- ✔ 学習利用OFFを徹底しよう！



## 入力禁止事項 (絶対に送信してはいけない3つの情報)



## 安全利用のための運用手順

組織でAIを利用する際は、個人の判断ではなく「決められた手順」に従います。

### 01 利用ツールの限定

会社や学校が許可したツール（学習されない環境）のみを使用する。

### 02 最終確認の義務

AI生成物は必ず人間がテスト・校閲する。AIは魔法の杖ではなく、責任は人にある。

### 03 記録の保持

プロンプトと出力ログを保管する。業務判断の透明性を確保するために推奨。

## プロの作法と責任

**抽象化・匿名化**  
AIに相談したい時は、必ず「ダミーデータ」に置き換えること。

**最終責任は人間**  
AIは魔法ではない。  
テスト・校閲は人間の義務。



## AIは進化し続けるが 原則は変わらない

「情報の機密性を守る」  
「出力の責任は人間が負う」

## 第1回講義まとめ

### 1. LLMの正体

次の言葉を予測する確率マシン  
意味は理解していない

### 2. 使い分け

事実の確認→「検索」  
構成や変換→「生成AI」

### 3. プロンプト

目的・条件・形式を明確にし  
高品質な出力を目指す

### 4. 安全利用

学習OFF設定(オプトアウト)  
機密情報の入力禁止の徹底

## Next: 検証テクニック

AIの「嘘」を見破る実践編へ

【科目s1】生成AIリテラシーとAI倫理  
02 誤りの起こり方と確かめ方

誤りの型の基本  
混同・作り話・過度な一般化等


ハルシネーションとは

言葉を紡ぐマシン

AIは事実を記憶しているわけではありません。単に「文脈に合う言葉のつながり」を確率的に計算しているだけです。そのため、事実が不明な場合でも、確率的に「ありそうな」言葉を繋いで、もっともらしい嘘をついてしまいます。



3つの「誤り」の型

 <b>1. 作り話</b> 存在しない関数や法律を勝手に発明してしまう「純粋な幻覚」	 <b>2. 混同</b> AとBの情報を混ぜて不正確な事実を作る	 <b>3. 過度な一般化</b> 一部の解決策を全体に当てはめる
--	--	--

1. 作り話

AIは「いかにもありそうな名前」を確率的に生成します。存在しないライブラリ関数や架空の書籍を自信満々に提案してくるのは、似たパターンの学習データが大量にあるためです。



2. 混同

情報の混合

異なるバージョンの仕様や、似たプロジェクトの情報が混ざって出力されます。

例：React v16の書き方が、v15の解説に混入する。

原因

AIの中で「情報の区別」が曖昧なまま、「関連性が高い」という理由だけでひとまとめに引き出されてしまうことが原因です。



3. 過度な一般化



メジャーな情報の圧力

「Windowsだけで有効な解決策」なのに、あたかも全環境で使えるかのように断定してしまうケースです。出現頻度の高いメジャーな情報が、マイナーな条件や文脈を押しつぶして出力されてしまいます。

エンジニアの鉄則

AIの回答は

「検証が通るまでは、ただの文字列」  
一歩引いた検証姿勢が、信頼性を守ります。

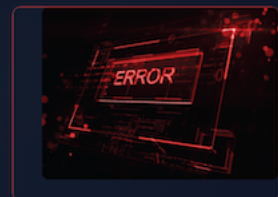
確かめ方の基本技術  
出典確認・複数照合・実行検証

未検証の恐怖

AIの回答をそのまま業務で使うことは、エンジニアにとって**自殺行為**に等しいです。

テスト（検証）を一度もしないで、AIの出力をそのまま本番環境にデプロイする

それと同じくらい、恐ろしく危険な行為だと認識してください。



## プロが実践する3ステップ

- 1. 一次情報**  
公式サイトやドキュメント情報の「元ネタ」を確認
- 2. 複数照合**  
複数のAIモデルや検索エンジンで再取り
- 3. 実行検証**  
サンドボックス環境で実際にコードを動かす

## Step 1: 一次情報への到達

必ず自分自身で「公式サイト」や「公式ドキュメント」を確認してください。

- GitHub リポジトリ
- AWS / Google 公式ドキュメント
- RFC (技術標準)

⚠️ ※AIが提示するURLは架空の可能性もあります。

## Step 2: クロスリファレンス

### 情報を重ね合わせる

マルチモデル検証: ChatGPT, Gemini, Claude等と同じ質問を投げる。

検索エンジンや技術情報サイトでの検証: キーワードをGoogleやQjitaで再検索する。

回答がバラバラなら、そこに「罠」が潜んでいる可能性があります。⚠️

## Step 3: サンドボックスでの実行検証

エンジニアにとって、動くものが真実です。

本番環境に移行する前に、必ず隔離された「サンドボックス」で実行し、エラーや挙動を確認してください。

- ⚠️ 動く = 正しいとは限りません。
- ⚠️ セキュリティや効率性も自身の目でレビューが必要です。

## 3つの重点チェック項目

- バージョン番号**  
古い情報と新しい情報が混ざっていませんか?  
廃止された機能を提案されている可能性があります。
- 数値・統計**  
桁の間違いや、単位の入れ替わりに注意。  
確率は計算できても、意味は理解していません。
- 固有名詞・API引数**  
関数の引数の順番が逆ではありませんか?  
「もっともらしい間違い」の常連です。

## 最後の一线を守る

品質を保証するのはいつだって「人間」の仕事

AIは道具であり、責任者ではありません。

## 言い切り表現の点検と書き直し

## AIの癖: 自信満々な言い切り (断定表現)

AIは「親切なアシスタント」として振る舞うよう調整されています。

そのため、「よくわかりません」と言葉を濁すよりも、真偽にかかわらず「OOが正解です」とはっさり答える傾向があります。(自信があるから正しい、とは限らない)

## 警戒すべきキーワード

- 間違いありません
- 100%安全です
- 完璧に動作します

これらは全て「書き直すべきポイント」の可能性が大きい。

## プロの技: 断定回避 Hedging (ヘッジング)

- AIが出力した言い切り表現を、事実に基づいた「誠実な表現」に書き換える技術です。
- ITの世界に「完璧」はありません。リスクを適切に表現しましょう。

## 書き換えの3つのコツ

### 1. 条件を付加



1. 条件を付加  
「〇〇という前提では」  
範囲を限定する

### 2. 根拠を明示



2. 根拠を明示  
「公式資料によれば」  
情報ソースを明示する

### 3. 可能性を残す



3. 可能性を残す  
「傾向があります」  
「考えられます」  
含みを持たせる

## 実践例 1：脆弱性

### AI (Bad)

「このプログラムに  
脆弱性はありません」  
責任を取りきれない断定

### Pro (Good)

「現時点の静的解析の範囲内では、  
既知の脆弱性は検出されませんでした」  
条件と根拠を明示

## 実践例 2：エラー原因

### AI (Bad)

「エラーの原因は  
メモリ不足です」  
断定すると他の可能性を  
捨てることになる

### Pro (Good)

「ログの傾向からメモリ不足が疑われます  
が、詳細な特定には追加調査が必要です」  
可能性と次アクションを提示

## なぜ慎重になるのか？

AIの言葉をそのまま対外的な文書や報告に載せることは、  
その間違った**全責任を「会社」が負うことを意味します。**

後でバグが見つかった時、  
「AIがそう言ったから」はプロの言い訳として通用しません。

## 大切なマインドセット

「わからない」を正しく伝えること



正直な報告も、立派な  
リスク管理能力です。



確証が持てない時は、  
無理に断定せず  
上司へエスカレーション  
しましょう。

## 信頼されるエンジニアへ

AIに振り回されるのではなく、  
その出力を自分の責任でコントロールする。  
それがプロフェッショナルの流儀です。

## 自動化の限界と 人間による最終確認

## 自動化のパラダイムシフト

### 従来のAI

### ルールベース

「AならばB」の定義に従う。  
テストによりコントロール可能。

### 生成AI (LLM)

### 自ら判断・生成

大量データから学習したパターンに基  
づき、自ら判断・生成する（確率で動  
く）不確実なシステム。  
100%の制御は不可能。

## Human-in-the-loop (人間がループに入る)



## 自動化の境界線

失敗の許容度とリスクに基づく判断基準

### 自動化 OK

失敗してもすぐに修正が効くもの

- ✓ 個人の学習用コード作成
- ✓ アイデア出しの壁打ち
- ✓ 社内ドキュメントのドラフト

### 自動化 NG

取り返しがつかないもの

- ▲ 顧客へのメール自動送信
- ▲ SNSへの自動投稿
- ▲ 検証なしの本番デプロイ

## 最終責任は「利用者」にある

「AIが言ったから」は、プロの世界では通用しません。

▲ 著作権侵害 ▲ セキュリティ事故 ▲ 誤情報...

**全ての責任はあなたと会社が負います。**

## 第2回講義まとめ

- 疑う力**  
ハルシネーションを見破る
- 確かめる技術**  
一次情報を実行検証する
- 整える作法**  
不確実なものを誠実に伝える
- 守る仕組み**  
人間が最終責任を持つ

## Next: 倫理・権利・プライバシー

【科目s1】生成AIリテラシーとAI倫理

### 03 倫理・権利・プライバシー

## 個人情報と機微情報の取扱い

### 入力のレッドライン（超えてはいけない一線）

## なぜ入力制限が必要？

- 多くの生成AIサービスでは、入力データがサーバーに送信・保存されます。
- 一度送信したデータは、AIの学習に使われる可能性があり、自分のPCからファイルを消すようには**削除できません**。

## 情報の「格付け」3分類

① 個人情報	② 機微情報・機密情報	③ 一般情報
個人を特定できる情報 原則禁止	漏洩＝損害 絶対禁止 (Red Line)	公開ドキュメント等 入力可能
<ul style="list-style-type: none"> <li>氏名、生年月日</li> <li>住所</li> <li>個人のメールアドレス</li> <li>マイナンバー</li> </ul>	<ul style="list-style-type: none"> <li>認証情報</li> <li>未公開ソースコード</li> <li>非公開の会議資料</li> </ul>	<ul style="list-style-type: none"> <li>公開ドキュメント</li> <li>ホワイトペーパー</li> <li>一般公開情報</li> </ul>

## ① 個人情報

特定の個人を識別できる情報です。

- 氏名、住所、生年月日
- 個人のメールアドレス
- マイナンバー

**TEST DATA WARNING**

テストデータに本物の顧客名簿の一部などを混し込むのは重大なコンプライアンス違反になる可能性があります。

## ② 機微・機密情報

### Red Line

漏洩すると組織に多大な損害を与える情報です。これらを送るのは、家の鍵を道端に置くのと同じだと思ってください。

<b>認証情報</b> (パスワード、APIキー) PASSWORD: ***** API KEY: xxxxx-xxxxx-xxxxx	<b>未公開コード</b> (独自のアルゴリズム) def secret_algorithm(): ...	<b>非公開資料</b> (議事録、仕様書) 議事録.docx 仕様書.pdf
---	--	--

### ③ 業務上の一般情報



- ・公開されている技術ドキュメントの要約
- ・一般的なプログラムに関する質問など

**入力可能ですが、組織のルールを確認しましょう。**

### 技術1：匿名化

固有名詞を意味のない記号に置き換えます。



### 技術2：抽象化

機密部分を排除し、純粋な「技術的課題」に変換します。



### 迷った時の自分への質問

このプロンプトを会社の掲示板や  
SNSに貼り付けて全世界に見られ  
ても大丈夫か？

一瞬でも「まずい」と思ったら、入力禁止。

### まとめ

- 🔍 区別する (個人・機微・一般)
- 🔧 加工する (匿名化・抽象化)
- 👉 判断する (入力しない勇氣)

## 著作権・ライセンスの基本と引用

AI出力の「権利」を守る知識

### 法的リスクの存在

AIが生成したコードや文章には、  
目に見えない「著作権」のルールが関わります。

これらを無視すると、

**製品の公開停止や法的トラブル**に発展する恐れがあります。

### AI生成物の著作権

#### 自動生成物



人間がほぼ手を加えていないもの  
原則：著作権なし (誰でも自由に使える)

#### 創作的寄与



プロンプトで指示し、大幅に修正したもの  
原則：著作権あり (あなたの著作物になる)

### 著作権侵害の判断基準



⚠️ 意図せず著作権侵害に問われる可能性があることに注意！

### ライセンスの落とし穴



## 対策: フィルタリング

GitHub Copilot等の開発支援AIを使う際の必須設定です。

公開コードに一致する提案を推奨する  
→無効にして意図しない侵害を防ぐ。

## 引用の4要件 (著作権法)

- 公表された著作物であること  
未発表のものは引用不可
- 引用の目的上「正当な範囲内」であること  
報道・研究等の目的で必要最小限
- 公正な慣行に合致すること  
社会通念上、妥当な方法で引用されていること
- 主従関係の明確性  
自分の著作物が「主」、引用が「従」という質的・量的バランス

## 出典表示の具体例

ドキュメントの場合

- ツールの使用方法を明記
- プロンプトと回答文の記載
- 明確な区別
- 出典情報の記載

具体例

AI Response Citation Record

「Gemini3を使用し、「生成AIの特徴を3つ挙げる」というプロンプトに対して、「1. コンテンツの新規生成能力、「2. 自然言語による直感的な操作」、「3. 多領域への汎用性と創造性の補充」(Gemini 3, 2026) という回答を得た。

## 出典表示の具体例

コードの場合の表示例 (プログラムコード内のコメントアウト)

```
JavaScript
/**
 * [引用・参考]
 * 引用元: [リポジトリ名/プロジェクト名]
 * 著作者: [作成者名]
 * ライセンス: MIT License (https://opensource.org/licenses/MIT)
 * 取得元URL: https://github.com/username/repo/blob/a1b2c3d4/path/to/file.js#L15-L30
 * 著作権表示: Copyright (c) 2024 [作成者名]
 * 備考: 元のコードをベースに、エラーハンドリング処理を追加して最適化。
 */
function originalLogic() {
  // 引用・変更したコードをここに記述
}
```

## 到達目標の確認

AIが生成したコードにはリスクがある。

- ✓ 公式ドキュメントで確認する
- ✓ 出典を正しく明記する

AGILE DEVELOPMENT CYCLE

Requirements → Analyze → Deploy → Release

## 共有時の注意と社内外の線引き

情報の「出口」を管理する責任

## 「シャドウAI」の危険性

公私の線引き

個人の無料版AIで業務を行うことは、意図せず「機密を持ち出す」ことと同じです。

入力内容は学習に利用されるリスクが高いため、必ず組織が指定した環境で行いましょう。

PRIVATE / 組織の環境

SHADOW AI / PERSONAL CLOUD

公私の境界線 (Public/Private Boundary)

## 組織内での「知恵」の共有

推奨

プロンプトの共有

「どう指示したか」というプロセスを共有することで、チーム全体の生産性が上がります。

危険

生の回答を丸投げ

検証なしで共有するのはNG。  
必ず「自分の検証結果・意見」を添えてください。

## 社外・SNSへの発信と「法的・倫理的責任」

「AIが書いた」は言い訳になりません。

権利侵害や誤情報による不利益など、その全ての法的・倫理的責任は、発信したあなたが背負うことになります。(自分の成果物として公開する覚悟が必要)

## 公開前のチェックリスト

- 匿名化  
会社名、同僚名、プロジェクト名は削除したか?
- 権利関係  
既存ライブラリや他人のプログラムと酷似していないか?
- 検証済みか  
実際に動作し、脆弱性がないか確認したか?
- 出典表記  
「AIを活用して作成した」事実を明記したか?

## コミュニケーションのマナー



### 誠実さ

相手はAIのテンプレではなく  
あなたの意見を聞きたい



### 自分の言葉

AIの下書きをそのまま送らず  
必ずリライトする



### 配慮

受け取る側の気持ちを考え  
表現を調整する

## 到達目標の確認

- ❌ 個人のアカウントを業務に使わない
- 👤 社外発信時は自己検証を徹底する
- 🔗 AI利用の出典を明示する

## 迷った時の相談と記録の残し方 エンジニアとしての「自分」を守る防護柵

### 「たぶん大丈夫」 は危険信号

- 🕒 生成AIの技術や法規制は変化し続けています。
- ⚠️ 昨日までの「グレー」が、今日から「禁止」になることも珍しくありません。
- 👤 最も危険なのは「自分一人の思い込み」で進めることです。
- 🛡️ そのため、IT業界では「証拠（エビデンス）」を残すことが、自分自身の身を守ることに繋がります。



### 「独断」を避け、「相談」をフローに組み込む



迷ったら、組織内の「相談窓口」を活用しましょう。  
相談することは、**プロとしての防衛策**です。

### あなたを守る「証拠（エビデンス）」としての記録

後から「なぜその判断をしたのか？」と問われた時、あなたを守るのは「**証拠 (Evidence)**」だけです。  
AIが他人の権利を侵害しても、適切な手順を踏んでいたという「**潔白の証明**」になります。



### 記録すべき4つの項目



### 迷った時の3ステップアクション



### 最終的な責任のありか

AIはあなたの代わりに責任を取ってくれません  
記録を残し、透明性を確保すること  
これがエンジニアの「誠実さ」であり「防衛術」です

### 第3回講義まとめ

#### 👉 入力・出力

機微情報は匿名化・抽象化  
著作権侵害リスクを自覚する

#### 👉 共有・相談

社内外の境界線を守る  
独断せず、記録を残す

## Next: 組織内AIポリシーと品質評価観点

現場配属の初日から正しく動くために  
実際のルールを読み解く総まとめ

【科目S1】生成AIリテラシーとAI倫理

### 04 組織内AIポリシーと品質評価観点

## 組織内AIポリシーの最小構成

プロの成果物としての品質を担保する

### あなたを守る 「安全柵」

AIポリシーはあなたの自由を奪うものではありません。

「ここまでは安全、ここからは危険」という境界線を示し、皆さんをトラブルから守るためのものです。

ITエンジニアとして守るべき「憲法」だと思ってください。

### AIポリシーの5つの必須要素



これらはグローバルなAI倫理指針にも基づいた、プロのITエンジニアが守るべき「憲法」のようなものです。

### 1. 公平性と非差別

AIは学習データの偏りを引き継ぎます。

#### エンジニアの視点

データの選定段階から「このデータに偏りはないか？」と疑い、多様な背景を持つ人々に配慮します。

### 2. 透明性と説明可能性

「ブラックボックス化」を防ぎます。

#### エンジニアの視点

「AIがこう言ったから」で済ませず、その判断根拠を人間が理解できる形で説明する責任を持ちます。

### 3. 説明責任とガバナンス

AIは責任を取ってくれません。  
最終責任は開発する人間にあります。

#### エンジニアの視点

リスク管理のための監視ルールを整備し「人間がコントロール（監督）している」体制を維持します。

### 4. プライバシーとデータガバナンス

データのライフサイクル全体を管理します。

#### エンジニアの視点

モデルインバージョン攻撃（学習データ復元）等のリスクを防ぎ、不正データの混入を管理します。

## 5. 安全性、セキュリティ、堅牢性

サイバー攻撃からAIシステムを防御します。

### エンジニアの視点

プロンプトインジェクションや予期せぬ動作による事故を防ぐため、常にセキュリティアップデートと脆弱性対策を行い、システムの安全性を担保します。



## エンジニアの心得

「知らなかった」では済みません。

配属されたら

まずはAIポリシーを読み込む

## 品質観点の基本 (事実性・一貫性・再現性等) ITエンジニアが見るべき「プロの基準」

## AIの生成物に求められる品質

### 趣味でAIを使う場合

「面白い」かどうか済む世界。

### プロが仕事でAIを使う場合

実用に耐え、安全で、価値があるか。  
客観的な品質の評価が必要。



## ITエンジニアが点検すべき品質の3大指標

### 1. 正確性

正しく動くか  
事実即しているか

### 2. 安全性

害がないか  
攻撃に強い

### 3. 効率性

速いか  
安いか  
満足できるか

## 1. 機能的正確性と事実整合性

### エンジニアの「腕」

AIのコードが「それっぽく」ても、エラーが出れば欠陥品です。



シンタックス  
エラーがないか



意図した  
ロジックか



事実関係に  
矛盾はないか

## 2. 安全性、堅牢性および倫理的整合性

### エンジニアの「守り」

技術的に正しくても、  
倫理的にNGなら「欠陥品」です。  
攻撃に対する堅牢性も重要です。

### Check Points:

- 有害な表現が含まれていないか
- プロンプトインジェクション対策



## 3. 運用効率性とユーザー体験

### エンジニアの「価値」



速度  
待ち時間は許容範囲か？  
(Latency)



コスト  
予算内で運用可能か？  
(Cost)



体験  
形式は適切か？  
(Satisfaction)

## 品質を「数値」で管理する姿勢

「なんとなく良い」ではなく、SLA (品質基準) を設定します。



### 正確性

95%以上

テスト通過率



### 応答速度

2秒以内

生成完了まで



### 記録

基準を満たした証拠を残す

## 最後の砦は「あなた」

どんなに優れたAIを使っても、

品質保証を行うのは人間です。

# 作業記録と承認の流れ

## プロとしての「証明」を残す技術

### ① なぜ記録が必要なのか？

記録を残す理由は、単に「上司に報告するため」だけではありません。



### ② 作業記録 (AIログ) の5つの必須項目



### ③ プロフェッショナルな「承認フロー」



### ④ 「例外」への向き合い方



### 透明性が信頼を作る

AIのブラックボックスを、  
**人間のプロセスで透明にする。**  
それがAIに支配されないプロの姿です。

## 見直しと改善の小さな循環

### AIと「共に成長する」ITエンジニアへ

### AI活用は「運用」が本番

AI導入はゴールではありません。  
AIの進化速度はOSやフレームワークの更新より遥かに高速です。  
一度作ったプロンプトやルールに固執せず、  
小さなPDCAサイクルを回し続ける姿勢が不可欠です。

### ① プロンプトのバージョン管理とブラッシュアップ



### ② 「失敗事例」という名の資産共有



### ③ 現場からポリシーを動かす

ポリシーは不変の「岩」ではありません。

#### ボトムアップ提案

「検証の結果、安全策を講じれば導入可能」といった技術的根拠に基づき、エンジニアから組織へ提案しましょう。



AIは「道具」であり  
皆さんは「指揮者」である

AIを調律し続けるのは、  
人間の役割です。

### 全4回講義の軌跡



AIという武器を手に

「AIに代わられる人」ではなく、  
「価値を生み出す指揮者」へ。

## 【科目S2】 プロンプト設計とLLM活用 01 LLMの仕組みとプロンプトの基本形

## LLMの生成の流れ 確率生成・トークン

### プロンプトエンジニアリング

- 1 前回の講義シリーズでは、
  - ・生成AIの基礎構造・限界・社会的影響
  - ・リスク評価と倫理的利用判断
  - ・組織内AIポリシーなどについて学びました。
- 2 今回の講義シリーズでは、LLMを自在に操り、高品質なシステムを生み出すための、**プロンプトエンジニアリングの世界**に入ります。



### ① トークン：AIが理解する「最小単位」

LLMにとっての最小単位は、「文字」でも「単語」でもなく、「トークン」と呼ばれる断片です。



### 日本語トークンの数え方

**英語**

短い単語：1単語と1トークンで計算  
長い単語：複数のトークンに分割  
例: apple → 1トークン  
friendship → friend + ship (2トークン)

**日本語**

漢字1文字と1〜2トークン  
ひらがなも複雑に分割される。  
文字の種類（漢字・ひらがな・カタカナ）が多いため、同じ意味の文章でも、英語よりもトークン数が膨らむ傾向がある。

### トークンの重要性 1：情報量の限界

#### コンテキストウィンドウ

LLMが一度に記憶・処理できる情報の量は、トークン数で決まります。

無駄に長いプロンプトは、重要な情報を押し出してしまうリスクがあります。



## ① トークンの重要性 2: コスト

### 従量課金制

API利用料の多くは「1,000トークンあたり数円」という体系です。  
冗長なプロンプトは、そのままコストの増大に直結します。



## ② LLMの生成の仕組み (トークンの予測と自己回帰)

LLMの生成は、驚くほど地道な「繰り返し作業」です。



## ③ 確率の計算: ソフトマックス

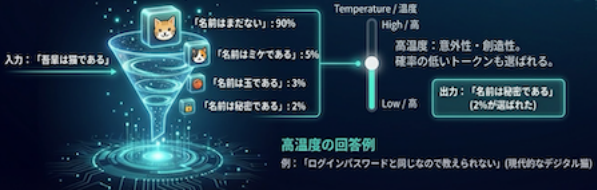
### 確率分布と温度による制御

#### AI内部の確率分布

LLMは、次に続く可能性のあるすべてのトークンに対して「確率」を割り振ります。

#### エンジニアの視点: 温度 (Temperature)

「温度」は、この確率分布をどう扱うかを決めるパラメータです。



## なぜAIの出力には「揺れ」があるのか

### 確率のゆらぎ

LLMの生成は確率に基づいているため、常に同じトークンが選ばれるとは限らない。そのため、同じプロンプトを入力して、微小な確率の差で全く違う文章が生成されることがある。



## なぜAIは「嘘」をつくのか

### AIの出力は事実ではなく確率に基づくから

AIは事実を探しているのではなく、統計的に「もっともらしいトークンの並び」を作っている。



## エンジニアの役割

「確率の波」を、  
プロンプトという  
「防波堤」で制御せよ。

## 入力長と設定が出力に与える影響 なぜ精度が落ちるのか? なぜ答えが変わるのか?

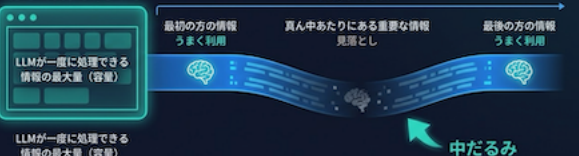
## 出力の「揺れ」と「劣化」

同じプロンプトを投げているのに、  
なぜ、毎回少しずつ答えが変わるのか?  
なぜ、急に精度が落ちることがあるのか?

その原因は「入力の長さ」と「設定」にあります。

## 入力の長さ「コンテキストウィンドウ」の限界

### コンテキストウィンドウ



このコンテキスト内での情報見落としのことを「中だるみ」と呼びます。

## パラメータの設定①: 温度

### 温度 (Temperature) - AIの「慎重さ」を調整するダイヤル

#### 温度を「低く」

(Low Temperature):

AIは常に最も確率が高い

「王道」の言葉を選ぶ。

→予バグや事実確認など、

正確性と再現性が

必要な場合は、温度を低く

設定する

#### 温度を「高く」

(High Temperature):

→キャッチコピーや物語

など、「意外性」が必要

な時は、温度を「高く」

設定する

慎重さ (Caution)

意外性 (Unpredictability)

## パラメータの設定②：Top-P

**Top-P (核サンプリング)**  
候補となる言葉の「足切りライン」を決める設定  
確率の高い順に並べて、合計が上位P%になるまでの単語に候補を絞り込む。

例：Top-Pを0.9 (90%) に設定すると、確率の高い順に単語を並べて、合計が90%に達するまでに含まれる「上位の候補」だけに選択肢を絞り込み、残りのわずかな可能性は完全に切り捨てる。

## 温度とTop-Pの違い

温度とTop-Pは似ていますが、次のような違いがあります。

**温度**  
各単語の「当選確率」  
そのものを変える。

**Top-P**  
選択肢の「範囲」を決める。  
(足切りラインの設定)

## エンジニアの「2つのモード」

実際では、これらを組み合わせて使います。

**正確性重視の場合**  
(デバッグ・コード生成・契約)

温度：0.0-0.3 / Top-P：低め 毎回 (ほぼ) 同じ結果を出す。

**創造性重視の場合**  
(アイデア出し・物語)

温度：1.0-1.5 / Top-P：高め 表現を豊かにする。

## ブラウザ版LLMでの対処法

ブラウザ版では温度やTop-Pを直接数値で指定することはできません。数値の代わりに、「プロンプトによる指示」で制御します。

**正確性重視の指示例**

「事実のみを箇条書きに述べてください。」  
「推測や創造的な表現は避けてください。」

**創造性重視の指示例**

「自由な発想で、独創的かつユーモアを交えて多様なアイデアを出してください。」

## 再現性の確保

「同じ入力には、同じ結果を」  
AIの気まぐれを制御し、信頼されるツールへ。

## 基本プロンプトの黄金律

目的・条件・出力形式

## プロンプトの本質

AIへの指示は、単なる「お願い」ではありません。  
ITエンジニアにとってのプロンプトは、システム開発における「仕様書」や「命令文」であるべきです。  
(曖昧な指示は、仕様未定の開発と同じ)

## プロンプトの3大要素

**1. 目的**  
誰が (役割・ペルソナ)  
何を (最終的なゴール)

**2. 条件**  
「やってはいけないこと」  
「必ず守るべき仕様」

**3. 出力形式**  
「回答の見た目の指定」

## ① 目的

役割を与える  
単に「教えて」ではなく、「役割 (ペルソナ)」と「最終ゴール」を設定します。

**悪い例**  
SQLのクエリを書いて。

**プロの例**  
あなたはDB管理のスペシャリストです。  
パフォーマンス改善のために、以下の処理を行う効率的なSQLクエリを生成してください。

## ② 条件

出力の精度に最も直結する部分です。  
AIが「やってはいけないこと」や「必ず守るべき仕様」を明確にします。

**技術スタック**  
言語はTypeScript、  
WEBフレームワークはNext.jsを使うこと

**対象レベル**  
プログラミング初学者にもわかるように、1行ごとにコメントを付けること

**禁止事項**  
外部ライブラリは使用せず、標準機能だけで実装すること

### ③ 出力形式

後の処理がしやすい形を指定します。  
エンジニアには必須の視点です。

**Markdown**

Markdownで、見出しと箇条書きを使い、コードはコードブロックで囲むこと。

**表形式**

表形式で、メリットとデメリットを比較表にして提示すること。

**JSON形式**

プログラムで解析できるよう、必ず有効なJSON形式で出力すること。

### 【実践比較】素人とITエンジニアのプロンプト

同じ「エラー解決」を依頼する場合でも、指示の出し方によって出力結果は大きく変わります。AIに丸投げせず、プロンプトの構造化を意識してください。

**素人のプロンプト**

指示内容  
Pythonで次のエラーが出たので直して、  
[エラー文]

制約  
なし

出力形式  
AI任せの文章

**ITエンジニアの構造化プロンプト**

指示内容  
あなたはPythonのデバッグ担当です。  
以下のエラーを解析し、原因と修正案を提示してください。  
[エラー文]

制約  
1. エラーの根本原因を特定する  
2. 修正後のコード全体を示す  
3. 不明な点を問うためのアドバイスも1つ添える

出力形式  
以下の構成で出力してください:  
[原因]  
[修正コード]  
[ポイント]

AIに丸投げせず、プロンプトの構造化を意識してください。

### なぜ構造化が必要なのか？

LLMは「確率」で動いています。

条件を指定すること=正解へと続く道に案内板を設置してあげる

**選択枝 (トークンの候補)**  
を絞り込んであげることで、

**ハルシネーション (嘘)**  
が入り込む余地を減らし、

**安定した高品質な回答を**  
引き出すことができる。

### エンジニアのアドバイス

**テンプレート化**

毎回手入力するのは非効率です。  
よく使うパターンはテンプレートとして保存し、『プロンプトの再利用性』を高めましょう。  
開発効率を上げる重要な技術です。

## プロンプトの黄金律

指示が具体的であればあるほど、  
**「確率のブレ」は少なくなる。**

## 入力制約と記録 (入力・設定・出力)

実務で最も差がつくITエンジニアの資産

### 実験と検証の繰り返しをプロンプトにも

- 1回でプロンプトが成功することは稀です。
- 科学者が実験データを蓄積するように、エンジニアも記録を残します。
- たまたまうまくいってOKとするのではなく、
- 何度でも、誰でも再現できる状態を目指しましょう。

### ① 入力制約の徹底 (情報の加工)

**エンジニアの防衛術**

実務データはプロンプトに入力してはいけません。  
必ず「匿名化されたデータ」を使いましょう。

- ダミーデータ**  
構造だけ維持した架空データを使う。
- コードの匿名化**  
社内ライブラリ名を一般名称に置換する。

### ② 記録すべき4つの項目 (高品質プロンプトの特定)

**入力 (プロンプト)**

どんなプロンプトが効いたか

**設定 (パラメータ)**

使用モデル  
温度設定

**出力 (結果)**

結果をそのまま残す

**評価と改善案**

満足度・エラー  
や改善案

### ③ 再現性を確保する (デバッグの考え方)

プログラムのバグ修正と同じく、出力を安定化させましょう。

**3回テスト**

LLMの出力結果の「揺れ」を前提とし、同じプロンプトを3回試行し、毎回合格点が出るかを確認しましょう。

### 検証中は温度設定を固定 Temperature = 0

検証中は温度を「0」に固定します。  
これにより、AIの「気まぐれ」を排除し、プロンプトの純粋な性能のみをテストします。

- 検証中は温度を「0」に固定します。
- これにより、AIの「気まぐれ」を排除し、プロンプトの純粋な性能のみをテストします。

### 記録を「論理的な根拠」にする

「なぜこのAI回答を採用したのか？」と問われた場合に

× 感覚で選んだと答えるのはNG。

「入力・設定・出力」の記録  
↓  
を根拠として提示せよ！

プロのエンジニアとしての説明責任

### 第1回講義まとめ

- 仕組み**  
トークンと確率予測による「自己回帰」
- 制御**  
・ 入力の長さ  
・ 温度設定
- 設計**  
目的・条件・形式 (構造化された指示)
- 記録**  
入力制約の厳守  
ログによる再現性確保

### Next: 基本プロンプトでの結果の揃え方

プロンプトの基本練習と出力改善ループの実践

### 【科目S2】プロンプト設計とLLM活用

## 02 基本プロンプトでの結果の揃え方

再現性のあるコーディング演習

## テンプレートに基づくプロンプト作成

### ITエンジニアのための「攻め」の技術

### 講義の狙い

単に動くコードではなく、「堅牢で、何度実行しても構造が崩壊」コードを出力させる技術を習得します。

AI生成コードの現実 (調査結果) :

- 75%多い**  
ビジネスロジック、買った在庫管理、在庫フローの欠陥 (人間との比較)
- 2倍多い**  
エラーハンドリングや例外処理の欠陥 (人間との比較)

### ケーススタディ：在庫チェックを伴う注文処理

ショッピングサイトのバックエンド機能  
注文処理ロジックを作成します。

TypeScript関数の作成

- ✓ 注文リクエスト受信
- ✓ 在庫チェック
- ✓ 合計金額計算 (税込)
- ✓ 処理結果の成否を返す

### 悪いプロンプト (丸投げ)

「在庫数と注文数、単価を受け取って、在庫があれば合計金額を計算して返す関数をTypeScriptで書いてください。在庫がなければエラーにしてください。」

指示が曖昧で、仕様が定義されていません。

出力結果に「揺れ」が生じる可能性

- × 引数の型が any になる
- × 税計算で誤差が出る
- × 戻り値が揺れる (null? error?)

### 良いプロンプト (制約付き)

目的  
在庫引き当てと金額計算を行う堅牢な関数 `processOrder` の作成

条件 (制約事項)

- ・ 型定義: `OrderRequest` インターフェースを定義し、項目名、単価、注文数、そして在庫数を引数にとるよう指定
- ・ バリデーション: 負の値は具体的なメッセージと共に `Error` を投げる
- ・ 計算精度: `Math.floor`で端数処理
- ・ エッジケース: 在庫0も考慮
- ・ セキュリティ: 標準機能のみ使用

## 出力形式の固定

構造を指定することで、実装のブレを防ぎます。

```
// 以下の構造を守ってコードのみを出力せよ
interface OrderRequest {...}
interface OrderResult {...}
function processOrder( req: OrderRequest, stock:
number ): OrderResult {...}
```

```
TypeScript
// ... (interfaceの定義を省略)
function processOrder(req: OrderRequest, stock: number): OrderResult {
  if (req.quantity <= 0) {
    throw new Error("注文数は1以上を指定してください。");
  }
  if (req.unitPrice <= 0) {
    throw new Error("単価は1以上を指定してください。");
  }
  if (stock < req.quantity) {
    throw new Error("在庫が不足しています。");
  }

  const subtotal = req.unitPrice * req.quantity;
  const tax = Math.floor(subtotal * 0.1);
  const totalAmount = subtotal + tax;
  const remainingStock = stock - req.quantity;
  // ... (return以下を省略)
}
```

## 結果が「揃う」メリット



### レビュー効率化

入力と出力が固定されるため  
「中身のロジック」だけに集中できる。



### 後続工程との連携

形式が固定されていれば  
テストコード生成等へそのまま投入できる。

## 再現性を高めるガードレール



### 否定の制約

「標準以外使わない」  
「多用禁止」等、  
何をしないかを指定。



### 境界条件

「0の時」・「負の時」など  
AIが間違いやしやすい箇所を  
先回りで指示。



### 型の強制

Interfaceを定義し  
AI自身の論理矛盾を  
自己抑制させる。

## 反復実行による揺れの観察 出力の「差分」を観察し、改善せよ

## 揺れの放置はリスクである

LLMの出力には必ず「揺れ」が生じます。  
これを放置することは「品質が運任せになるコード」を  
製品に組み込むことと同じです。



実際にプロンプトを3回実行し、差分（出力の違い）を検証してみましょう。

## 2回目の出力結果

```
TypeScript
(interfaceの定義を省略)
/**
 * 在庫引き当てと金額計算を行う関数
 */
function processOrder(req: OrderRequest, stock: number): OrderResult {
  // バリデーションチェック
  if (req.quantity <= 0) {
    throw new Error("不正な注文数です:$req.quantity, 0より大きい値を入力してください。");
  }
  if (req.unitPrice <= 0) {
    throw new Error("不正な単価です:$req.unitPrice, 0より大きい値を入力してください。");
  }
  if (stock < req.quantity) {
    throw new Error("在庫不足 (現在の在庫: $stock, 注文数: $req.quantity)");
  }

  const totalPriceExcTax = req.unitPrice * req.quantity;
  const taxAmount = Math.floor(totalPriceExcTax * 0.1);
  const totalPriceIncTax = totalPriceExcTax + taxAmount;
  const remainingStock = stock - req.quantity;
  (return以下を省略)
}
```

## 2回目の出力結果の特徴：運用・チーム開発への配慮

1. JSDocによる「コードの仕様」の明文化  
プロンプトの「堅牢な（壊れにくい）」という要求を、単にエラーが出ないことではなく、「後から読む人が使い方を間違えないようにすること」と解釈しています。
2. エラーメッセージへの「具体的コンテキスト」の付与  
エラー通知に留まらず、「なぜそのエラーが起きたのか」を即座に特定できる情報が含まれている。現在の在庫などの動的な数値も埋め込んであり、ログを見ただけで原因が特定できる。
3. 曖昧さを排除した「厳密な命名規則」  
名前から「何が含まれている数値か」を直感的に理解できるよう、あえて詳細な名前を選択している。  
`subtotal` (小計) → 広義的な名前 → `totalPriceExcTax` (税抜合計価格) → 厳密な名前に変更。  
後の工程での計算ミスや、仕様の勘違いによるバグを未然に防ぐ。

## 3回目の出力結果

```
TypeScript
(interfaceの定義を省略)
function processOrder(req: OrderRequest, stock: number): OrderResult {
  // 異常値の除外
  if (req.quantity <= 0 || req.unitPrice <= 0) {
    throw new Error("注文数および単価には0以下の数値を設定できません。");
  }

  // 在庫確認 (在庫がちょうど0になる場合は許容)
  if (req.quantity > stock) {
    throw new Error("注文数が現在の在庫数を超えています。");
  }

  const amountExcludingTax = req.unitPrice * req.quantity;
  const consumptionTax = Math.floor(amountExcludingTax * 0.10); // 10%の税率を適用、端数は切り捨て
  const amountIncludingTax = amountExcludingTax + consumptionTax;
  const remainingStock = stock - req.quantity;
  (return以下を省略)
}
```

## 3回目の出力結果の特徴：正確性とメンテナンス性の両立

1. ドメイン（業務）に基づいた厳密な命名  
単なるプログラミング用語ではなく、会計や事務などの実業務で使われる用語を変数名に採用している。  
`totalPrice` (汎用的な名前) → `amountExcludingTax` (税抜金額) や `consumptionTax` (消費税) (具体的な用語) に変更。
2. 「ガード節」によるロジックの整理  
複雑になりがちなチェック処理（バリデーション）を、効率的な構造に書き換えている。  
複数の `if` 文で隔離化させるのではなく、異常な条件を冒頭でまとめて弾く「ガード節」を採用。
3. 「意図」を伝えるプロセス・コメント  
「何をしているか」だけでなく、「なぜそうしているか」という開発者の意図を言語化している。  
在庫がちょうどゼロになるケースは許容する」といった、仕様上の違いが生じやすい箇所注射液を挿入。  
後任の担当者が「これはバグか、仕様か」と悩まないようにしている。

### AIの思考の「揺れ」

同じ指示でも、優先する価値観が変わります。

同じ指示

- シンプルさ  
最低限の機能実装 (Run 1)
- デバッグ容易性  
詳細なエラーログ (Run 2)
- ビジネス厳密性  
正確な用語と構造 (Run 3)

### プロンプトエンジニアリングの極意

揺れを「不安定」と切り捨てない。  
揺れの観察をヒントに条件を書き足し、  
理想の回答に固定していく。

### 条件追加・形式固定による調整

AIの「自由度」を奪い、再現性を高める

### 「推測」を封じ込める

AI生成コードの問題点、特にコンパイルエラーの94%は「型チェックの失敗」です。  
AIが勝手な想定でコードを書く「空白」を埋めるため、4つのガードレールを設置します。

### 1. 型による「契約」の固定

最も強力なガードレールは  
プロンプト内でインターフェースを厳格に定義することです。

```

// 以下の型を必ず使用すること
interface OrderRequest {
  itemName: string;
  unitPrice: number; // 円
  quantity: number;
}

interface OrderResult {
  itemName: string;
  unitPrice: number;
  quantity: number;
  subtotal: number; // 税抜合計
  tax: number; // 消費税 (10%)
  total: number; // 税込合計
  remainingStock: number;
}

```

### 2. 境界条件の「受入基準」化

AIはエラーハンドリングを省略しがちです。

- 条件を明記
  - 「在庫0」の扱い
  - 「負の値」の挙動を指示
- メッセージ固定
  - 「現在の在庫数を含める」など

### 3. 依存関係の「封鎖」

ハルシネーション対策  
AIは存在しない便利なパッケージを捏造することがあります。  
「悪意あるパッケージ」を掴まされるリスクを排除するため、標準機能のみに限定します。

悪意あるパッケージ / ハルシネーション

標準機能

### 4. 処理プロセスの「段階化」(Chain of Thought)

計算ステップの省略や誤差を防ぐため、手順を指示します。(Chain of Thought)

- バリデーション
- 合計金額算出
- 端数処理

### 調整後のプロンプト (定義)

```

# 目的在庫引き当てと金額計算を行う堅牢な関数 processOrder を作成する。
# システム制約: 言語はTypeScript (strictモード)
# 外部パッケージ使用禁止。
# 標準の Math クラスのみ使用。
# 下記のインターフェースを必ず使用し、変更しないでください。
TypeScript interface OrderRequest { unitPrice: number; quantity: number; }
interface OrderResult { success: boolean; totalAmount: number; message: string; }

```

### 調整後のプロンプト (定義)

```

# アルゴリズム手順
1. バリデーション: qty <= 0 or price <= 0 なら Errorを投げる。
2. 在庫チェック: stock < qty なら success: false と 「在庫不足 (現在: ${stock})」を返す。
3. 計算: subtotal = price * qty tax = Math.floor(subtotal * 0.1) total = subtotal + tax
4. 終了: success: true と結果を返す。

```

調整後のプロンプトを再度3回実行し、以下を確認します。

- 型定義の一致**  
インターフェースが完全一致しているか
- ロジックの統一**  
端数処理やエラー文言が同じか
- 安全性**  
危険な関数や謎のパッケージが含まれていないか

再現性 確認済

## KANBAN BOARD

### 設計とは制約である

「条件」として書き込んだ内容は、そのまま「評価指標」になります

VALIDATE

## 差分記録の作成と提出

### ITエンジニアにとっての「証跡」の確立

### エンジニアリングとしての「証跡」

実務では、「どの制約が、どのリスクを回避するために追加されたのか」を論理的に説明し、記録に残す必要があります。

### 開発プロセスへの統合

ここでは、従来の「評価」や「改善ループ」、そしてチーム間におけるコードレビューの土台となる差分記録の作成と提出方法について学びます。

開発プロセス: 評価 → 差分記録 → コードレビュー → 改善ループ

## 記録すべき必須要素 (前半)

- 1. 入力の変遷**  
初期: 曖昧なプロンプト  
比較記録  
最終版: ガードレール追加  
初期の「曖昧なプロンプト」と、ガードレールを追加した「最終版」を比較記録。どの条件が効いたかを特定する。
- 2. 推論設定**  
使用モデル (GPT-4o等) と Temperature 設定値を記録。再現性の担保に必須。

## 記録すべき必須要素 (後半)

- 3. 揺れと収束**  
調整前の不一致 (型矛盾、命名揺れ) と、調整後の収束状況をコード断片で示す。
- 4. 判断根拠**  
「なぜ直したか」を記述。  
例: 「在庫0判定が漏れていたため、境界値を明記した」

Before/After

Why? Rationale & Example

## Diff分析の「急售」

品質リスクの改善を見ます。

- I/F整合性**  
型チェック失敗が排除されたか?
- 例外網羅**  
負の値やNullがガードされたか?
- 幻覚排除**  
標準ライブラリのみ限定されたか?

## 提出時の遵守事項

- 機密保護**  
実務コードや個人情報をそのまま含めない。
- トレーサビリティ**  
将来の脆弱性対応のための追跡ログとなる。
- エスカレーション**  
未解決の懸念事項は正直に明記する。

## 第2回講義まとめ

- テンプレート作成**  
目的・条件・形式を構造化し、曖昧さを排除した「設計図」をAIに提示する。
- 反復実行**  
同一条件で複数回実行し、AIの「推測」による揺れを可視化する。
- 条件追加・調整**  
型・境界・依存・段階化のガードレールでAIの自由度を奪い、再現性を高める。
- 差分記録**  
修正の判断根拠とBefore/Afterの証跡を残し、品質向上を論理的に説明する。

## Next: 出力評価の基本

今回の記録が「採点基準」になる。

**事実性・一貫性・再現性**  
の3軸による出力評価

## 【科目S2】プロンプト設計とLLM活用 03 出力評価の基本 — よい答えの見分け方

事実性・一貫性・再現性の点検

## 3 観点（事実性・一貫性・再現性）の点検 品質の「急所」を見抜く目を養う

### 評価の3観点



#### 1. 事実性

現実の事実  
信頼できる情報源  
と一致しているか



#### 2. 一貫性

プロンプトで指定した  
「条件」や「型定義」  
を守っているか



#### 3. 再現性

複数回実行しても  
同等の品質が得られるか

### AIコードの品質リスク

AI出力の盲信は危険

ITエンジニアとして、  
厳格な出力品質の評価  
が必要です。



ビジネスロジックの欠陥

人間より

75%

多い

というデータもあります。

### ① 事実性

一見して10%の消費税を端数切り捨てで計算しており、指示通りに見えます。

```
const subtotal = req.unitPrice * req.quantity;  
const tax = Math.floor(subtotal * 0.1);  
const totalAmount = subtotal + tax;  
const remainingStock = stock - req.quantity;
```

「インボイス制度」という実務コンテキストに照らして、確認する必要があります。  
この例は合計額に10%を掛け、1円未満を切り捨てる (Math.floor) 計算なのでOK。

### ② 一貫性

AI生成コードのエラーの94%は「型チェックの失敗」です。

#### ✓評価ポイント

指定した OrderRequest  
等のインターフェースを正確  
に使用しているか？  
ここを死守しているなら高スコアです。



### ③ 再現性

何度実行しても同等の品質が得られるか。  
特定の回だけ「たまたま」良い出力を出していないか？

ガード節による安定  
整理された構造が毎回出力される  
「安定感」こそが、  
実務で採用できるかどうかの境界線。

## 検証優先箇所の特定（数値・固有名詞） AIの弱点を突くチェックポイント

### 2つの検証優先ポイント

#### 数値と境界条件

→境界値分析 (BVA)が有効。  
「値の境界 (最大値・最小値の端)」  
に注目し、その点と、その直上の  
値・直下の値を検証する。

#### 固有名詞と外部依存

パッケージハルシネーション  
→スロップスクワッティング  
などのリスク

これらはAIが特に苦手とする領域です。

### 数値と境界条件のチェック

3回目の出力結果にある条件式に注目してください。

```
// 在庫確認 (在庫がちょうど0になる場合は許容)  
if (req.quantity > stock) {  
  throw new Error("注文数が現在の在庫数を超過しています。");  
}
```

在庫が0になる瞬間であっても正常系の処理に進みます。

## 固有名詞と外部依存のチェック

出力結果にある条件式に注目してください。

```
// 3.金額計算 (浮動小数点誤差と端数処理への配慮)
const subtotal = req.price * req.quantity;
const tax = Math.floor(subtotal * 0.1);
const total = subtotal + tax;
```

今回の出力は「標準の Math クラスのみ使用」という制約も遵守しています。

## 品質ゲートと停止基準の設定 どこまで修正を試みるかの判断基準

## 品質ゲートとは

ソフトウェア開発において、ある工程から次の工程へ進む（あるいはリリースする）ために満たすべき「最低限の品質基準・条件」のことです。

このゲートを通過できない場合、たとえスケジュールが迫っていても作業は次工程へ進めず、不具合の修正や基準達成のための修正が必要となります。

## 必須の最低ライン

最低でも以下の基準を満たす必要があります。

- 即時かつ最優先で対処すべき最も深刻な脆弱性がないこと  
クロスサイトスクリプティングなどの深刻なリスク排除
- 指定したInterfaceとの適合率100%  
指定したInterface (API、関数シグネチャ、データ型、スキーマなど)との適合率が100%でないと、コンパイルエラーやランタイムエラーを起こす可能性があります。

## 「停止基準」を設ける

特定の誤りを3回以上繰り返す場合は、再プロンプトを中止します。

例: 「消費税10%を5%で計算し続ける」

STOP



AIにはこの文脈が理解できないと判断する。

## なぜ中止するのか？



泥沼化

自己修正能力には限界がある



技術的負債

修正ループは開発速度を低下させる



文脈理解

AIのコンテキスト理解の限界点

何度やっても同じ誤りを繰り返す場合、いったん人間が引き取ります。

## 「停止後のアクション」

人間が介入  
直接コードを修正する

分解・再構築  
要件をより小さな単位に分ける

「AIを使いこなす」/「AIに振り回される」の境界線を、停止基準によって明確化する。



## 判断根拠の作業記録 共有前提の証跡を残す

## 説明責任を 果たすために

なぜそのコードを「良し」としたのか。これは個人のメモではなく、チームやクライアントに対して説明責任を果たすための「証跡」です。

「動作確認済み」の一言では不十分です。

## 具体的な根拠の記述例 ①



I/F適合率 100%

指定した TypeScript の Interface との完全一致を確認。



境界値チェック

在庫と注文数の境界値の挙動を目標とテストで点検済み。



ビジネスロジック

金額計算において端数処理が明示されているが、合算ルールについては「上位モジュールで検討必要」と判断。

## 具体的な根拠の記述例②

### ハルシネーションなし

標準ライブラリ以外の依存がないことを確認。

### デバッグ性評価

エラーメッセージに動的な値が含まれており、適用しやすいと判断。

## 記録を残すメリット

### 改善ループ

「どの部分を、なぜ直すべきか」  
AIへの的確なフィードバックが可能になる。

### 追跡可能性

将来障害が起きた際、  
「なぜそう判断したか」  
を追跡できる。

## 第3回講義まとめ

### 1. 3つの観点

事実性・一貫性・再現性で  
厳格に点検する

### 2. 優先箇所

境界値の不等号や  
パッケージ埋道を疑う

## 第3回講義まとめ

### 3. 停止基準

品質ゲートで担保し  
ループ時は人間が介入

### 4. 判断記録

採択理由の根拠を残し  
チーム開発の鍵とする

## 品質の門番へ

AIを盲信する「利用者」から、  
出力を厳格に審査する  
「品質の門番」への進化。

## Next: 再プロンプトによる改善ループ (演習)

評価表で問題点を抽出  
再プロンプトで是正





### 5.3 教育コンテンツ—自習問題

科目	コマ	問題	出題形式	問題文	正答肢	誤答肢	誤答肢	誤答肢
S1	01	01	空欄補充	「生成AIへの指示では、目的と条件に加えて回答の( )を明示すると結果が揃い易い」という文の空欄に入る最も適切なものはどれか。	出力形式	推論設定	入力制約	監査ログ
S1	01	02	空欄補充	「生成AI利用の安全ルールでは、個人情報や機密等、外部送信できない情報を「( )」に区分して入力を避ける」という文の空欄に入る最も適切なものはどれか。	入力禁止	出力形式	作業記録	改善ループ
S1	01	03	事象理解	生成AIにおいて、同一の入力に対して毎回同じ出力が返る性質を何と言うか。	決定論的	確率的	反復的	適応的
S1	01	04	事象理解	生成AIに対して目的と条件と出力形式を含む指示文を設計する行為を何と言うか。	プロンプト設計	監査ログ	データ分類	品質ゲート
S1	01	05	正誤判定	「生成AIの出力は確率的であり、同一の入力でも条件や設定によって結果が揺れることがある」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S1	02	01	空欄補充	「生成AIの出力を確かめる場合、複数の資料を( )して一致点と相違点を確認する」という文の空欄に入る最も適切なものはどれか。	照合	要約	暗号化	量子化
S1	02	02	空欄補充	「生成AIの出力を共有する際、根拠が不明な箇所は言い切りを避けて( )表現に修正する」という文の空欄に入る最も適切なものはどれか。	断定回避	推論設定	出典表示	監査ログ
S1	02	03	事象理解	生成AIが事実でない内容を事実のように尤もらしく出力する現象を何と言うか。	幻覚	過学習	正規化	蒸留
S1	02	04	正誤判定	「生成AIの出力を検証する際、複数の資料を照合して共通点と矛盾点を確認することは有効である」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S1	02	05	正誤判定	「生成AIの出力は内部で自動的に出典確認が行われているため、利用者による出典確認は不要である」という文の内容は正しいか、誤っているか。	誤っている	正しい		
S1	03	01	空欄補充	「生成AIに入力する文章に氏名・連絡先等の( )が含まれると、プライバシー侵害のリスクが高まる」という文の空欄に入る最も適切なものはどれか。	個人情報	機密情報	公開情報	メタデータ
S1	03	02	事象理解	他者の著作物を利用する際、作者名や出典を明示する表示を何と言うか。	出典表示	出典確認	断定回避	データ分類
S1	03	03	事象理解	生成AIへの入力可否や共有範囲に迷った場合、責任者に判断を仰ぐ行為を何と言うか。	エスカレーション	スロットリング	量子化	モニタリング
S1	03	04	正誤判定	「生成AIが生成した文章であっても、第三者の著作物の表現が含まれる可能性があるため、著作権やライセンスの確認は必要である」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S1	03	05	正誤判定	「社外秘情報であっても、利用目的が社内業務であれば、生成AIに入力してよい」という文の内容は正しいか、誤っているか。	誤っている	正しい		
S1	04	01	空欄補充	「組織内AIポリシーでは、生成AIの利用を後から監査できるようにシステム側で自動的に残す( )を定める必要がある」という文の空欄に入る最も適切なものはどれか。	監査ログ	作業記録	出力形式	推論設定
S1	04	02	空欄補充	「生成AIの出力が同じ入力と条件で複数回実行した場合に同程度に再現されるかを点検する観点( )である」という文の空欄に入る最も適切なものはどれか。	再現性	事実性	一貫性	透明性
S1	04	03	事象理解	生成AIの出力評価において、内容が現実の事実と合致しているかを点検する観点を何と言うか。	事実性	一貫性	再現性	安全性
S1	04	04	事象理解	生成AIの長文出力で、同一文書内の説明や数値が矛盾しないかを点検する観点を何と言うか。	一貫性	事実性	再現性	網羅性
S1	04	05	正誤判定	「組織内AIポリシーは、一度作れば改訂不要であり、教育や周知の仕組みも不要である」という文の内容は正しいか、誤っているか。	誤っている	正しい		
S1	05	01	空欄補充	「生成AIが要約した会議メモの文書で根拠を確認できない箇所は、誤情報の断定を避けるため、「( )」と明示して残すことがある」という文の空欄に入る最も適切なものはどれか。	要確認	要公開	要承認	要自動化
S1	05	02	事象理解	生成AIの出力を点検し、プロンプトを修正して再実行する反復手順を何と言うか。	修正ループ	承認フロー	データ分類	品質ゲート

S1	05	03	事象理解	生成AIの出力改善をいつ打ち切って共有するかを決める基準を何と言うか。	停止基準	チェックリスト	推論設定	監査ログ
S1	05	04	正誤判定	「生成AIが作成した社内共有文を点検する際、数値や固有名詞は検証優先箇所として根拠資料に照合することが望ましい」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S1	05	05	正誤判定	「生成AIの出力に誤りが見付かった場合も、差分や判断根拠を記録せずに修正後の文章だけ残せば、運用上充分である」という文の内容は正しいか、誤っているか。	誤っている	正しい		
S1	06	01	空欄補充	「生成AIの利用ルールを作る際、情報を「公開」「社内」「機密」「個人情報」等に（ ）して入力可否と条件を定める」という文の空欄に入る最も適切なものはどれか。	区分	翻訳	圧縮	正規化
S1	06	02	空欄補充	「氏名や社員番号等、個人を識別できる情報は（ ）に該当し、生成AIへの入力条件や保存条件を慎重に定める必要がある」という文の空欄に入る最も適切なものはどれか。	PII	API	KPI	SLO
S1	06	03	事象理解	生成AIに入力した内容や判断根拠を後から追跡できるように残す記録を何と言うか。	作業記録	推論設定	出力形式	コンテキスト長
S1	06	04	正誤判定	「生成AI利用のルールでは、入力可否だけ決めればよく、生成物の保存や共有の条件を定める必要はない」という文の内容は正しいか、誤っているか。	誤っている	正しい		
S1	06	05	正誤判定	「機密や個人情報を生成AIに入力する必要がある場合、疑似化等の安全措置を講じた上で条件を定めることが望ましい」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S2	01	01	空欄補充	「大規模言語モデルは、入力文を（ ）と呼ばれる単位に分割し、次の語を予測しながら文章を生成する」という文の空欄に入る最も適切なものはどれか。	トークン	テンプレート	パラメータ	ベクトル
S2	01	02	空欄補充	「生成AIの出力の揺れや長さは、温度やTop P等の（ ）を変えることで影響を受け得る」という文の空欄に入る最も適切なものはどれか。	推論設定	学習設定	監査ログ	データ分類
S2	01	03	事象理解	大規模言語モデルに与える入力文の長さを何と言うか。	入力長	出力長	学習率	乱数種
S2	01	04	事象理解	生成AIに提示する基本プロンプトの要素として最も適切なものはどれか。	目的条件 出力形式	目的装飾語 語尾	条件温度乱数	形式比喩 絵文字
S2	01	05	正誤判定	「推論設定の変更は、大規模言語モデルの出力の揺れや長さに影響し得る」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S2	02	01	空欄補充	「生成AIの出力の揺れを把握するためには、同じ入力と設定で（ ）して結果を比較する方法が有効である」という文の空欄に入る最も適切なものはどれか。	反復実行	単回実行	手動採点	暗号化
S2	02	02	空欄補充	「生成AIの回答を毎回同じ形で得るためには、回答の箇条書きや表等の（ ）を明示して固定することが重要である」という文の空欄に入る最も適切なものはどれか。	出力形式	入力禁止	監査ログ	学習率
S2	02	03	事象理解	生成AIの出力を複数回比較した場合に内容や形式の違いとして現れる部分を何と言うか。	差分	余白	復号	係数
S2	02	04	正誤判定	「生成AIの出力を安定化するためには、推論設定を毎回変更することが有効である」という文の内容は正しいか、誤っているか。	誤っている	正しい		
S2	02	05	正誤判定	「出力を安定化するために条件追加と形式固定を行った場合、その理由と結果の変化を作業記録に残すことは再現性確認に役立つ」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S2	03	01	空欄補充	「生成AIの回答を点検する際、数値や固有名詞等の誤りが影響しやすい箇所を「（ ）」として先に確認する」という文の空欄に入る最も適切なものはどれか。	検証優先	速度優先	装飾優先	感覚優先
S2	03	02	事象理解	生成AIの出力を共有可能と判断するために設定する最低合格水準を何と言うか。	品質ゲート	停止基準	承認フロー	入力制約
S2	03	03	事象理解	生成AIの出力に根拠が乏しい内容が含まれる場合、言い切りを避けて表現を弱める対応を何と言うか。	断定回避	出典照合	形式固定	自動化
S2	03	04	正誤判定	「生成AIの出力評価で再現性を点検するためには、同一入力と設定で複数回実行して結果が大きく揺れないことを確認する」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S2	03	05	正誤判定	「生成AIの出力評価では、一貫性だけ確認すれば充分であり、事実性の確認は不要である」という文の内容は正しいか、誤っているか。	誤っている	正しい		
S2	04	01	空欄補充	「生成AIの出力を点検する場合、問題点を「誤り」「不足」「冗長」等に整理する（ ）を用いると、作業が体系化される」という文の空欄に入る最も適切なものはどれか。	評価表	設計書	暗号鍵	用語集

S2	04	02	空欄補充	「生成AIの出力を是正するために、指示文を修正して同じ課題を再生成させる操作を（ ）として実施する」という文の空欄に入る最も適切なものはどれか。	再プロンプト	再学習	再配布	再起動
S2	04	03	事象理解	生成AIの出力を検証する際、原契約書や原仕様書等、一番に根拠になる資料を何と言うか。	一次情報	二次資料	参考意見	推論設定
S2	04	04	事象理解	生成AIの出力を評価して再プロンプトでは正し、品質ゲートを充すまで繰り返す手順を何と言うか。	改善ループ	承認フロー	データ分類	スロットリング
S2	04	05	正誤判定	「再プロンプトでの是正では、誤りの根拠を一次情報に照合した上で断定回避を含む修正を行うことが望ましい」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S2	05	01	空欄補充	「生成AIに特定の立場や役割を与えて回答方針を揃えるパターンが（ ）である」という文の空欄に入る最も適切なものはどれか。	役割付与	量子化	暗号化	蒸留
S2	05	02	事象理解	複雑な要求を論点毎に分け、各論点の回答を統合するように生成AIに指示する設計を何と言うか。	分割統治	役割付与	出力例提示	スロットリング
S2	05	03	事象理解	生成AIが誤誘導されないよう、してはいけないことや参照範囲を明示する記述を何と言うか。	制約条件	推論設定	監査ログ	疑似化
S2	05	04	正誤判定	「生成AIへの要求が曖昧な場合に追加質問を先に行わせる設計は、誤解のまま回答を生成するリスクを下げ得る」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S2	05	05	正誤判定	「分割統治を用いる場合、論点を分けずに一括で回答させる方が再現性が高くなる」という文の内容は正しいか、誤っているか。	誤っている	正しい		
S2	06	01	空欄補充	「同じ業務課題に対して二つのプロンプト案を作り、品質と再現性を比べて採択する行為は（ ）に当る」という文の空欄に入る最も適切なものはどれか。	比較評価	暗号化	発散学習	量子化
S2	06	02	空欄補充	「複数案から最終案を選んだ理由や注意点を短文で整理して（ ）として残す」という文の空欄に入る最も適切なものはどれか。	設計意図	学習率	監査ログ	エンコード
S2	06	03	事象理解	業務で利用する回答の様式を揃えるために、生成AIへの指示で項目名や構造を固定することを何と言うか。	出力形式固定	推論設定変更	データ分類	例外承認
S2	06	04	正誤判定	「複数のプロンプト案を比較する際、同一入力と同一推論設定で反復実行し、出力の揺れも含めて評価することが有効である」という文の内容は正しいか、誤っているか。	正しい	誤っている		
S2	06	05	正誤判定	「業務プロンプトを作成する際、役割付与や制約条件を入れずに自由に回答させた方が誤誘導を避け易い」という文の内容は正しいか、誤っているか。	誤っている	正しい		

AI 補完型 IT エンジニア養成プログラム  
プラットフォーム設計書

文書分類	設計書
システム名称	AI 補完型 IT エンジニア養成プログラム プラットフォーム
バージョン	4.1.0
最終更新日	2025/12/20

## 目次

1. システム概要	P.1
2. システム構成	P.2
3. DB テーブル一覧	P.3
4. DB テーブル定義詳細	P.3
5. セキュリティ仕様	P.6
6. インフラ構成仕様	P.7
7. 外部インターフェース一覧（インフラ層）	P.8

---

## 1. システム概要

### 1.1. 背景

従来のオンプレミス型やVM（仮想マシン）ベースのインフラストラクチャでは、スケーラビリティの確保やサーバーメンテナンスに多大なコストと人的リソースが必要とされていました。特に、教育機関向けシステムでは、テスト実施時など特定の時間帯にアクセスが集中する傾向があり、ピーク時に合わせた過剰なリソース確保や、急激な負荷変動への対応が課題となっていました。また、AI技術の進展に伴い、セキュリティを担保しつつ、外部APIとの安全な連携や高負荷な計算処理を実行できる柔軟な基盤が求められています。

### 1.2. 目的

本プラットフォームは、Google Cloud Platform (GCP) および Firebase を全面的に採用した、フルマネージドかつサーバーレスなアーキテクチャの構築を目的としています。インフラ管理の負担を最小限に抑えつつ、高い可用性と柔軟なスケーラビリティを実現します。また、機密性の高い教育データや個人情報を保護するため、認証・認可基盤として Firebase Authentication と Security Rules を活用し、多層的なセキュリティ対策を実装します。さらに、Python コードの実行環境として隔離されたサンドボックス (Cloud Run) を用意し、安全なプログラミング学習環境を提供します。

### 1.3. 導入効果

#### 1.3.1. 運用コストの最適化と負荷耐性

- **完全なサーバーレス構成:** サーバーのプロビジョニングやパッチ適用が不要となり、運用工数を大幅に削減します。
- **オートスケーリング:** Cloud Functions と Cloud Run により、トラフィックの増減に応じてリソースが自動的にスケールアウト/インするため、アクセス集中時でも安定したパフォーマンスを維持しつつ、アイドル時のコストを最小化します。

#### 1.3.2. 高度なセキュリティと信頼性

- **堅牢な認証・認可基盤:** Firebase Authentication による安全なID管理と、Firestore Security Rules による詳細なアクセス制御により、不正アクセスやデータ漏洩のリスクを低減します。

- **隔離された実行環境:** ユーザーが記述した任意のコードは、ネットワークから隔離された Cloud Run コンテナ上で実行されるため、ホストシステムへの影響を防ぎ、安全性を担保します。

### 1.3.3. 開発・デプロイの俊敏性

- **高速なコンテンツ配信:** Firebase Hosting (CDN) により、低レイテンシでアプリケーションにアクセス可能です。
- **統一された開発体験:** フロントエンドからバックエンドまで一貫した Google Cloud エコシステムを利用することで、開発サイクルの短縮と品質向上を実現します。

## 2. システム構成

### 2.1. 全体アーキテクチャ

本システムは **Google Cloud Platform (GCP)** および **Firebase** を基盤とした、フルマネージドでスケラブルなサーバーレスアーキテクチャを採用している。

- **Frontend Hosting: Firebase Hosting**
  - 静的アセット (React/Vite ビルド生成物) の高速配信 (CDN)。
- **Backend Logic: Cloud Functions for Firebase (2nd Gen)**
  - Node.js 20 ランタイム。
  - API エンドポイントおよびバッチ処理、Firestore トリガーの実行。
- **Database: Cloud Firestore**
  - NoSQL ドキュメントデータベース。リアルタイム同期、オフラインサポート。
- **Storage: Cloud Storage for Firebase**
  - 講義資料 (PDF/動画)、画像アセットの格納。
- **Authentication: Firebase Authentication**
  - ID/Password による認証基盤。
- **External Compute: Cloud Run (Executor Service)**
  - Python コード実行用の隔離されたサンドボックス環境 (コンテナ)。

### 2.2. ネットワーク構成

- **リージョン:**
  - **Firestore / Cloud Functions:** asia-northeast1 (東京) 推奨。  
レイテンシ (通信遅延) 低減のため、ユーザー地理的所在地に合わせる。
  - **Cloud Run:** 同一リージョン (asia-northeast1) に展開し、通信遅延を最小化。
- **接続制御:**
  - **Cloud Functions -> Cloud Run:**
    - インターネット経由の HTTPS 通信を使用 (将来的な VPC Connector 利用を想定)。
    - 認証: IAM 認証 (roles/run.invoker) を導入予定。

- Cloud Functions -> Gemini API:
  - Google の公開 API エンドポイントへアクセス (API Key 認証)。

### 3. DB テーブル一覧 (Firestore Collections)

コレクション名	ドキュメント ID	用途・格納データ概要
users	uid (Auth ID)	ユーザーの基本情報 (表示名, メール, role, 作成日時)
courses	uuid (自動生成)	コースメタデータ、サムネイル URL、公開ステータス
enrollments	courseId_userId	受講登録情報。ユーザーとコースの多対多リレーション、進捗率
questionBank	uuid (自動生成)	問題データ (本文, タイプ, 選択肢, 正解/コード, スキルタグ)
testSets	uuid (自動生成)	テストセット情報 (タイトル, 問題 ID 配列, 合格ライン, 制限時間)
testSubmissions	uuid (自動生成)	テスト回答結果 (ユーザーID, テスト ID, 各問の正誤・スコア, 可否)
resources	uuid (自動生成)	講義資料メタデータ (タイトル, ファイル種別, Storage パス)
skillCriteria	uuid (自動生成)	スキル評価基準 (ループリック), 階層化されたスキル定義
dashboard_metrics	latest (固定)	ダッシュボード表示用の集計済みデータ (ランキング, 不正解率等)
lectureMaterials	-	(Legacy: 実装上は resources に統合対象)

## 4. DB テーブル定義詳細

### 4.1. users (ユーザー)

フィールド名	型	必須	説明
uid (DocID)	String	Yes	Auth ユーザーID
displayName	String	Yes	表示名
email	String	Yes	メールアドレス
role	String	Yes	'teacher' または 'student'
createdAt	Timestamp	Yes	アカウント作成日時

### 4.2. courses (コース)

フィールド名	型	必須	説明
uuid (DocID)	String	Yes	コース ID (自動生成)
title	String	Yes	コースタイトル
description	String	No	コース概要
skillCriteriaId	String	No	紐づくスキル評価基準 ID (skillCriteria への参照)
isPublished	Boolean	Yes	公開状態 (true: 公開, false: 非公開)
authorId	String	Yes	作成者のユーザーID (users への参照)
thumbnailURL	String	No	サムネイル画像のダウンロード URL
thumbnailStoragePath	String	No	サムネイル画像の Storage パス (削除用)
activities	Array	No	学習活動 (教材・テスト) のリスト
↳ id	String	Yes	活動ブロック ID

└ type	String	Yes	'resource' (教材) または 'test' (テスト)
└ title	String	Yes	活動タイトル
└ contentId	String	No	実コンテンツの ID (resources または testSets への参照)
└ keywords	Array	No	検索用キーワード
createdAt	Timestamp	Yes	作成日時
updatedAt	Timestamp	Yes	更新日時

#### 4.3. enrollments (受講登録)

フィールド名	型	必須	説明
courseId_userId (DocID)	String	Yes	コース ID とユーザーID の複合キー
userId	String	Yes	受講者のユーザーID
courseId	String	Yes	受講コース ID
progress	Number	Yes	進捗率 (0-100)
completedBlocks	Array	Yes	完了した活動ブロック ID のリスト
startedAt	Timestamp	Yes	受講開始日時

#### 4.4. questionBank (問題バンク)

フィールド名	型	必須	説明
uuid (DocID)	String	Yes	問題 ID (自動生成)
type	String	Yes	'multiple-choice', 'fill-in-the-blank', 'programming'
text	String	Yes	問題文 (Markdown 対応)
skillId	String	No	スキル ID (評価基準に基づく階層 ID)
skillArea	String	No	スキル領域名
difficulty	String	No	'level_beginner', 'level_intermediate', 'level_advanced'
(多肢選択)			
options	Array	Cond	選択肢のリスト (文字列配列)
correctOptionIndex	Number	Cond	正解の選択肢インデックス (0 開始)
(穴埋め)			
correctAnswer	String	Cond	正解文字列
(プログラミング)			
language	String	Cond	プログラミング言語 (例: 'python')
initialCode	String	Cond	初期コード (テンプレート)
solutionCode	String	Cond	模範解答コード
testCases	Array	Cond	採点用テストケースのリスト
└ callCode	String	Yes	実行コード
└ expectedOutput	String	Yes	期待される出力

#### 4.5. testSets (テストセット)

フィールド名	型	必須	説明
uuid (DocID)	String	Yes	テストセット ID
title	String	Yes	テストタイトル

description	String	No	説明文
skillCriteriaId	String	No	スキル評価基準 ID
questionIds	Array	Yes	含まれる問題 ID のリスト (questionBank への参照)
duration	Number	Yes	制限時間 (分)
passMark	Number	Yes	合格点 (0-100)
isPublished	Boolean	Yes	公開フラグ
lectureMaterialId	String	No	関連する講義資料 ID (resources への参照)

#### 4.6. testSubmissions (テスト提出結果)

フィールド名	型	必須	説明
uuid (DocID)	String	Yes	提出 ID
userId	String	Yes	受講者 ID
testId	String	Yes	テストセット ID
testTitle	String	Yes	受験時のテストタイトル (スナップショット)
score	Number	Yes	総合スコア (0-100)
passed	Boolean	Yes	合否結果
submittedAt	Timestamp	Yes	提出日時
answers	Map	Yes	各問題の回答データ (Key: questionId)
└ answer	Any	Yes	ユーザーの回答
└ isCorrect	Boolean	Yes	正誤
└ score	Number	Yes	問題ごとのスコア

#### 4.7. resources (教材リソース)

フィールド名	型	必須	説明
uuid (DocID)	String	Yes	リソース ID
title	String	Yes	教材タイトル
type	String	Yes	'document', 'video', 'excel' 等
storagePath	String	Yes	Firebase Storage 上のパス
createdAt	Timestamp	Yes	アップロード日時

#### 4.8. skillCriteria (スキル評価基準)

フィールド名	型	必須	説明
uuid (DocID)	String	Yes	評価基準 ID
rubricTitle	String	Yes	評価基準セットのタイトル
evaluationItems	Array	Yes	評価項目リスト
└ skillArea	String	Yes	スキル領域名
└ level_*	Object	Yes	各レベル(beginner, intermediate, advanced) の定義
└ description	String	Yes	レベルの説明
└ tags	Array	Yes	関連スキルタグ

#### 4.9. dashboard\_metrics (集計データ)

フィールド名	型	必須	説明
latest (DocID)	String	Yes	固定 ID
dailySubmissions	Array	Yes	日別提出数推移データ
lowProficiencySkills	Array	Yes	苦手スキルランキングデータ
studentRanks	Map	Yes	テストセット別の受講者ランキング
passFailRate	Map	Yes	テストセット別の合格率データ
lastUpdated	Timestamp	Yes	最終集計日時

## 5. セキュリティ仕様

### 5.1. 認証 (Authentication)

- **プロバイダ:** Firebase Authentication (Identity Platform)
- **認証方式:** Email/Password 認証を採用。
- **セッション:** Firebase SDK による JWT (JSON Web Token) 管理 (ID Token)。クライアント側で自動リフレッシュ。
- **パスワードポリシー:** Firebase Default 設定に準拠 (6 文字以上)。

### 5.2. アクセス制御 (Authorization)

アクセス制御は「データアクセス層 (Firestore)」と「ビジネスロジック層 (Functions)」で多層防御を行う。

#### A. Firestore Security Rules (データアクセス制御)

firestore.rules により、データレベルで厳格な権限管理を行う。

コレクション	読み取り (Read/Get/List)	書き込み (Create/Update/Delete)	備考
users	認証済みユーザー全体	自分のみ (request.auth.uid == uid)	他人のプロフィール改ざんを防止
courses	認証済みユーザー全体	教員のみ (resource.data.role == 'teacher')	受講者は閲覧のみ
questionBank	認証済みユーザー全体	教員のみ	
testSets	認証済みユーザー全体	教員のみ	
testSubmissions	教員 または 本人	本人のみ (作成) / 教員のみ (削除)	他人の成績閲覧を防止
enrollments	教員 または 本人	本人のみ	
resources	認証済みユーザー全体	教員のみ	

#### B. Cloud Functions Security (サーバーサイドロジック制御)

Callable Functions 内でコンテキスト (context.auth) を検証し、不正な実行を阻止する。

- **ロールベース制御 (RBAC):**
  - 教員専用関数 (createUser, updateUser 等) は、実行ユーザーの token.role または DB 上のロールを確認し、teacher 以外ならエラー (PermissionDenied) を返す。

- **認証必須制御:**
  - 一般関数 (submitTest, generateTest) は context.auth が存在することを確認し、未認証ならエラー (Unauthenticated) を返す。

### 5.3. API キー・環境変数管理 (Secrets Management)

- **Gemini API Key:**
  - ソースコードへのハードコードは厳禁。
  - **Firebase Functions Config** (functions.config().gemini.key) または **Google Cloud Secret Manager** で管理。
  - 関数実行時にサーバーサイドでのみ参照する。
- **Client Config:**
  - firebase.js には apiKey, authDomain, projectId 等の公開情報を安全に含める (これらは公開してもリソースへの不正アクセスは Security Rules で防がれるため問題ない)。

## 6. インフラ構成仕様

### 6.1. 実行環境 (Cloud Run Executor)

信頼できないコードを実行するためのサンドボックス環境。

- **Image:** Python 3.9+ ベースの軽量カスタムイメージ (Alpine or Slim)。
- **Dependencies:** データサイエンス・数値計算用の主要ライブラリをプリインストール。
  - numpy, pandas, scikit-learn, matplotlib
- **セキュリティ要件 (Sandbox):**
  - **ネットワーク遮断:** 外部への通信を一切許可しない (不正なデータ送信や踏み台化の防止)。
  - **リソース制限:**
    - CPU: 1 vCPU 以下
    - Memory: 512MB - 1GB
    - Timeout: 10 秒 - 30 秒 (無限ループ防止)
  - **ステートレス:** 実行ごとにコンテナ/環境はリセットされ、ファイルは永続化されない。

### 6.2. ストレージ (Cloud Storage)

- **バケット構成:**
  - course\_thumbnails/{uuid}\_{filename}: コースのサムネイル画像。
  - resources/{uuid}\_{filename}: 講義資料ファイル (PDF, Video 等)。
- **アクセス権:**
  - 読み取り: 認証済みユーザー全体に許可。
  - 書き込み: 教員アカウントのみに許可 (storage.rules)。

### 6.3. スケーラビリティと制限

- **Cloud Functions:**
  - インスタンス数: 自動スケーリング (0 to N)。
  - コールドスタート対策: 必要に応じて `minInstances` を設定 (現状は未設定、コスト優先)。
- **Firestore:**
  - 書き込み制限: 約 1 回/秒 (ドキュメント単位)。バッチ書き込み時は 500 件/リクエスト制限。
  - クエリ制限: IN 句は最大 30 要素まで (アプリ側でチャンク分割処理を実装済み)。

## 7. 外部インターフェース一覧 (インフラ層)

### 7.1. 外部接続詳細

接続先	プロトコル	認証	用途
Google Gemini API	gRPC / REST	API Key	テスト生成、フィードバック生成 AI。 モデル: gemini-3 系を推奨 (2025 年 12 月現在)
Google Cloud Run	HTTPS	IAM (OIDC Token) / Internal	Python コード実行サービスのエンドポイント。
Firebase Auth	HTTPS	API Key	ユーザー認証、トークン発行。
Firebase Storage	HTTPS	Security Rules / Signed URL	ファイルのアップロード・ダウンロード。

### 7.2. 内部環境変数定義

バックエンド (functions/.env または Runtime Config) で管理すべき値。

- `GEMINI_API_KEY`: 生成 AI 利用のための API キー。
- `EXECUTOR_URL`: Cloud Run サービスの URL。
- `TZ`: Asia/Tokyo (ログおよびスケジューラ用)。

以 上

AI 補完型 IT エンジニア養成プログラム  
アプリケーション設計書

文書分類	設計書
システム名称	AI 補完型 IT エンジニア養成プログラム アプリケーション
バージョン	5.1.0
最終更新日	2025/12/20

## 目次

1. システム概要	P.1
2. コーディング規約	P.2
3. 機能一覧	P.3
4. 処理機能記述	P.4
5. 画面一覧	P.7
6. 画面仕様（主要画面詳細）	P.9
7. 項目ラベル名一覧	P.15
8. エラーメッセージ	P.17
9. ファイル一覧	P.18
10. 外部インターフェース仕様（API）	P.19
11. バッチ処理一覧	P.21
12. アプリケーション仕様	P.21

## 1. システム概要

### 1.1. 背景

現代の IT 社会において、高度なスキルを持つ IT エンジニアの不足は深刻な課題となっています。従来の教育手法では、座学中心の知識習得に偏りがちであり、実務で求められる「実践的なコーディング能力」や「問題解決能力」を効率的に育成することが困難でした。また、教育現場においては、学習者一人ひとりのコードを詳細にレビューし、適切なフィードバックを行うために多大な人的リソースが必要とされ、指導者の負担増加やフィードバックの遅れが課題となっています。

### 1.2. 目的

本システム「AI 補完型 IT エンジニア養成プログラム アプリケーション」は、最新の生成 AI (Gemini) とクラウド技術を活用し、これらの課題を解決することを目的としています。教員に代わって AI が問題作成、採点、フィードバックを行うことで、質の高い教育をスケーラブルに提供します。特に、プログラミング学習においては、実際にコードを実行・検証する環境を備え、単なる知識確認にとどまらない実践的なスキルの習得を支援します。

### 1.3. 導入効果

#### 1.3.1. 受講者への効果

- **リアルタイムフィードバックによる学習効率の向上:** 提出したコードや回答に対し、即座に採点結果と AI による詳細な解説が得られるため、躓きをすぐに解消し、学習サイクルを加速させることができます。
- **実践的スキルの習得:** 実際のプログラム実行環境を用いたテストにより、現場で通用するコーディング能力を養うことができます。
- **モチベーションの維持・向上:** ダッシュボードやスキル分析機能により、自身の学習進捗や苦手分野が可視化され、自律的な学習を促進します。

#### 1.3.2. 教育者・管理者への効果

- **業務負荷の劇的な削減:** テストの自動生成・自動採点機能により、準備や評価にかかる時間を大幅に削減し、より本質的な指導やメンタリングに時間を割くことが可能になります。
- **データに基づく指導の最適化:** 受講者全体の進捗や誤答傾向を AI が分析し、講義資料の改善点や重点的に指導すべきポイントを提案するため、教育の質を継続的に向上させることができます。
- **教材作成の効率化:** AI アシスタントにより、多様なレベル・形式の問題や教材を短時間で作成することができます。

## 2. コーディング規約

### 2.1. 基本方針

- **フロントエンド:** React (Vite) + Tailwind CSS
- **バックエンド:** Firebase Cloud Functions (Node.js 20, ES Modules)
- **補助システム:** Google Cloud Run (Python Executor Service)

### 2.2. リンター・フォーマッター設定

#### フロントエンド (frontend/)

- **Lint Config:** eslint.config.js (Flat Config)
- **Extends:** js.configs.recommended, reactHooks.configs.recommended, reactRefresh.configs.vite
- **ルール:**
  - no-unused-vars: エラー ( \_で始まる変数は無視)
  - react-refresh/only-export-components: 警告
- **フォーマット規約 (Observed):**
  - **インデント: 4 スペース**
  - **クォート:** シングルクォート (') 推奨 (JSX 属性はダブルクォート)
  - **セミコロン:** 必須
  - **コンポーネント定義:** アロー関数 (const Comp = () => { ... })

#### バックエンド (functions/)

- **Lint Config:** .eslintrc.cjs
- **Extends:** google (Google JavaScript Style Guide 準拠)
- **ルール特記事項:**
  - quotes: ダブルクォート (") 必須 (テンプレートリテラルは許可)
  - prefer-arrow-callback: エラー
  - no-restricted-globals: name, length の使用禁止
- **フォーマット規約:**
  - **インデント: 2 スペース (Google Style)**
  - **セミコロン:** 必須
  - **Console 出力:** console.log ではなく firebase-functions/logger を使用すること

### 2.3. 命名規則

- **変数名・関数名:** camelCase (例: fetchUserRole, calculateStudentRanks)
- **定数:** UPPER\_SNAKE\_CASE (例: IN\_QUERY\_LIMIT)
- **React コンポーネント:** PascalCase (例: AnalysisPage, StudentDashboard)
- **ファイル名:**
  - コンポーネント: PascalCase.jsx (例: TestPage.jsx)
  - フック: camelCase.js (例: useAuthState.js)
  - 設定・ユーティリティ: camelCase.js または kebab-case.js

### 2.4. コメント規約

- **JSDoc:** 関数やクラスの定義時には、引数・戻り値を説明する JSDoc (/\*\* ... \*/) を記述する。
- **TODO:** 未実装箇所や修正が必要な箇所には // TODO: または // FIXME: を付与する。

### 2.5. エラーハンドリング規約

- **バックエンド (Cloud Functions):**

- クライアントへのエラー返却には必ず `HttpError` を使用する。
- 予期せぬエラーは `internal` コードでラップし、詳細は `logger.error` に記録する。

### 3. 機能一覧

#### 3.1. 教員用機能 (Teacher Role)

大分類	機能 ID	機能名称	詳細機能・備考
ダッシュボード	T-001	ダッシュボード表示	<ul style="list-style-type: none"> <li>・システム全体の統計 (受講者数、テスト実施数)</li> <li>・要注意受講者のアラート表示</li> <li>・システム稼働状況の確認</li> </ul>
コース管理	T-002	コース編集・作成	<ul style="list-style-type: none"> <li>・コース新規作成、基本情報 (タイトル、概要) 編集</li> <li>・サムネイル画像のアップロード</li> <li>・公開/非公開の設定</li> </ul>
	T-003	活動 (Activity) 管理	<ul style="list-style-type: none"> <li>・コースへの教材・テストの追加・削除</li> <li>・ドラッグ&amp;ドロップによる順序変更</li> <li>・活動ごとのキーワード設定</li> </ul>
	T-004	進捗管理	<ul style="list-style-type: none"> <li>・コースごとの受講者の進捗率確認</li> <li>・完了/未完了ステータスの管理</li> </ul>
問題・テスト管理	T-005	問題作成 (AI 生成)	<ul style="list-style-type: none"> <li>・Gemini API を用いた自動生成</li> <li>・対応形式: 選択式、穴埋め、プログラミング</li> <li>・生成パラメータ: トピック、難易度、問題数</li> </ul>
	T-006	問題作成 (手動)	<ul style="list-style-type: none"> <li>・マニュアル入力による問題作成</li> <li>・プログラミング問題のテストケース定義 (入力/出力/セットアップコード)</li> </ul>
	T-007	問題バンク管理	<ul style="list-style-type: none"> <li>・作成済み問題の一覧表示、検索、フィルタリング</li> <li>・既存問題の再編集・削除</li> </ul>
	T-008	テストセット管理	<ul style="list-style-type: none"> <li>・複数の問題を組み合わせた「テストセット」の作成</li> <li>・合格点、制限時間 (将来構想) 等の設定</li> </ul>
分析・評価	T-009	マルチモーダル分析	<ul style="list-style-type: none"> <li>・テスト結果と講義資料 (PDF/テキスト) を連携させた AI 分析</li> <li>・誤答傾向に基づく教材改善提案の生成</li> </ul>
	T-010	スキル/ルーブリック管理	<ul style="list-style-type: none"> <li>・評価基準 (ルーブリック) の定義</li> <li>・スキルセットの編集とコースへの紐付け</li> </ul>
	T-011	結果詳細確認	<ul style="list-style-type: none"> <li>・受講者個人のテスト提出内容の閲覧</li> <li>・プログラミングコードの diff 確認</li> <li>・実行結果ログの確認</li> </ul>
ユーザー管理	T-012	受講者管理	<ul style="list-style-type: none"> <li>・受講者アカウントの一覧表示、登録、削除</li> <li>・(管理者機能としての側面あり)</li> </ul>

### 3.2. 受講者用機能 (Student Role)

大分類	機能 ID	機能名称	詳細機能・備考
学習	S-001	ダッシュボード表示	<ul style="list-style-type: none"> <li>学習進捗の概況表示</li> <li>最新のお知らせ確認</li> </ul>
	S-002	コース受講	<ul style="list-style-type: none"> <li>受講可能なコースの一覧閲覧</li> <li>コース詳細画面へのアクセス</li> </ul>
	S-003	教材閲覧	<ul style="list-style-type: none"> <li>PDF やテキスト形式の講義資料の閲覧</li> </ul>
テスト	S-004	テスト受験	<ul style="list-style-type: none"> <li>Web インターフェース上での解答</li> <li>プログラミングコードの記述とブラウザ上での実行 (バックエンド処理)</li> <li>リアルタイム正誤判定 (選択式など)</li> </ul>
	S-005	フィードバック確認	<ul style="list-style-type: none"> <li>採点結果 (スコア、合否) の即時確認</li> <li>AI によるコードレビュー・改善アドバイスの閲覧</li> </ul>
履歴	S-006	結果履歴	<ul style="list-style-type: none"> <li>過去に受験したテストの結果履歴一覧</li> <li>自身のスキル習得状況の可視化 (レーダーチャート等)</li> </ul>

## 4. 処理機能記述

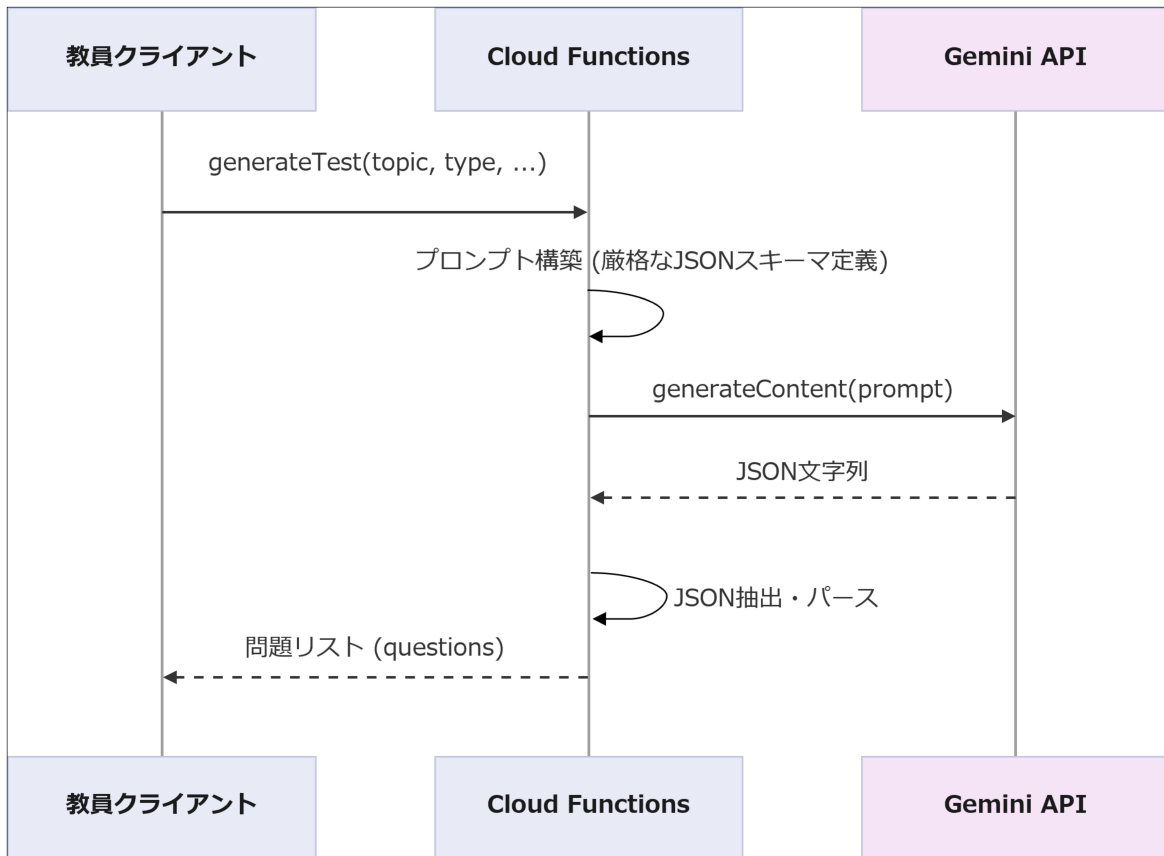
### 4.1. テスト生成 (generateTest)

**概要:** 教員の指定したトピックに基づいて、AI がテスト問題を自動生成します。特にプログラミング問題では、正解コードとテストケースの整合性を担保する仕組みがあります。

**処理フロー:**

- リクエスト:** 教員がトピック、問題数、形式、言語を指定。
- プロンプト構築:**
  - プログラミング問題の場合: 「Python の専門家」としての役割を与え、JSON 形式 (solutionCode, testCases を含む) での出力を強制。
  - 自己検証指示:** AI に対し、「自身の生成した正解コードを実行し、テストケースの期待値と一致すること」を強く指示。
- 生成とパース:** Gemini API からの応答 (テキスト) から JSON 部分を抽出し、オブジェクト化。
- レスポンス:** 生成された問題リストをクライアントに返却 (この時点では DB 保存されない)。

### テスト生成 (generateTest) シーケンス図



### 4.2. テスト提出・採点 (submitTest)

**概要:** 受講者の解答を受け取り、自動採点を行います。プログラミング問題は Google Cloud Run 上で稼働する Executor Service を使用して実際にコードを実行して評価します。

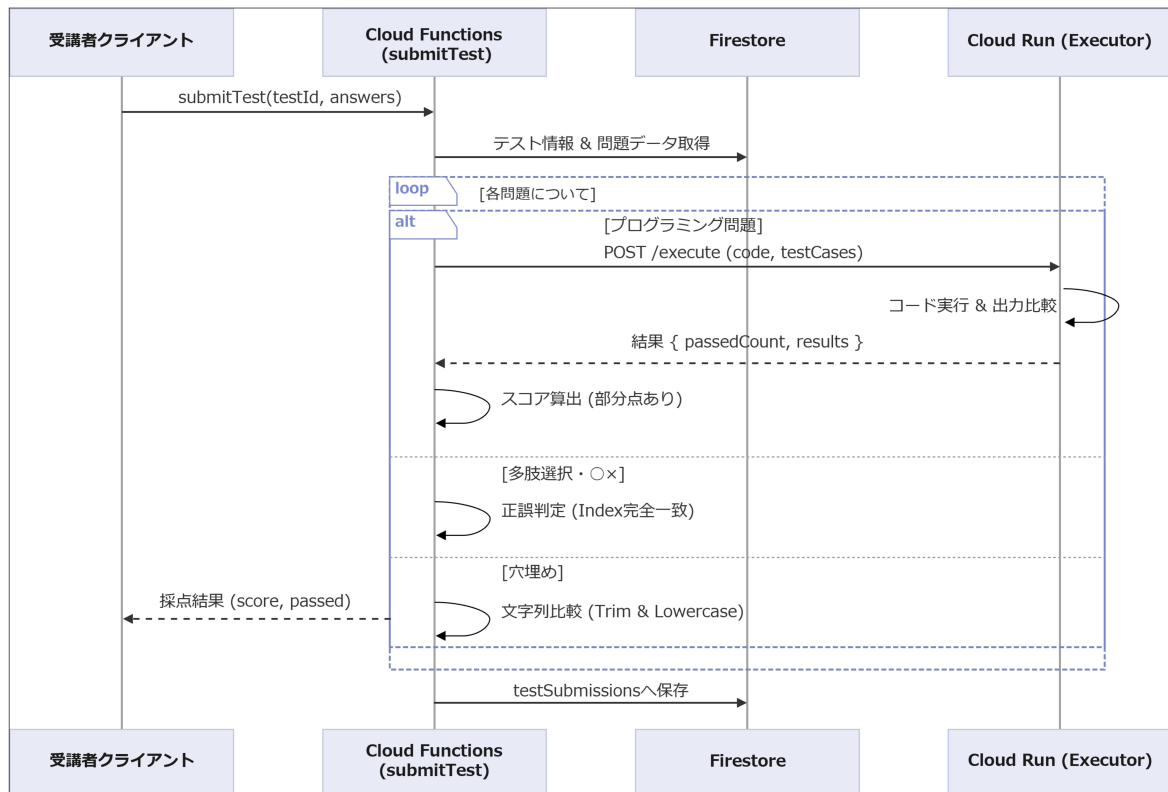
**処理フロー:**

1. **データ取得:** testId を元にテスト情報を取得。問題 ID リストから IN クエリ (30 件分割) を用いて全問題データを取得。
2. **ループ処理:** 各問題について採点を実施。
  - **多肢選択 (Multiple Choice):**
    - ○×問題 (True/False) を含む。
    - ロジック: userAnswer (選択された index) が correctOptionIndex と完全一致する場合に正解 (100 点)。それ以外は 0 点。
  - **穴埋め (Fill-in-the-blank):**
    - ロジック: userAnswer (文字列) と correctAnswer (文字列) を比較。
    - **正規化:** 両方の文字列に対して .trim().toLowerCase() を適用し、大文字・小文字の差異や前後の空白を無視して比較する。完全一致で 100 点。
  - **プログラミング:**
    - Cloud Run 上の Executor Service へ language, studentCode, testCases を送信。
    - Executor はコンテナ環境でコードを実行。
    - テストケースごとの通過数 (passedCount) に基づいて部分点を算出。
3. **集計・保存:** 総合スコアと合否判定を行い、testSubmissions コレクションに保存。

### Executor Service の検証ロジック:

- Python の pandas, numpy, matplotlib 等をサポート。
- **数値判定:** 浮動小数点の誤差 (1e-9) を許容。
- **特殊判定:** K-Means や PCA など、順序が不定な出力のための専用比較ロジックを実装。

### テスト提出・採点 (submitTest) シーケンス図



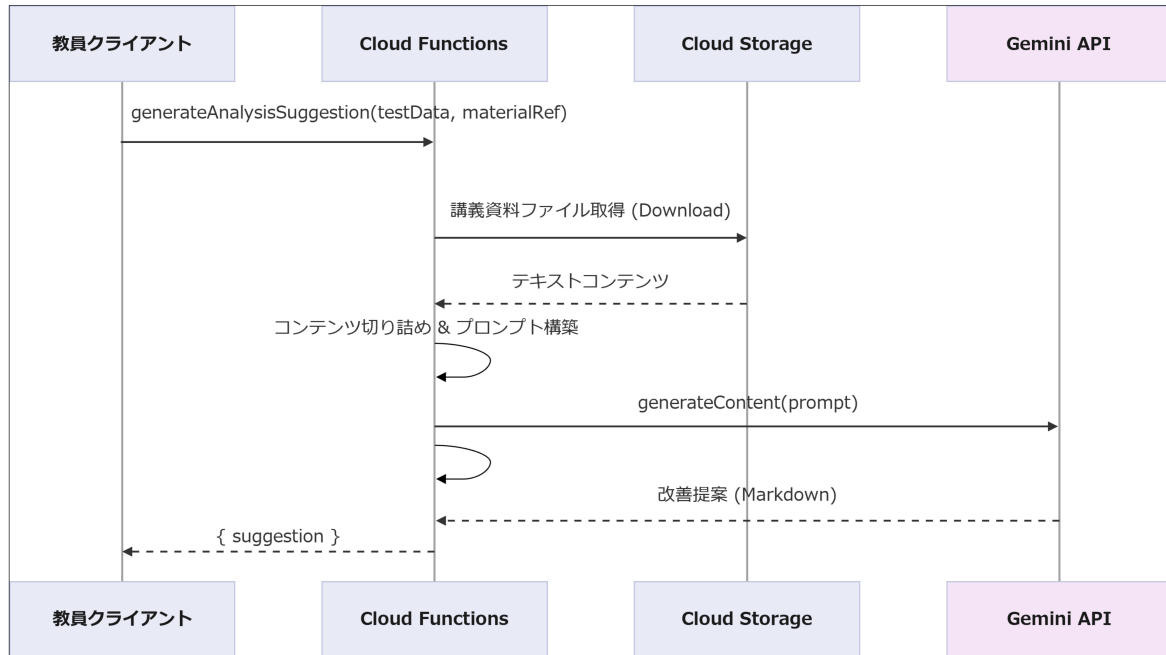
### 4.3. AI 分析提案 (generateAnalysisSuggestion)

**概要:** テスト結果と講義資料を突き合わせ、AI が講義資料の改善案を提示します。

#### 処理フロー:

1. **データ準備:**
  - クライアントから「誤答率の高い問題リスト」を受領。
  - **講義資料取得:** まず Firebase Storage からのダウンロードを試行。失敗または未設定時は Firestore のスニペットを使用。
2. **プロンプト構築:**
  - 「教育工学の専門家」としての役割を設定。
  - 誤答パターンから学生の躓きポイントを推測させ、資料のどの部分をどう修正すべきか（具体的なアクション）をリスト形式で出力するように指示。
3. **生成:** Gemini API により改善提案マークダウンを生成。

## AI 分析提案 (generateAnalysisSuggestion) シーケンス図



## 5. 画面一覧

## 5.1. 共通・認証 (Common / Auth)

画面 ID	画面名称	URL / Route	ページコンポーネント	主な機能・アクション
G-001	ログイン画面	/login	LoginPage	<ul style="list-style-type: none"> <li>メールアドレス/パスワード認証</li> <li>ログイン失敗時のエラー表示</li> </ul>
G-000	リダイレクター	/	RootRedirector	<ul style="list-style-type: none"> <li>ロールに基づくダッシュボードへの自動転送</li> <li>認証状態のチェック</li> </ul>

## 5.2. 教員用画面 (Teacher Role)

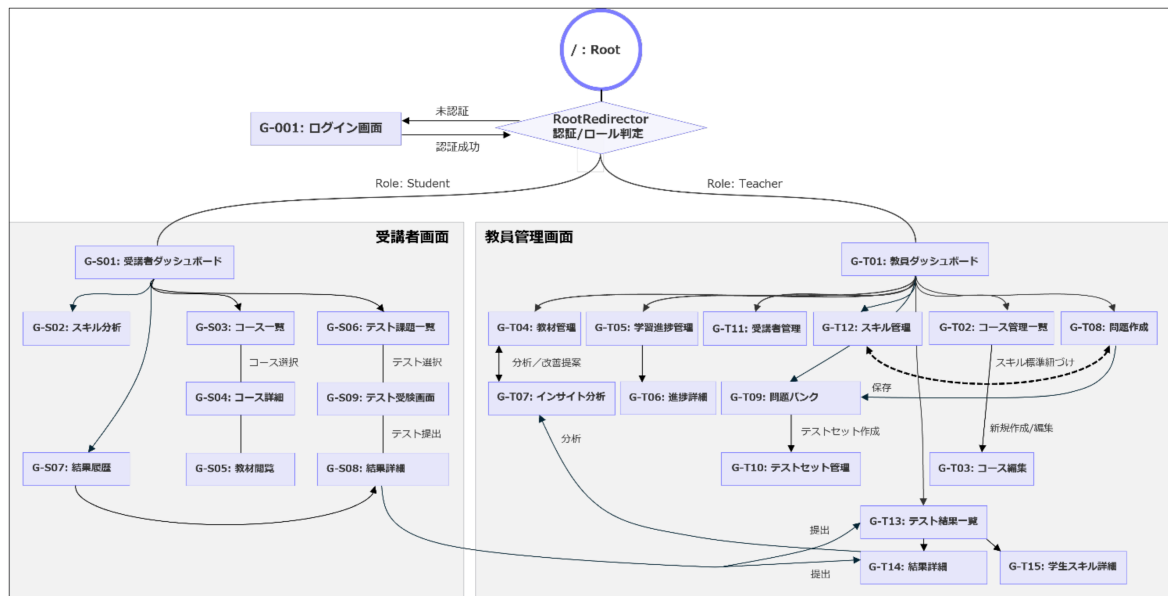
画面 ID	画面名称	URL (/teacher/...)	ページコンポーネント	主な機能・アクション
G-T01	ダッシュボード	/(index)	Dashboard	<ul style="list-style-type: none"> <li>システム全体の統計確認</li> <li>要注意学生のアラート確認</li> </ul>
G-T02	コース管理一覧	/course-management	CourseManagementPage	<ul style="list-style-type: none"> <li>作成済みコースの検索・一覧表示</li> <li>新規作成ボタン、編集ボタン</li> </ul>
G-T03	コース編集	/course-management/edit/:id /course-management/new	CourseEditPage	<ul style="list-style-type: none"> <li>基本情報 (タイトル、サムネイル) 編集</li> <li>活動 (教材・テスト) の追加・並べ替え</li> </ul>
G-T04	教材管理	/resources	ResourceManagementPage	<ul style="list-style-type: none"> <li>講義資料 (PDF 等) のアップロード</li> <li>資料のプレビュー、削除</li> </ul>
G-T05	学習進捗管理	/progress	ProgressManagementPage	<ul style="list-style-type: none"> <li>コースごとの進捗率一覧表示</li> </ul>

				・詳細ページへの遷移
G-T06	進捗詳細	/progress/:courseId	CourseProgressDetailPage	・受講者ごとの進捗状況マトリクス表示
G-T07	インサイト分析	/analysis	AnalysisPage	・テスト結果のAI分析実行 ・改善提案の閲覧とコピー
G-T08	問題作成 (各種)	/create /edit/general /edit/programming	QuestionCreationPage	・AIによる問題生成 (Gemini 使用) ・手動による問題編集・作成
G-T09	問題バンク	/manage/questions	QuestionBank	・ストック問題の一覧、検索、編集、削除
G-T10	テストセット 管理	/manage/test-sets	TestSetManagement	・問題セットの組み立て ・配点、合格点設定
G-T11	受講者管理	/manage/students	StudentManagementPage	・受講者アカウントの作成、編集、削除
G-T12	スキル管理	/manage/skills	SkillManagementPage	・評価基準 (ルーブリック) の定義 ・スキルレベル設定
G-T13	テスト結果一覧	/results	TestResultsPage	・全受講者のテスト提出履歴一覧
G-T14	テスト結果詳細	/results/:submissionId	TeacherResultDetailPage	・個別の提出内容 (コード、正誤) の詳細確認
G-T15	学生スキル詳細	/skills/:userId	StudentSkillsPage	・特定学生のスキル習得状況確認

### 5.3. 受講者用画面 (Student Role)

画面 ID	画面名称	URL (/student/...)	ページコンポーネント	主な機能・アクション
G-S01	ダッシュ ボード	/(index)	StudentDashboardContent	・学習の進捗概要、お知らせ表示
G-S02	スキル分析	/skills	StudentSkillsPage	・自身のスキル習得状況の可視化
G-S03	コース一覧	/courses	StudentCourseList	・受講可能コースの閲覧
G-S04	コース詳細	/courses/:courseId	StudentCourseViewerPage	・コース内アクティビティ (教材・テスト) の選択
G-S05	教材閲覧	/resource/:resourceId	ResourceViewerPage	・PDF/テキスト資料の学習
G-S06	テスト課題 一覧	/tests	AssignmentList	・未受験/受験済みテストの一覧
G-S07	結果履歴	/results	StudentResultsPage	・過去のテスト結果一覧表示
G-S08	結果詳細	/results/:submissionId	StudentResultDetailPage	・採点結果の確認 ・AI フィードバックの確認
G-S09	テスト 受験画面	/test/:testId (Root Path)	TestPage	・問題解答 (選択、記述、コード) ・リアルタイム採点実行 ・(注: Role に関わらずテスト実行時はここを使用)

## 5.4. 画面遷移図



## 6. 画面仕様（主要画面詳細）

### 6.1. 共通・認証 (Common / Auth)

#### [G-001] ログイン画面

- URL: /login
- コンポーネント: LoginPage.jsx
- UI 要素:
  - メールアドレス入力フィールド (type="email", required)
  - パスワード入力フィールド (type="password", required)
  - 「ログイン」ボタン (Submit)
  - エラーメッセージ表示領域 (赤文字)
- 主なアクション:
  - ログイン: Firebase Auth (signInWithEmailAndPassword) を実行。
  - 成功時は / へ遷移。失敗時はエラーを表示。
- 備考: すでにログイン済みの場合は / へ自動リダイレクト。

#### [G-000] リダイレクター (Root)

- URL: /
- コンポーネント: main.jsx (RootRedirector)
- ロジック:
  - 未認証 → /login へリダイレクト
  - 教員 (role='teacher') → /teacher へリダイレクト
  - 受講者 (role='student') → /student へリダイレクト
  - 読み込み中 → "読み込み中..." ローディング表示

## 6.2. 教員用画面 (Teacher Role)

### [G-T01] 教員ダッシュボード

- URL: /teacher
- コンポーネント: Dashboard.jsx
- UI 要素:
  - 統計ウィジェット:
    - テストセットごとの総回答数 (折れ線グラフ + バーチャート)
    - 合格率/不合格率 (円グラフ)
    - コース進捗率 (バーチャート)
  - ランキング:
    - 正解率が高い受講者 Top 5
    - 正解率が低い受講者 Top 5 (要注意学生)
  - スキル分析: 習熟度が低いスキル Top 5 (バーチャート)
- 主なアクション:
  - フィルタリング: テストセット ID、年月、コース ID による表示データの絞り込み。
  - 手動更新: データの再集計トリガー (Cloud Functions callAggregateDashboardMetrics 実行)。
  - 詳細トグル: 各ウィジェットの拡大/縮小表示切り替え。

### [G-T02] コース管理一覧

- URL: /teacher/course-management
- コンポーネント: CourseManagementPage.jsx
- UI 要素:
  - コース一覧テーブル (タイトル、スキル基準、作成日、公開/非公開、進捗)
  - [インポート/エクスポート] ボタン (JSON 形式)
  - [+新規コースを作成] ボタン
- 主なアクション:
  - 一覧表示: Firestore courses コレクションから一覧を取得・表示。
  - 削除: コースの物理削除。
  - 進捗確認: コンテキストメニューから進捗詳細画面へ遷移。

### [G-T03] コース編集・作成

- URL: /teacher/course-management/edit/:id / new
- コンポーネント: CourseEditPage.jsx
- UI 要素:
  - 一般設定フォーム:
    - コース名 (必須)、評価基準選択、概要
    - サムネイル画像アップロード (プレビュー付き)
    - 公開設定 (公開/非公開ラジオボタン)
  - 活動リスト (Drag & Drop):
    - 活動ブロックの追加/削除/並べ替え (react-beautiful-dnd)
    - 各ブロック: タイトル、種別(教材/テスト)、コンテンツ選択、キーワード
- 主なアクション:
  - 保存: Firestore courses コレクションへの保存。サムネイルがある場合は Storage へアップロード。

### [G-T04] 教材管理

- URL: /teacher/resources
- コンポーネント: ResourceManagementPage.jsx
- UI 要素:
  - 教材一覧テーブル (タイトル、種類、作成日時)
  - [+新規教材を追加] ボタン (モーダル起動)
  - 編集モーダル: タイトル入力、ファイルアップロード (Drag & Drop or Select)
  - ファイル種別自動判定 (PDF, Word, Video 等)

- **主なアクション:**
  - **アップロード:** Firebase Storage へファイルをアップロードし、メタデータを Firestore resources に保存。
  - **編集/削除:** タイトル変更、ファイルの差し替え、物理削除。

#### [G-T05] 学習進捗管理 (一覧)

- **URL:** /teacher/progress
- **コンポーネント:** ProgressManagementPage.jsx
- **UI 要素:**
  - コース別進捗サマリーテーブル (受講者数、平均進捗率)
  - [エクスポート (JSON)] ボタン
- **機能:**
  - コースごとの全受講者の進捗平均を計算して表示。

#### [G-T06] 学習進捗詳細 (コース別)

- **URL:** /teacher/progress/:courseId
- **コンポーネント:** CourseProgressDetailPage.jsx
- **UI 要素:**
  - 受講者一覧テーブル
  - 受講者ごとの進捗バー (0-100%)
  - 未完了の活動リスト表示
- **機能:**
  - 特定コース内の各受講者の進捗状況を詳細表示。

#### [G-T07] インサイト分析

- **URL:** /teacher/analysis
- **コンポーネント:** AnalysisPage.jsx
- **機能:**
  - 特定のテストを選択し、誤答率の高い問題を抽出。
  - 関連する講義資料と突き合わせ、AI (Gemini) に改善提案を生成させる。
  - 結果は Markdown 形式で表示され、クリップボードにコピー可能。

#### [G-T08] 問題作成 (各種)

- **URL:** /teacher/create //teacher/edit/...
- **コンポーネント:** QuestionCreationPage.jsx
- **構成:** タブ切り替えによる複数モード対応
  - 通常問題(手動):** 多肢選択/穴埋め。スキル評価基準の選択 UI。
  - プログラミング(手動):** 記述式。テストケース入力、模範解答コード入力。
  - AI 生成:** Gemini API を用いた自動生成。プロンプト入力と生成結果のプレビュー/修正。
- **主なアクション:**
  - **問題バンクへ保存:** 作成した問題を Firestore questionBank に保存。

#### [G-T09] 問題バンク

- **URL:** /teacher/manage/questions
- **コンポーネント:** QuestionBank.jsx
- **UI 要素:**
  - スキル評価基準ごとのアコーディオンリスト
  - 問題検索/フィルタリング
  - 問題一覧テーブル (スキル領域、レベル、タグ、問題タイプ)
  - 一括操作バー (削除、エクスポート)
  - インポート機能 (JSON)
- **機能:**
  - 登録済み全問題の閲覧、編集画面への遷移、一括削除。

#### [G-T10] テストセット管理

- URL: /teacher/manage/test-sets
- コンポーネント: TestSetManagement.jsx
- UI 要素:
  - テストセット一覧テーブル
  - 編集モーダル:
    - 基本設定 (タイトル、説明、制限時間、合格点、公開設定)
    - 紐付け設定: スキル評価基準、講義資料 (分析用)
    - 問題選択: 基準に基づく問題をチェックボックスで選択可能 (グルーピング表示)
- 主なアクション:
  - 作成/更新: 問題 ID のリストを含むテストセット定義を Firestore testSets に保存。

#### [G-T11] 受講者管理

- URL: /teacher/manage/students
- コンポーネント: StudentManagementPage.jsx
- UI 要素:
  - 受講者検索バー (名前/Email)
  - 受講者一覧テーブル
  - [+新規受講者を登録] ボタン (モーダル)
  - CSV エクスポート
- アクション:
  - CRUD: Cloud Functions (createUser, updateUser, deleteUser) を経由して Firebase Auth ユーザーを管理。

#### [G-T12] スキル管理

- URL: /teacher/manage/skills
- コンポーネント: SkillManagementPage.jsx
- UI 要素:
  - ルーブリック(評価基準)一覧テーブル
  - 編集フォーム (ルーブリックタイトル、評価項目: スキル領域/レベル別タグ定義)
  - インポート/エクスポート
- 機能:
  - 問題やコースで使用されるスキル評価基準のマスターデータ管理。

#### [G-T13] テスト結果一覧

- URL: /teacher/results
- コンポーネント: TestResultsPage.jsx (List: TestResults.jsx)
- UI 要素:
  - 受講者ごとのアコーディオンリスト (展開すると受験履歴を表示)
  - 一覧項目: テスト名、スコア、可否、バッジ、スキルマップへのリンク
  - 削除ボタン
- 機能:
  - 全受講者のテスト提出履歴を一覧表示。

#### [G-T14] 結果詳細

- URL: /teacher/results/:submissionId
- コンポーネント: TeacherResultDetailPage.jsx (Content: SubmissionDetail.jsx)
- UI 要素:
  - サマリー: スコア、可否、プログレスバー
  - 詳細リスト: 各問題の正誤、モードに応じた詳細表示
    - プログラミング問題: コード Diff、テストケース実行結果ログ
    - AI フィードバック生成機能: 各問題に対して「AI 解説を作成」ボタン (Teacher 権限で実行可能)

### 6.3. 受講者用画面 (Student Role)

#### [G-S01] 受講者ダッシュボード

- URL: /student
- コンポーネント: StudentDashboardContent.jsx
- UI 要素:
  - 学習状況カード: 最新の受講コース名とプログレスバー、[コースを続ける] リンク。
  - 挑戦できるテスト: 最新の公開テスト 3 件のリスト、[挑戦する] ボタン。
  - 最新のテスト結果: 直近の回答スコアと合否判定、[詳細を見る] リンク。
  - スキルサマリー: スキル分析ページへのショートカット。
- 機能:
  - ログインユーザーの最新データを Firestore からリアルタイム取得し、学習の継続を促す。

#### [G-S02] コース一覧

- URL: /student/courses
- コンポーネント: StudentCourseList.jsx
- UI 要素:
  - 公開済みコースのカードグリッド表示
  - コースサムネイル、タイトル、説明文
  - [学習を始める] バッジ
- 機能:
  - 公開フラグ(isPublished: true)が立っている全コースを表示。クリックでコース詳細へ遷移。

#### [G-S03] コース受講画面

- URL: /student/courses/:courseId
- コンポーネント: StudentCourseViewerPage.jsx
- UI 要素:
  - 活動(Activity) リスト
  - 各活動のステータス表示 (完了済み/未完了)
  - コンテンツ種別アイコン (PDF/動画/テスト)
  - アクションボタン: [閲覧する] / [受験する] / [もう一度見る]
- 処理:
  - 初回アクセス時に enrollments コレクションに受講データを自動作成。
  - 活動をクリックすると進捗 (completedBlocks, progress) を更新し、対象コンテンツへ遷移。

#### [G-S04] 教材閲覧

- URL: /student/resource/:resourceId
- コンポーネント: ResourceViewerPage.jsx
- UI 要素:
  - PDF の場合: iframe による埋め込み表示。
  - 動画の場合: HTML5 Video Player + 再生速度変更プルダウン (0.5x ~ 3.0x)。
- 機能:
  - 指定された教材コンテンツを表示する。

#### [G-S05] テスト一覧 (課題リスト)

- URL: /student/tests
- コンポーネント: AssignmentList.jsx
- UI 要素:
  - 公開テスト一覧テーブル
  - ステータス表示: 未解答 / 合格 / 不合格
  - アクションボタン: [挑戦する] / [再挑戦する] / [結果を見る]
- 機能:
  - 受験可能なテストと、自身の過去の受験状況を一覧表示。

**[G-S06] テスト受験画面**

- URL: /test/:testId
- コンポーネント: TestPage.jsx
- 備考: AuthGuard で保護されているが、Header 等は非表示の独立画面 (FullScreen 推奨)。
- 機能: 問題回答、制限時間カウントダウン、回答送信 (submitTest 呼び出し)。
- ステート:
  - 読み込み中: スピナー表示
  - 回答中:
    - 問題リスト (全問表示)
    - 各問題タイプに応じた入力 UI (多肢選択/穴埋め/コードエディタ)
    - 「テストを提出する」ボタン
  - 提出中: 全画面オーバーレイ。プログレスバーと進捗率(%)を表示。
  - 結果表示: スコア(%), 合否判定 (🟢合格/🔴不合格)、ダッシュボードへ戻るボタン。

**[G-S07] テスト結果一覧**

- URL: /student/results
- コンポーネント: StudentResultsPage.jsx
- UI 要素:
  - 受験履歴テーブル (テスト名、スコア、合否、日時)
  - [結果を見る] リンク
- 機能:
  - 過去の全ての受験結果を表示。

**[G-S08] 結果詳細**

- URL: /student/results/:submissionId
- コンポーネント: StudentResultDetailPage.jsx (Content: SubmissionDetail.jsx)
- UI 要素:
  - スコアサマリー、合否判定
  - 正誤一覧 (合格時または設定により正解/解説を表示)
  - AI フィードバック (各問題ごとの解説)
- 制御: viewerRole="student" としてコンポーネントを利用し、教員用とは異なる表示制御を行う (例: 不合格時は正解を隠すなど)。

**[G-S09] スキル分析**

- URL: /student/skills
- コンポーネント: StudentSkillsPage.jsx (Content: SkillsDashboard.jsx)
- UI 要素:
  - レーダーチャート: スキル領域ごとの習熟度
  - スキル詳細リスト: 各評価基準ごとのスコア
- 機能:
  - 自身の全テスト結果に基づいたスキル習熟度を可視化。

## 7. 項目ラベル名一覧

### 7.1. ログイン画面

項目名	必須	タイプ	初期値/備考
メールアドレス	必須	Email	placeholder: email@example.com
パスワード	必須	Password	placeholder: Password

## 7.2. コース編集画面 (/teacher/course-management)

### 基本設定

項目名	必須	タイプ	備考
コース名	必須	Text	
紐づけるスキル評価基準	任意	Select	登録済みのスキル基準から選択
概要	任意	Textarea	コースの説明文
サムネイル画像	任意	File	画像ファイル (アップロード時にプレビュー表示)
公開設定	必須	Radio	公開 / 非公開 (デフォルト: 公開)

### 活動ブロック (リスト項目)

項目名	必須	タイプ	備考
活動タイトル	任意	Text	デフォルト: "新しい活動"
種別	必須	Select	教材 (Resource) / テスト (Test)
コンテンツを選択	必須	Select	種別に応じた登録済みコンテンツを選択
キーワード	任意	Text	カンマ区切りで入力

## 7.3. 問題作成画面 (/teacher/create)

### 共通項目

項目名	必須	タイプ	備考
問題文	必須	Textarea	Markdown 対応
スキル評価基準	任意	Select	
スキル領域	任意	Select	選択した基準に基づく領域
難易度	任意	Select	初級 / 中級 / 上級
タグ	任意	Text	自由入力

### モード別項目

モード	項目名	必須	備考
多肢選択	選択肢	必須	2つ以上追加。正解の選択肢にチェックを入れる。
穴埋め	正解キーワード	必須	完全一致で判定
プログラミング	模範解答コード	必須	Python コード
プログラミング	テストケース	必須	入力(stdin)と期待される出力(stdout)のペア

## 7.4. テストセット管理画面 (/teacher/manage/test-sets)

項目名	必須	タイプ	備考
タイトル	必須	Text	テストセットの名称
説明	任意	Textarea	受講者に表示される説明
制限時間	必須	Number	分単位 (デフォルト: 60)
合格点	必須	Number	0-100 点 (デフォルト: 60)
公開設定	任意	Checkbox	チェックで公開
紐づけるスキル評価基準	任意	Select	分析に使用
関連講義資料	任意	Select	分析・復習提案に使用
問題選択	必須	Checkbox List	スキル領域ごとにグループ化された問題リスト

### 7.5. 教材管理画面 (/teacher/resources)

項目名	必須	タイプ	備考
タイトル	任意	Text	未入力時はファイル名が使用される場合あり
ファイル	必須	File	PDF, MP4, Office ファイル等

### 7.6. 受講者管理画面 (/teacher/manage/students)

項目名	必須	タイプ	備考
氏名	必須	Text	表示名
メールアドレス	必須	Email	ログイン ID として使用
パスワード	必須	Password	新規登録時のみ必須（編集時は空欄で変更なし）

## 8. エラーメッセージ

### 8.1. 認証・権限エラー

メッセージ例	発生条件	対処法
メールアドレスまたはパスワードが間違っています。	ログイン時に誤った認証情報を入力	正しいメールアドレスとパスワードを確認して再入力してください。
この操作を行うには認証が必要です。 / Authentication is required.	未ログイン状態で保護された操作を実行	再度ログインしてください。
この操作は先生 (teacher) のみ許可されています。	受講者アカウントで教員用機能にアクセス	教員アカウントでログインし直してください。

### 8.2. 入力バリデーションエラー

メッセージ例	発生条件	対処法
コース名は必須項目です。	コース作成・編集時にタイトルが空	コース名を入力してください。
メールアドレス、パスワード、表示名は必須です。	受講者登録時に必須項目が不足	全ての必須項目を入力してください。
ファイルをアップロードしてください。	教材保存時にファイルが未選択	アップロードするファイルを選択してください。

### 8.3. AI 生成・外部連携エラー

メッセージ例	発生条件	対処法
AI response did not contain a valid JSON array.	Gemini が不正な形式のデータを返却	一時的な AI の不調です。数分待って再試行してください。
検証サーバーへの接続中にエラーが発生しました。	Executor Service (Python 実行環境) が応答しない	システム管理者に連絡するか、しばらく待って再試行してください。
AI フィードバックの生成に失敗しました。	Gemini API のエラーまたはタイムアウト	ネットワーク状況を確認し、再試行してください。
Gemini API キーが設定されていません。	サーバー側の環境変数設定ミス	システム管理者に連絡してください。

### 8.4. システム・通信エラー

メッセージ例	発生条件	対処法
コースの保存に失敗しました。	Firestore への書き込み失敗 (オフライン等)	ネットワーク接続を確認し、再試行してください。

受講者リストの取得に失敗しました。	Firestoreからの読み込み失敗	ページをリロードするか、ネットワークを確認してください。
サーバー内部で致命的なエラーが発生しました。	予期せぬバックエンドエラー	コンソールログを保存し、開発者/管理者に報告してください。

## 9. ファイル一覧

### 9.1. 入力ファイル (ユーザーアップロード)

用途	ファイル種別	形式	保存先 (Storage)	関連画面
コースサムネイル	画像	JPG, PNG, etc.	course_thumbnails/{uuid}_{filename}	コース編集画面
講義資料	ドキュメント/動画	PDF, MP4, Office, CSV, etc.	resources/{timestamp}_{filename}	教材管理画面
評価基準インポート	データバックアップ	JSON	N/A (クライアントサイドで解析・DB登録)	スキル管理画面
問題データインポート	データバックアップ	JSON	N/A (クライアントサイドで解析・DB登録)	問題バンク画面

### 9.2. 出力ファイル (エクスポート/ダウンロード)

用途	形式	生成方法	トリガー	関連画面
評価基準エクスポート	JSON	クライアント生成 (data:text/json)	[エクスポート] ボタン	スキル管理画面
問題データエクスポート	JSON	クライアント生成 (data:text/json)	[全件エクスポート] ボタン	問題バンク画面
コース進捗データ	JSON	クライアント生成 (data:text/json)	[エクスポート (JSON)] ボタン	進捗管理画面
受講者リスト	CSV	react-csv ライブラリ	[エクスポート (CSV)] ボタン	受講者管理画面

## 10. 外部インターフェース仕様 (API)

### 10.1. Cloud Functions API (Frontend -> Backend)

#### [F-01] generateTest

- **概要:** Gemini を使用してテスト問題を生成する。
- **認証:** 必須
- **リクエスト:**
  - {
  - "topic": "Python の基礎",
  - "numQuestions": 3,
  - "questionType": "programming" | "multiple-choice" | "fill-in-the-blank",
  - "language": "python",
  - "functionName": "calculate\_sum" // プログラミング問題のみ
  - }
- **レスポンス:**
  - {
  - "questions": [
  - {
  - "type": "programming",

- "text": "...",
- "initialCode": "...",
- "solutionCode": "...",
- "testCases": [...]
- }
- ]
- }

#### [F-02] submitTest

- **概要:** テストの回答を提出し、採点を行う。
- **認証:** 必須
- **リクエスト:**
- {
- "testId": "test\_set\_document\_id",
- "answers": {
- "question\_id\_1": "user\_answer\_1",
- "question\_id\_2": 2 // 選択肢インデックス
- }
- }
- **レスポンス:**
- {
- "submissionId": "submission\_document\_id",
- "score": 85,
- "passed": true
- }

#### [F-03] getAIFeedback

- **概要:** 不正解だったプログラミング問題に対して AI フィードバックを取得する。
- **認証:** 必須
- **リクエスト:**
- {
- "studentCode": "...",
- "questionText": "...",
- "solutionCode": "...",
- "testCases": [{"callCode": "...", "expectedOutput": "..."}] // 失敗したケース
- }
- **レスポンス:**
- {
- "feedback": "Markdown 形式のフィードバックテキスト..."
- }

#### [F-04] suggestRecipe (Recipe Suggester)

- **概要:** 指定された食材に基づいてレシピを提案する。
- **認証:** 必須
- **リクエスト:**
- {
- "requiredIngredients": [{"name": "鶏肉", "amount": "300g"}],
- "optionalIngredients": [{"name": "玉ねぎ", "amount": "1 個"}]
- }
- **レスポンス:**
- {
- "recipes": [
- {
- "name": "鶏肉のトマト煮込み",
- "ingredients": [...],

- "instructions": ["..."],
- "nutrition\_estimate\_per\_2\_servings": {...},
- "image\_prompt": "..."
- }
- ]
- }

## 10.2. Executor Service API (Cloud Functions -> Cloud Run)

- **概要:** 信頼されていない Python コードをサンドボックス環境で実行する。
- **エンドポイント:** POST /execute
- **リクエスト:**
  - {
  - "language": "python",
  - "code": "def add(a, b): return a + b",
  - "testCases": [
  - {"setupCode": "", "callCode": "add(1, 2)", "expectedOutput": "3" }
  - ]
  - }
- **レスポンス:**
  - {
  - "passedCount": 1,
  - "totalCount": 1,
  - "results": [
  - {"status": "Passed", "output": "3", "expected": "3" }
  - ]
  - }

## 11. バッチ処理一覧

処理名称	トリガー/スケジュール	実行内容	関連コレクション
ダッシュボード集計 (aggregateMetricsScheduled)	毎日 0:00 (JST) 00***	全期間の提出データとユーザー情報を集計し、以下の指標を更新する。 1. 日次提出数: dailySubmissions 2. スキル習熟度: lowProficiencySkills (正答率の低い順) 3. 受講者ランキング: studentRanks (テストセット別スコア順) 4. テスト合格率: passFailRate	testSubmissions users questionBank dashboard_metrics

### 11.1. 補足: スケジュール実行の仕組み

- **Firebase Google Cloud Scheduler** を利用。
- タイムゾーンは Asia/Tokyo に設定。
- 手動実行も可能 (教員用ダッシュボードの「集計実行」ボタン - > callAggregateDashboardMetrics)。

## 12. アプリケーション仕様

### 12.1. 動作推奨環境

項目	内容
PC ブラウザ	Google Chrome (最新版), Microsoft Edge (最新版), Firefox (最新版), Safari (最新版)
SP/タブレット	レスポンス対応 (Tailwind CSS により実装)
画面解像度	PC: 1280px 以上推奨 (管理画面), SP: 375px 以上

### 12.2. 言語・タイムゾーン・国際化 (i18n)

項目	仕様
対応言語	日本語 (JA) のみ (ハードコード実装)
タイムゾーン	Asia/Tokyo (JST)
日時フォーマット	YYYY/MM/DD または YYYY-MM-DD 形式を標準使用
日時処理ライブラリ	date-fns v3.6.0

### 12.3. 環境変数・構成値 (Configuration)

環境変数は .env ファイルを使用せず、以下の方式で管理する。

分類	管理場所	内容・例
Frontend Config	frontend/src/firebase.js	Firestore SDK 初期化情報 (API Key, AuthDomain, ProjectID 等)
Backend Config	functions.config()	gemini.key (Gemini API キー), executor.url (コード実行基盤 URL)

### 12.4. 依存ライブラリ (Dependencies)

主要な依存パッケージとそのバージョン

カテゴリ	パッケージ名	バージョン	用途
Core (Frontend)	react	^18.3.1	UI ライブラリ
	vite	^5.3.4	ビルドツール
	react-router-dom	^6.25.1	ルーティング
Firebase	firebase	^10.12.3	Firebase SDK
	react-firebase-hooks	^5.1.1	React Hooks for Firebase
UI/UX	tailwindcss	^3.4.6	CSS フレームワーク
Editor	@uiw/react-codemirror	^4.23.0	コードエディタ
Chart	recharts	^3.3.0	グラフ描画 (ダッシュボード)

以上

---

文部科学省委託 専修学校による地域産業中核的人材養成事業——外人口減少地域の職業人材を確保するための専修学校振興プログラム

人口減少地域を支えるAI補完型ITエンジニア養成プログラムの開発

## 事業報告書 令和7年度

---

発行日——2026年3月1日 初版第1刷発行

---

発行者——学校法人電子学園 日本電子専門学校

〒169-8852 東京都新宿区百人町1-25-4  
TEL 03-3363-7761 URL <https://www.jec.ac.jp/>

---

本報告書は、文部科学省の教育政策推進事業委託費による委託事業として、  
学校法人電子学園 日本電子専門学校が実施した令和7年度「専修学校の国際化推進事業」の成果を取り纏めたものです。

---