日本総研
The Japan Research Institute, Limited

AI駆動開発とシステムズエンジニアリングへのAI適用研究の動向

(株)日本総合研究所
先端技術ラボ

2025年4月8日

本資料に関するお問い合わせ 市原紘平 (ichihara.kohei@jri.co.jp)

本資料は作成日時点で一般に信頼できると思われる情報に基づき弊社が作成したものですが、情報の正確性・完全性を保証するものではありません。記載内容は経済情勢等の変化により変更されることがあります。
本資料の情報に起因してご閲覧者様及び第三者に損害が発生したとしても、執筆者、執筆にあたっての取材先及び弊社は一切責任を負わないものとします。本資料の著作権は株式会社日本総合研究所に帰属します(引用部分を除く)。

© 2025 The Japan Research Institute, Ltd. (except where indicated)

サマリ -AI駆動開発とシステムズエンジニアリングへのAI適用研究の動向-

- AI駆動開発への期待
  - 生成AIの活用によって、プログラミングが大きく変わることが多くの専門家により指摘されている。
  - CHOP(チャット指向プログラミング)のような新しいアプローチを取り入れることで、ジュニア開発者が高い生産性を発揮できる期待もある。
- AI駆動開発への懸念
  - 一方で逆に、ジュニア開発者が任されるような小さなタスクがAIコーディングエージェントに代替されてしまうことで、成長機会が喪失されるといった懸念もある。
  - また、表面上正しく動いているように見えるバグを含んだコードが検証されずにリリースされてしまう恐れもあり、シニア開発者などによる適切な検証が必要である。
- AIコーディングエージェントの台頭 ~開発者が「運転席」を譲る可能性~
  - Clineの様に、コードの生成に留まらずコマンド実行やその結果に基づいた修正まで自動で行うAIコーディングエージェントが登場した。CursorのagentモードやCopilot Agentのように、開発者支援中心だったツールにもエージェント機能が追加されてきている。
  - 人間主体のコーディングに対するコードの提案や修正の推薦といった支援の役割から、AIエージェント主体のコーディング結果を人間が確認、修正するスタイルへの移行の予測も出ており、今後の技術進展が注目される。
- AI駆動開発時代におけるアーキテクチャモデリング
  - AI駆動開発が取り入れられる中でも、次のような点で引き続きアーキテクチャモデリングは重要と考えられる。
    - ✓ 関係者間での共通理解の醸成
    - ✓ 要件充足の追跡性・信頼性、構成の確認可能性、設計の説明可能性
  - グラフィカル・モデリング言語によるMBSEによって、人間が視認でき、機械可読性も高いモデリングを行うことが可能である。更に、こうしたアーキテクチャモデリング自体も生成AIにより効率化可能なことが、研究により確認されている。
  - 適切な指示出しやタスク分割が必要なAI駆動開発の利便性や信頼性を、生成AIによって効率化されたアーキテクチャモデリングにより更に向上させることが期待される。

© 2025 The Japan Research Institute, Ltd. (except where indicated)

日本総研 The Japan Research Institute, Limited

目次・用語集 -AI駆動開発とシステムズエンジニアリングへのAI適用研究の動向-

章	節	頁
1. AI駆動開発の動向	1.1 AI駆動開発の概要	p.4
	1.2 AIを活用した開発者支援ツール例	p.5
	1.3 AI駆動開発で注目される概念	p.7
	1.4 AI駆動開発の発展見通し	p.8
2. システムズエンジニアリングへのAI適用研究の動向	2.1 大規模システム開発へのAI支援の適用可能性	p.10
	2.2 MBSEとエンタープライズアーキテクチャ(EA)	p.11
	2.3 アーキテクチャモデリングへのAI適用研究事例	p.13
3. まとめ	3.1 まとめと考察	p.15

用語	意味	頁
LLM (Large Language Model)	• LLM(大規模言語モデル)とは、文章の理解や生成ができるAIモデルのこと。自然言語処理技術を利用し、機械学習の一種であるディープラーニングによって大量のテキストデータを学習している。	p.4
CPS (Cyber-Physical System)	• CPS(サイバーフィジカルシステム)とは、現実(フィジカル)の情報を、コンピュータ(サイバー)に取り込み、分析等を行った上で現実へフィードバックし、最適な結果を導き出す、サイバー空間とフィジカル空間が緊密に連携するシステム。	p.11
生成AI (Generative AI)	• 生成AI(ジェネレーティブAI)とは、学習したデータをもとに新しいデータや情報を生成する人工知能のこと。文章や音声、画像、動画などのコンテンツを生み出すことができる。LLMが利用されている。	p.13
Few-shot Learning	• Few-shot Learning(フューショット学習)とは、AIモデルに対し少数の例文をデモンストレーションとして提示するだけで、新たに与えたに問題に対し正確な予測を行えるようにする技術。	p.13
SLM (Small Language Model)	• SLM(小規模言語モデル)とは、予測や判断に用いるパラメータ数がLLMに比べ少ないAIモデルのこと。パラメータ数が多いほど高性能となる反面、開発コスト等も増大するため、高性能で安価なSLMを模索する動きもある。	p.13

© 2025 The Japan Research Institute, Ltd. (except where indicated)

1. AI駆動開発の動向

1.1 AI駆動開発の概要
1.2 AIを活用した開発者支援ツール例
1.3 AI駆動開発で注目される概念
1.4 AI駆動開発の発展見通し

© 2025 The Japan Research Institute, Ltd. (except where indicated)

日本総研 The Japan Research Institute, Limited

1.1 AI駆動開発の概要

- AI駆動開発(AI-driven development)とは、LLM(大規模言語モデル)等のAIによる支援を活用したソフトウェア開発アプローチである。
- LLMを用いることで、自然言語(日本語や英語等の人間が話す言語)の指示を元に、プログラムコードを生成することなどができる。
AIとのチャットを通じてコードを書いていく「Chat-Oriented Programming(CHOP)」も同様の概念として存在する。
- AIはソフトウェア、システム開発の様々な工程に活用可能性があるが、近年進歩の著しいLLMはテキストデータを扱うことから、プログラムコードの自動生成や記述支援、及びシステムに関連したドキュメント作成の効率化や自動化などに注目が集まる。

開発者は指示や要望をテキスト入力でAIへ伝える
↓
AIにより成果物や回答が生成

AIによる開発支援の機能例

機能例	内容
テキスト指示による コード生成	● 自然言語のテキストによる指示を元に プログラムコードが生成、または編集される。
画像による コード生成	● スクリーンショットや手書きのイメージから コードが生成される。
コードに関する 質疑応答	● プログラムの処理内容についての質問をすると、 自然言語で説明してくれる。
自動バグ修正	● バグを自動で検出し、修正案を提示してくる。

© 2025 The Japan Research Institute, Ltd. (except where indicated)

日本総研 The Japan Research Institute, Limited

1.2 AIを活用した開発者支援ツールの例 -Cursor(カーソル)-

- Cursor(カーソル)は、AIによる支援を利用できるプログラムコードエディタ(AIコーディングエディタ)。機能の限られた無料版の利用も可。
- Microsoft社によるオープンソースの著名なエディタであるVisual Studio Code(VS Code)をベースに開発。
- 開発元のAnysphere社はMIT卒業生らによって創業され、OpenAIからの800万ドルを含む計1100万ドルの資金調達を達成。[1]

Cursorの概要

AIモデルは選択可能[2]

- CursorはAI機能を提供するAIモデルを選択できる。
応答が通常速く有料プランでは無制限に使用できる「無料モデル」と、
高性能でリクエスト回数に制限がある「プレミアムモデル」に分類される。

Cursorの機能例

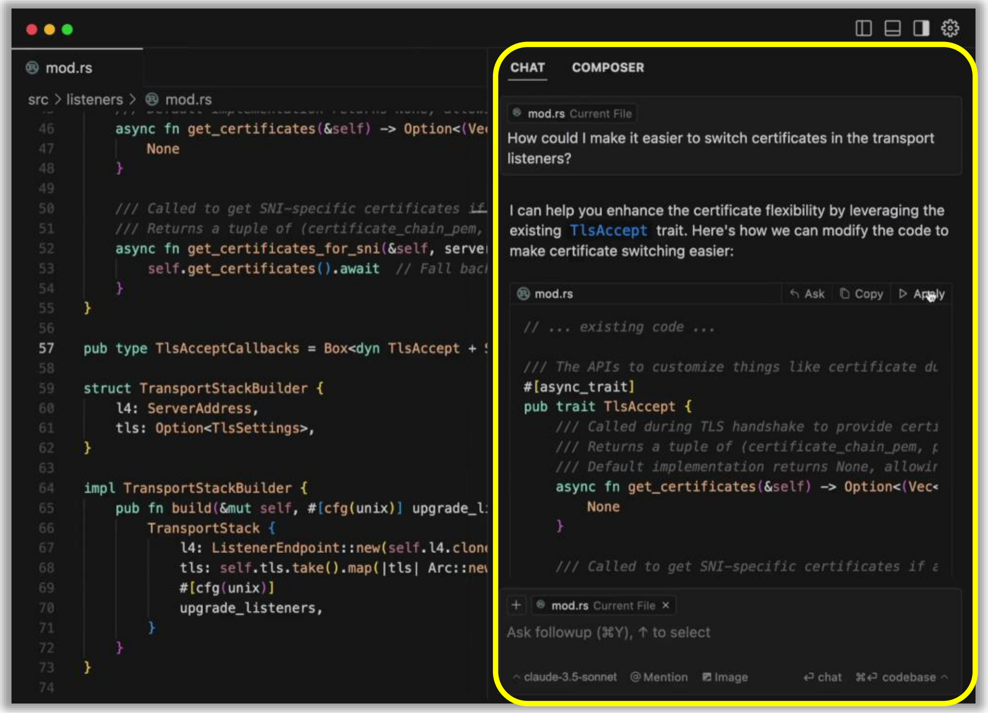
自動生成	自然言語で所望の関数等を伝えてコード生成
チャット	Chatウィンドウに質問やリクエストを入力して活用
自動補完	Tabキーを押すとコードがオートコンプリートされる
Auto-Debug	ターミナルで発生したエラーの原因を自動検出
エージェント	コマンド実行や修正の反復等を自動で行う新機能

利用プラン

利用プラン	Hobby	Pro	Business
料金	無料	$20/月	$40/月
低速リクエスト(プレミアムモデルへの)	50回/月	無制限	無制限
高速リクエスト(プレミアムモデルへの)	×	500回/月	500回/月
プライバシーモード	×	×	○

Cursorの画面[3]

画面右側がAIとのやり取りを行う領域



[1]Anysphere.inc, "We raised $8M from OpenAI.", Anysphere.inc, https://anysphere.inc/blog/openai-fund (accessed on 2025/3/19)
[2]Anysphere.inc, "Cursor – Models", Cursor.com, https://docs.cursor.com/settings/models (accessed on 2025/3/19)
[3]Anysphere.inc, "Cursor - The AI Code Editor", Cursor.com, https://www.cursor.com/ja (accessed on 2025/3/19)

© 2025 The Japan Research Institute, Ltd. (except where indicated)

日本総研 The Japan Research Institute, Limited

1.2 AIを活用した開発者支援ツールの例 -Cline(クライン)-

- Cline(クライン)は、Visual Studio Code(VS Code)等のエディタの拡張機能として動作するAIを活用した開発支援ツールである。コーディングの支援だけでなく、タスクやエラー解決等の自動実行も行うAIエージェントとして機能する。
- 開発元のCline Bot Inc.は開発者コミュニティを尊重しておりGitHubにソースコードが公開されている他、Discordコミュニティ等がある。
- Cline自体の利用は無料であり、ChatGPT等の主要なAIプロバイダを切り替えて使うことができる。(AIのAPI課金は別途発生しうる)

Clineの概要

強力なエージェント機能が特徴

- ➢ パッケージインストールやビルド、デプロイ、テスト実行等、ターミナルコマンドを用いた様々なタスクが可能。(コード生成等の一般的な機能も有する)

Webブラウザ操作に対応

- ➢ Webページ上でのクリック等の操作やスクショ取得等の自律実行が可能。

MCP(Model Context Protocol)による拡張性

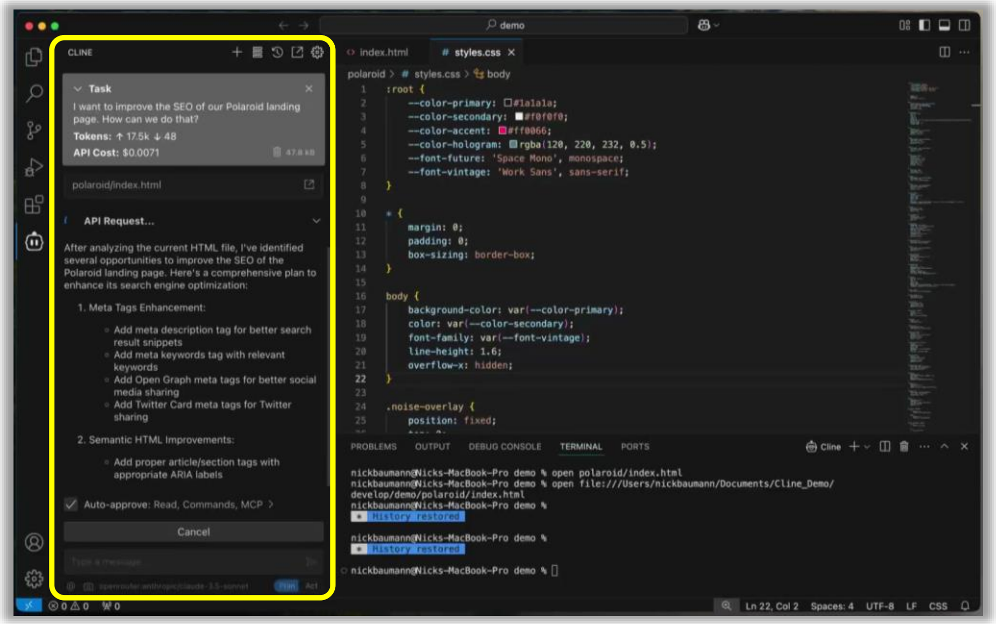
- ➢ MCPとは、Anthropic社によって開発されオープンソース化された、AIモデルをその他のシステムに接続するための標準規格。
- ➢ Clineは、MCPにより外部データソースやツールと接続が可能。

オープンソースのため複数の派生版(フォーク)が存在

Cline	ベースとなるオリジナル版。GitHub Star数最多。
Roo Code	更新頻度が高く、実験的機能を積極的に取り入れ
Bao Cline	Roo Cline同様独自の機能を取り入れ
Cool Cline	他のフォーク版の良点をバランス良く集約した統合版

Clineの画面[1]

画面左側がVS Code上で稼働するCline



[1]Cline Bot Inc., “Cline - AI Autonomous Coding Agent for VS Code”, cline.bot, https://cline.bot/ (accessed on 2025/3/19)

© 2025 The Japan Research Institute, Ltd. (except where indicated)

日本総研 The Japan Research Institute, Limited

1.3 AI駆動開発で注目される概念

- AI駆動開発では、開発に関するコンテキスト(背景情報)をAIに認識させることでより望ましい結果(ソースコード等)を得ることができる。
- また、関係ファイルを参照させて所定のワークフローを実行することで、開発管理ドキュメントの生成等も可能である。
- このようにAI駆動開発では開発に関係する情報をAIツールが参照できるようにしておくことが望ましく、以下のような概念も注目される。

Docs as Code, Docs from Code

Docs as Code(DaC)

- 『Docs as Codeとは、コードと同じツールを使用してドキュメントを作成すべきという哲学』[1]
- 製品に関するユーザ向けのドキュメントの更新を、プログラムコードの更新と同じバージョン管理やデリバリのパイプラインに乗せることで、ドキュメントの更新漏れや、実態(コード)との乖離を防ぐことができる。
- WordやExcel等の特定のソフトに依存した形式ではなく、コードと同じ様にプレーンテキストで記述し、Git等でバージョン管理する。

Docs from Code(DfC)

- Docs from Code(DfC)は、DaCの考え方を発展させ、ドキュメント作成を自動化するもの。
- プロジェクト内のソースコードや設定ファイルを解析することで、製品ドキュメントを生成する。[2]
- 文書生成にはLLMによる自動化は有効と考えられ、AI駆動開発が注目される中、DfCの導入も併せて導入することが有効な可能性がある。

Project as Code

Project as Code(PaC)とは

- 『システム開発プロジェクトの全てをコード管理・バージョン管理していこうという取り組み』[3]
- 開発ルール、コード規約、フォルダ構成、ファイル構成、フレームワーク・アーキテクチャの原則など、プロジェクトのポリシーに沿ったコードを生成するためにAIに参照させるべき情報は多々あり、AIによる開発支援ツールでは指定の設定ファイルに記述することになっている。
- PaCでは、上記のようなコードの形式的な品質を担保するための情報のみならず、開発要件を十分に充足しているかという観点での品質を担保するため、コンテキストをAIに適切に与えることを目指す。『そのためAIが理解しやすい形式でプロジェクトの全ての情報をコード管理していく必要がある』[3]とされる。
- 提唱者によれば、上記を実現するためPaCでは、人の「こうしたい」といった頭の中の思考を直接、機械可読性の高いYAMLファイルやマークダウン記法のテキストファイルの形式にまず落とし込む。その後、それらのファイルから合意形成のためのドキュメント(要件定義書や設計書)を生成する。

[1]Write the Docs, “Docs as Code — Write the Docs”, writethedocs.org, https://www.writethedocs.org/guide/docs-as-code/ (accessed on 2025/3/19)
[2]Radhika Puthiyetath, “Docs-as-Codeの採用：ハッカソンから本番環境まで – Sysdig”, SYSDIG SECURE, https://sysdig.jp/blog/adopting-docs-as-code/ (accessed on 2025/3/19)
[3]熊井悠, “AI駆動開発では「Project as Code」が超重要である！”, Qiita, https://qiita.com/kumai_yu/items/0aa2fc294f8e1347e36c (accessed on 2025/3/19)

© 2025 The Japan Research Institute, Ltd. (except where indicated)

1.4 AI駆動開発の発展見通し

- 多くの著名なエンジニアや研究者がAIによってソフトウェア開発が大きく変化すると予想している。
- AI駆動開発においてはAIの能力を引き出すために、適切にタスクを設計しAIに情報を与えることが重要と考えられる。

著名研究者らの見通し

Tim O'Reilly 『私たちが知っているプログラミングの終焉』[1]

- 「AIがプログラマに取って代わることはないが、仕事内容は変わる。」
- 「CやJavaのような高水準言語がアセンブリ言語に取って代わったように、現在の変化は最新の再発明の始まりである。」

Chip Huyen 「AI Engineering」著者

- 『人口のごく一部しか読み書きができなかった昔は、書くことは知的なものと考えられていた。今日では、「書くこと」という言葉は、もはやこの物理的な行為ではなく、アイデアを読み取り可能な形式に整理するというより抽象化された行為を指す。同様に、コーディングという物理的な行為が自動化されると、「プログラミング」の意味は、アイデアを実行可能なプログラムに整理する行為を指すように変わる。』[1]

Steve Yegge 『頑固な開発者の死』[2]

- 『チャットベースのプログラミング(CHOP)を主要な手段として採用しないと、取り残されてしまう。』
- 『CHOPは既に上手にできる作業を大幅に高速化できるとは限らない。しかし、あまり得意ではない作業、つまり、手間がかかりすぎるとわかっていて先延ばしにしていた作業は、驚くほど高速化できる。』
- 「リーフノード*より上のタスクグラフを管理するのはAIではなく人間であると考えている。」(*リーフノード:大きな幹のタスクから枝分かれした個別のタスク)

AI駆動開発における課題・ポイント

AIモデル使用による注意点

- AIモデルを使用することで開発に関する情報が外部に送信されてしまう。自社サーバ内で稼働するローカルAIモデルの使用等で回避可能である。

コンテキストウィンドウの限界

- AIモデルの認識できるコンテキストの量には限界がある。また、不必要な情報を与えてもAIモデル使用のコストを無駄に増加させることになる。
- より良い結果を得るためには、エンジニアはAIに与える情報の選定を工夫する必要がある。(c.f. 『AI共同開発者が作業するコンテキストの大部分を人間が提供しなければならない』[1] Sam Schillace, Deputy CTO at Microsoft)

設計と実装フェーズの分離 (Architect/Editorアプローチ)[3]

- コーディングタスクを完了するために「アーキテクトモデル」と「エディタモデル」の2つを使用することで良好な結果(テストを通過するコード)が出るという報告がある。[3]
- 「アーキテクトモデル」には問題を解決する方法を説明することが求められ、推論能力に優れたAIモデル(推論モデル)を用いる。
- 「エディタモデル」は「アーキテクトモデル」から提案されたソリューションを一連の適切な形式のコード編集に変換する。(コスト、速度等に基づいて適切なモデルを選択可能)

[1]Tim O'Reilly, "The End of Programming as We Know It", O'Reilly, https://www.oreilly.com/radar/the-end-of-programming-as-we-know-it/ (accessed on 2025/3/19)
[2]Steve Yegge , "The Death of the Stubborn Developer", Medium, https://steve-yegge.medium.com/the-death-of-the-stubborn-developer-b5e8f78d326b (accessed on 2025/3/19)
[3]Aider AI, "Separating code reasoning and editing", aider, https://aider.chat/2024/09/26/architect.html (accessed on 2025/3/19)

© 2025 The Japan Research Institute, Ltd. (except where indicated)

2. システムズエンジニアリングへのAI適用研究の動向

2.1 大規模システム開発へのAI支援の適用可能性
2.2 MBSEとエンタープライズアーキテクチャ(EA)
2.3 アーキテクチャモデリングへのAI適用研究事例

© 2025 The Japan Research Institute, Ltd. (except where indicated)

2.1 大規模システム開発へのAI支援の適用可能性

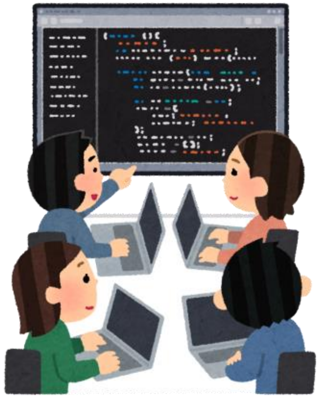
- 大規模システム開発では、複雑な仕様や多くの依存関係が存在し、全体は多くのコンポーネントから構成される。
- AIがこうした全てのコンテキストを把握した上でコード生成を行うことは現状困難であるため、コンポーネントやタスクへの分解等をアーキテクチャ(設計、構成)的な観点に基いて適切に行うことは、AI駆動開発を活用する上でも依然として重要である。
- アーキテクチャを適切に設計するための手法としてシステムズエンジニアリングが存在し、同手法へもAI支援を適用しようとする研究もある。

業務要件

ステークホルダの意向　ユーザニーズ

その他
仕様策定に関わる情報

アーキテクチャ設計
(コンポーネントや個別タスクへ整理)



AI駆動開発

[注目される研究領域]

- アーキテクチャ設計へのAI活用
- AI開発エージェントのマネジメント (AIにとって適切なタスク分解)　他

[注目される研究領域]

- AIへの適切なコンテキスト付与方法
- コードによるドキュメント作成・管理　他

本章ではアーキテクチャ設計の手法(MBSE,EA:次頁で解説)と
同手法へのAI(LLM)適用の研究について紹介する。

© 2025 The Japan Research Institute, Ltd. (except where indicated)

2.2 MBSEとエンタープライズアーキテクチャ(EA)

- アーキテクチャ設計の手法として、MBSE(Model Based Systems Engineering)やエンタープライズアーキテクチャ(EA)が注目される。
- いずれの手法でも、関係者間(企画部署とエンジニア等)のコミュニケーションや意思決定の質の向上を図るため、「モデル」を用いる。
- 「モデル」はコンポーネント間の関係などを視覚的に表現する方法を提供し、専用の「グラフィカル・モデリング言語」を用いて記述する。

MBSE

MBSE(Model Based Systems Engineering)とは

- 『モデルを使用して、構想・設計から検証および妥当性確認、廃棄に至るまでの、システムのライフサイクル全体をサポートする手法』[1]。
  従来のテキストドキュメントベースの手法に比べ以下のようなメリットがある。

MBSEのメリット

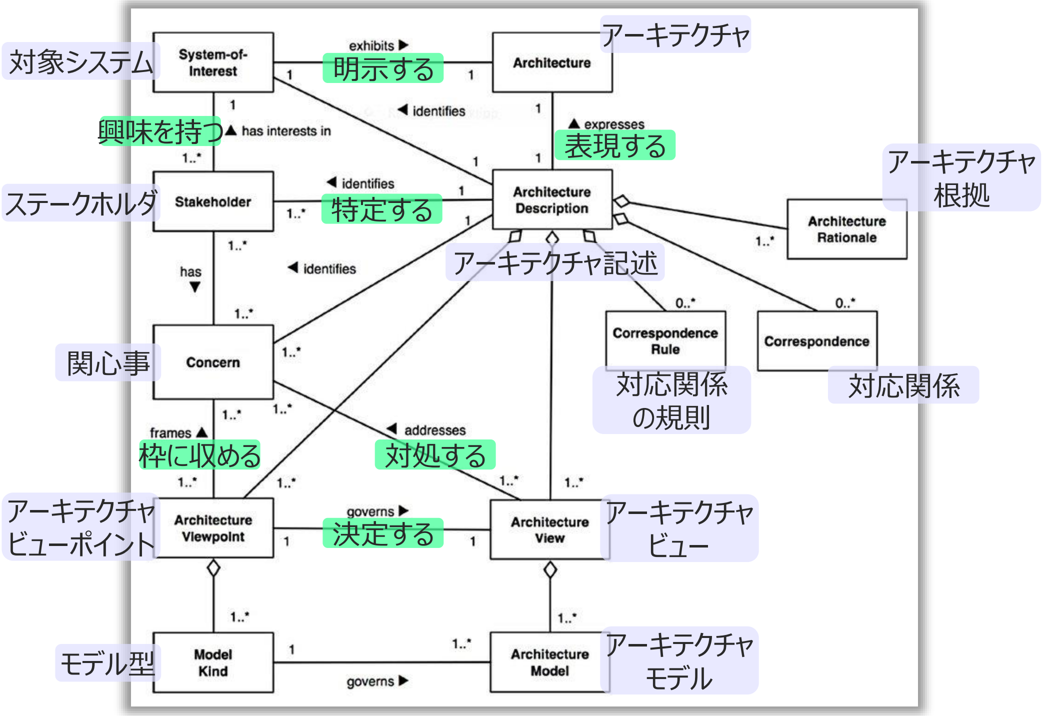
コミュニケーション強化	視覚的なモデルにより関係者の理解を促進
品質向上	要件から仕様の追跡性が向上し品質向上
効率向上	追跡性により早期の問題検出と修正が可能
複雑システムへの対応	システム同士の関係性等の表現にも対応

MBSEで使用されるツール

モデリング言語	システムのコンポーネントや動作を視覚化する図を記述するためのモデリング言語
モデリングツール	モデルの編集等に使用するソフト(次頁詳細)
要件管理ツール	システムの要件を追跡し、設計に反映されているかを確認するためのソフト
分析ツール	システムの正常な動作をシミュレーション等で確認するためのソフト(特にCPSなどで使用)

エンタープライズアーキテクチャ(EA)

EAは『組織全体の業務とシステムを共通言語と統一的手法でモデル化し最適化を図る設計手法』[2]。ISO 42010のような国際標準に基づきアーキテクチャを記述する(下図)。(詳細:弊社発行 EAの活用動向とArchiMateの概説)



[1]Amanda McGrath、Alexandra Jonker, "モデルベース・システムズ・エンジニアリング（MBSE）とは", ibm.com, https://www.ibm.com/jp-ja/topics/model-based-systems-engineering (accessed on 2025/3/19)
[2]デジタル庁, "データ環境整備のためのアーキテクチャ管理 導入実践ガイドブック", デジタル社会推進標準ガイドライン, 2024/3/27, https://www.digital.go.jp/assets/contents/node/basic_page/field_ref_resources/4e06cf86-4403-47a8-bd67-52f8ea5ec66b/cf2a7c71/GIF-1.39.zip (accessed on 2025/3/19)

© 2025 The Japan Research Institute, Ltd. (except where indicated)

[参考]モデリング対象に応じたグラフィカルモデリング言語

- アーキテクチャモデリングでは、対象に応じて各種モデリング言語とツールを使用する。

モデリング対象	モデリング言語	言語の開発組織	モデリングツール例
ソフトウェアやデータモデル	**UML** (Unified Modeling Language) ISO19505として標準化	The Object Management Group(OMG) IT業界の非営利標準化コンソーシアム	• PlantUML 個人によるOSS(GNU GPL) 人間可読なドメイン固有言語で図を表記するためGit等での管理が可能。 • Mermaid OSS(MIT License) Markdown形式で図を表記するため、Git等での管理が可能。 • astah* UML ((株)チェンジビジョン) 他
システム	**SysML** (Systems Modeling Language)	OMGとINCOSE*による共同 UMLのサブセットを拡張したもの (*システムズエンジニアリングに関する非営利専門組織)	• Modelio (Modeliosoft社) OSS(GNU GPL) • Papyrus (仏 原子力・代替エネルギー庁) OSS(EPL) • MagicDraw (No Magic社) 他
エンタープライズアーキテクチャ	**ArchiMate**	The Open Group(TOG) UNIX(商標)やTOGAFを管理する業界団体	Archi 個人によるOSS(MIT License)
業務プロセス	**BPMN** (Business Process Modeling and Notation) ISO19510として標準化	The Object Management Group(OMG) IT業界の非営利標準化コンソーシアム	• Camunda BPM (Camunda Services GmbH) OSS(Apache 2.0) • IM-BPM ((株)NTTデータイントラマート) • Pega Infinity (Pegasystems Inc.) 他

© 2025 The Japan Research Institute, Ltd. (except where indicated)

2.3 アーキテクチャモデリングへのAI適用研究事例

- ChatGPTの登場以降の生成AIへの大きな注目もあり、アーキテクチャモデリングへLLM(生成AI)を活用しようという研究は活発に行われており、有効性を示す結果が報告されている。

MBSEへの生成AI適用研究

タイトル	生成AIを活用してMBSEモデルを作成、変更、およびクエリする (Leveraging Generative AI to Create, Modify, and Query MBSE Models)[1]
著者	米海軍大学院(NPS)教授ら (Ryan Longshore, Ryan Bell, Ray Madachy)
公開日	2024/4/30 (@21st Annual Acquisition Research Symposium)
研究概要	• 現在の**LLMがシステムモデリング言語(SysML v2)のモデルを生成、変更、およびクエリする能力を調査** • LLMに関連する将来のシステムズエンジニアリング研究の説明

- ChatGPT-3.5, ChatGPT-4, SysML v2仕様の知識等をRAGで与えたカスタムGPT(ChatGPT-4ベース)の3つを用いてプロンプトからモデルを正しく生成できるか実験。
- 自転車を題材として、自然言語による部品と構造の説明を元にGPT-4とカスタムGPTでは接続まで含めた正しいモデル化とグラフィカル表示ができた。(GPT-4ではFew-shot Learningを要した)
- 今後の研究の展望として、LLMのシステムモデリング能力を定量評価するベンチマークの開発が挙げられ、「SysEngBench」[2]等が進展中。
- 更なるモデリング能力向上のために、SysML v2モデル例のコーパスが必要になる可能性や、ドメイン固有のSLMが有効な可能性も考えられる。

ソフトウェアアーキテクチャモデリングへのLLM適用研究

タイトル	ユーザーストーリーからのUMLシーケンス図の自動導出 (Automated Derivation of UML Sequence Diagrams from User Stories)[3]
著者	Thompson Rivers Univ., Univ. of Prince Edward Island, Univ. of Saskatchewan (カナダ)
公開日	2024/9/22 (@ACM/IEEE 27th MODELS)
研究概要	• **ユーザーストーリーとして表現された自然言語の要件**から振る舞いモデル、特に**シーケンス図を生成するタスク**に対する**LLMの有効性を調査**

- ChatGPT-3.5、及び独自のルールベースの手法を用いて、ユーザーストーリーからシーケンス図を正しく生成できるか実験。
- 公開されているユーザーストーリーに関するベンチマークデータセット(題材は図書館及びレストランのマネジメントシステム等)を使用。生成された図の正確さをソフトウェア業界経験のある専門家が評価した。
- GPT-3.5を使用して生成された図は、オブジェクト間の相互作用やインタラクション表現の正確さ等の面で、10点中9点以上の高スコアとなり、ルールベースの結果をいずれも上回った。
- ソースコード、データセット等がGitHubで公開されている。(https://github.com/Jahan2021/From_User_Stories_to_SD)

[1]Ryan Longshore, Ryan Bell, Ray Madachy, "Leveraging Generative AI to Create, Modify, and Query MBSE Models", The Naval Postgraduate School, 2024/4/30, https://www.dair.nps.edu/bitstream/123456789/5113/1/SYM-AM-24-048.pdf (accessed on 2025/3/19)
[2]Ryan Bell, Ryan Longshore, Raymond Madachy, "Introducing SysEngBench: A Novel Benchmark for Assessing Large Language Models in Systems Engineering", The Naval Postgraduate School, 2024/4/30, https://dair.nps.edu/bitstream/123456789/5135/1/SYM-AM-24-072.pdf (accessed on 2025/3/19)
[3]Munima Jahan, Mohammad Mahdi Hassan, Reza Golpayegani, Golshid Ranjbaran, Chanchal Roy, Banani Roy, Kevin Schneider, "Automated Derivation of UML Sequence Diagrams from User Stories: Unleashing the Power of Generative AI vs. a Rule-Based Approach", MODELS '24: Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, 2024/9/22, https://www.researchgate.net/profile/Golshid-Ranjbaran/publication/384491393_Automated_Derivation_of_UML_Sequence_Diagrams_from_User_Stories_Unleashing_the_Power_of_Generative_AI_vs_a_Rule-Based_Approach/links/6707febee3c0600c7c98e604/Automated-Derivation (accessed on 2025/3/19)

© 2025 The Japan Research Institute, Ltd. (except where indicated)

3. まとめ

3.1 まとめと考察

© 2025 The Japan Research Institute, Ltd. (except where indicated)

日本総研 The Japan Research Institute, Limited

3.1 まとめと考察

- AI駆動開発への期待
  - 生成AIの活用によって、プログラミングが大きく変わることが多くの専門家により指摘されている。
  - CHOP(チャット指向プログラミング)のような新しいアプローチを取り入れることで、ジュニア開発者が高い生産性を発揮できる期待もある。
- AI駆動開発への懸念
  - 一方で逆に、ジュニア開発者が任されるような小さなタスクがAIコーディングエージェントに代替されてしまうことで、成長機会が喪失されるといった懸念もある。
  - また、表面上正しく動いているように見えるバグを含んだコードが検証されずにリリースされてしまう恐れもあり、シニア開発者などによる適切な検証が必要である。
- AIコーディングエージェントの台頭 ～開発者が「運転席」を譲る可能性～
  - Clineの様に、コードの生成に留まらずコマンド実行やその結果に基づいた修正まで自動で行うAIコーディングエージェントが登場した。CursorのagentモードやCopilot Agentのように、開発者支援中心だったツールにもエージェント機能が追加されてきている。
  - 人間主体のコーディングに対するコードの提案や修正の推薦といった支援の役割から、AIエージェント主体のコーディング結果を人間が確認、修正するスタイルへの移行の予測も出ており、今後の技術進展が注目される。

- AI駆動開発時代におけるアーキテクチャモデリング
  - AI駆動開発が取り入れられる中でも、次のような点で引き続きアーキテクチャモデリングは重要と考えられる。
    - ✓ 関係者間での共通理解の醸成
    - ✓ 要件充足の追跡性・信頼性、構成の確認可能性、設計の説明可能性
  - グラフィカル・モデリング言語によるMBSEによって、人間が視認でき、機械可読性も高いモデリングを行うことが可能である。更に、こうしたアーキテクチャモデリング自体も生成AIにより効率化可能なことが、研究により確認されている。
  - 適切な指示出しやタスク分割が必要なAI駆動開発の利便性や信頼性を、生成AIによって効率化されたアーキテクチャモデリングにより更に向上させることが期待される。

© 2025 The Japan Research Institute, Ltd. (except where indicated)

執筆者・先端技術ラボのご紹介

執筆者

市原 紘平
Ichihara　Kohei

シニア・リサーチャー
Senior Researcher

Web3研究に取り組み、2018年からセキュリティトークン等をSMBCグループ内で啓蒙。現在、事業企画への生成AI活用に興味。

■ 執筆レポート

- EAの活用動向とArchiMateの概説
- 自己主権型アイデンティティの動向と考察 ~企業のデジタルアイデンティティ戦略へ向け~
- デジタル証券とRWAトークンの動向　他

■ 学術論文

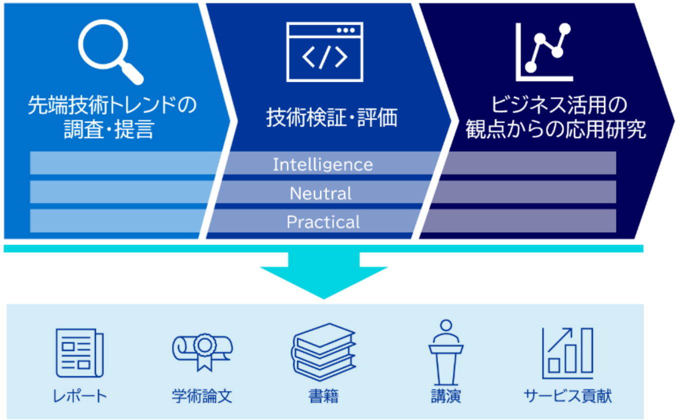
- Practices of Community Vitalization Using SBT (Soulbound Token) and FT (Fungible Token) on Blockchain. (IEEE ICCE 2024) 他

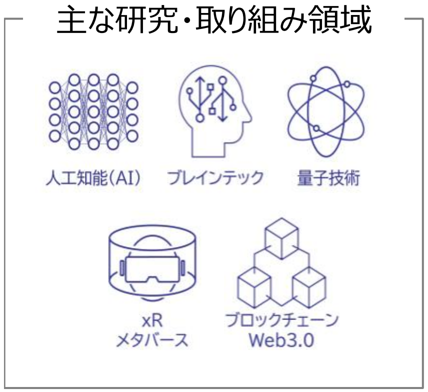
■ 専門委員活動 他

- ISO/TC 68委員会 委員、ISO/TC 68/JWG 1（デジタル通貨） エキスパート

先端技術ラボ

先端技術を活用したITサービスの創出に向けた技術の目利き役として、
「先端技術トレンドの調査・提言」、「技術検証・評価」、
「ビジネス活用の観点からの応用研究」に取り組んでいます。





弊社ホームページの **特集サイト** では、IT分野における先端技術の調査レポート、及び所属する部員のプロフィール詳細がご覧いただけますので、ぜひご参照ください。

本レポート執筆者へのメディア取材や講演などに関するご相談につきましては、当社ホームページの **問い合わせフォーム** よりご連絡ください。

株式会社日本総合研究所

日本総研は、シンクタンク・コンサルティング・ITソリューションの3つの機能を有するSMBCグループの総合情報サービス企業です。

東京本社　〒141-0022　東京都品川区東五反田2丁目18番1号　大崎フォレストビルディング
大阪本社　〒550-0001　大阪市西区土佐堀2丁目2番4号

© 2025 The Japan Research Institute, Ltd. (except where indicated)

日本総研

The Japan Research Institute, Limited