日本総研
The Japan Research Institute, Limited

生成AIを活用したシステム開発の現状と展望

- 生成AI時代を見据えたシステム開発に向けて -

株式会社日本総合研究所　先端技術ラボ

2024年09月30日

<本資料に関するお問い合わせ>
伊藤蓮(ito.ren@jri.co.jp) 近藤浩史(kondo.hirofumi@jri.co.jp)

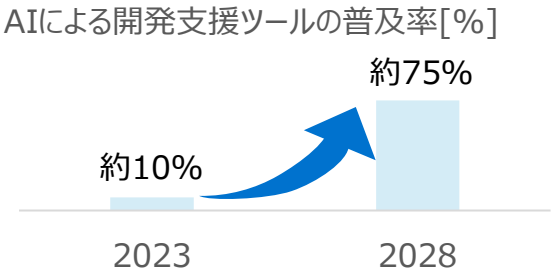
はじめに

- **システム開発における生成AI活用が注目**を浴び、**各企業での取り組みやアカデミアでの研究が活発化**している。
- 本資料では、システム開発における生成AIの活用について、**ユースケースや生成AIサービス**、**各企業での取り組み事例**、**アカデミアでの研究事例**を紹介した後、最新の状況を踏まえて**今後の展望**について記述する。なお、本資料では生成AIをシステムに組み込む開発（生成AIを搭載したチャットボットの開発など）は原則として対象外とする。

システム開発における生成AI活用の注目度

企業での活用見通し

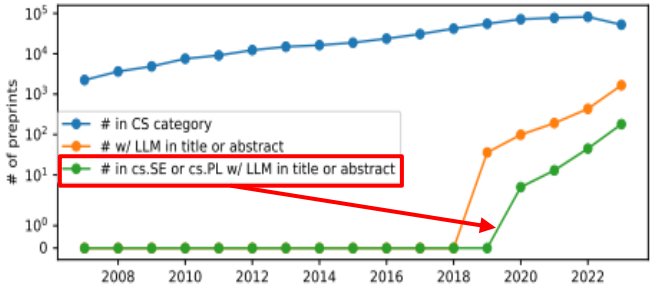
Gartner社によると、2028年には**75%**の企業のソフトウェアエンジニアがAIによる開発支援ツールを活用すると言われている。



[出所] Gartner Newsroom, Gartner Says 75% of Enterprise Software Engineers Will Use AI Code Assistants by 2028, 2024/4/11, https://www.gartner.com/en/newsroom/press-releases/2024-04-11-gartner-says-75-percent-of-enterprise-software-engineers-will-use-ai-code-assistants-by-2028 を基に日本総研作成

研究動向

生成AIをソフトウェア開発に活用する研究論文*が増化中。



[出所] Angela Fan et al., "Large Language Models for Software Engineering: Survey and Open Problems", arXiv:2310.03533v4, https://arxiv.org/pdf/2310.03533 のFig.2を基に日本総研作成

*arXivで出版されたソフトウェア工学の論文のうち、LLM等のキーワードを含んだ論文

生成AIがシステム開発にもたらす効果

開発効率の向上	• AIによるコードの自動生成やデバッグの支援などにより、開発作業が効率化される。 • システムの迅速なリリースにも繋がる。
品質の向上	• AIによるデバッグの支援やレビューの支援などにより、エラーが減少する。 • パフォーマンスの最適化など品質担保に必要な作業もAIが支援してくれる。
開発の負担軽減	• AIによる支援を受けることで、開発時間が短縮され、人経費削減などが期待できる。 • 開発従事者の労働時間が減少することで負担も軽減される。
サポートの強化	• 運用フェーズで生成AIを活用し、例えばユーザ向けのサポートを自動で行えるようにする等、ユーザ体験の向上につながる。

エクゼクティブサマリ

システム開発において生成AIはどのような場面で活用できるか？(1章)	• ChatGPTなどの汎用的な生成AIはシステム開発の全工程で役立つ。 • GitHub Copilotなどのツールは、エンジニアのコーディング作業の効率化に役立つ。 • テスト自動化、SQL生成、脆弱性対応などのエンジニア特有の業務やプロジェクト管理に特化して効率化を支援するツールも存在する。 • クラウドベンダが提供する生成AIサービスにより、開発プラットフォームが強化されつつある。
各企業でどのような取り組みが行われているか？(2章)	• ITベンダ・SIerでは、上流工程も含めた開発の全工程で効率化を試みる取り組みや、モダナイゼーションに利用する取り組み、開発ガイドラインや独自ツールの整備の取り組みなどがある。 • 信頼性の高いシステムが求められる金融業界でも、日本での事例は少ないが、システム開発に生成AIを活用し始めている。海外ではGitHub Copilotをエンジニア向けに取り入れて、開発の効率化を試みる取り組みが存在する。 • その他にも、生成AIを活用してコーディング業務や内製開発を効率化した企業がある。
研究事例としてどのようなものがあるか？(3章)	• コーディング支援やテスト自動化に関する研究が多い。 • システムの要件定義や保守運用に関する研究も出てきている。 • LLMエージェントを用いて自律的にシステム開発を行う研究も出てきており、今後に期待がかかる。
今後どのような変化がもたらされるか？(4章)	• 生成AIの進化により下記のような変化が生じる。 ➢ 短期的変化：人間とAIの協調による開発効率化 ➢ 中期的変化：AI導入が前提となるシステム開発手法の確立 ➢ 長期的変化：システム内製化のハードルが下がることによるIT業界のビジネス構造の変革

目次

システム開発における生成AI活用ユースケース

- システム開発における上流工程から下流工程、保守・運用に至るまで**幅広いユースケースが想定**される。
  設計・実装フェーズやテストフェーズを中心に、エンジニアの作業を支援するユースケースが多い。
- 入力プロンプトに従ってAIが成果物を生成し、人間が出力を確認したうえで対話や編集により成果物の品質を高める。

企画・要件定義

アイデア創出

AIとの対話によってアイデアを膨らませる。

要求・ニーズ分析

AIにユーザのペルソナを演じてもらい、要求やニーズを洗い出す。

システム要件作成

AIに機能・非機能要件の案を生成させる。対話を通じてブラシュアップしたり、作成した要件の漏れがないかレビューしてもらう。

設計・実装

ソースコード・SQL自動生成

AIに作成してほしいコードの概要を入力し、プログラムやSQL文を生成させる。対話を通じてブラシュアップする。

リファクタリング

既存のソースコードから機能をそのままにデザインパターンに則ったコードに修正したり、可読性の高いコードに修正する。

コード変換

COBOLからJavaなど、あるプログラミング言語で記述されたコードを別の言語のコードへ変換する。

コード説明

AIに既存の複雑なコードの内容を要約・説明してもらい、ドキュメント作成やリファクタリングに繋げる。

コードレビュー

AIにソースコードについて可読性や機能面、パフォーマンス、セキュリティ等の観点からアドバイスをもらう。

UIデザイン作成

AIにUIデザイン案を出してもらい、対話を通じてブラシュアップし、コードに変換してもらう。

テスト

テストケース生成

プロンプトにより、システムの機能テストや総合テストのテストケースを作成する。

テストコード生成

プロンプトにソースコードを入力し、それに応じた単体テストコードを作成する。

バグ検出・修正

AIに作成したコードのデバッグ時のエラーメッセージを基に修正案を出してもらう。

保守・運用

インフラ・リソース管理

AIにインフラやリソースの管理、デプロイを支援してもらう。

脆弱性検知・修正

使用ライブラリ管理など、システムに脆弱性がある場合に検知し、修正案を出してもらう。

運用効率化

システムで発生したエラーメッセージなどを解読し、パッチ適用による修正案を出してもらう。

情報調査：各工程で必要な情報を、対話やWeb検索により効率的に取得する。

ドキュメント作成支援：プロンプトにより、システム開発に用いる各種ドキュメントの雛形や下書きを作成し、対話を通じてブラシュアップする。

PJ管理支援：PJ計画案・タスク案の作成、タスク消化実績の要約によるPJ進捗管理、チケットの要約によるシステムの課題管理等をAIに支援してもらう。

システム開発ユースケースに対応した生成AIサービス一覧

- 前頁のユースケースに対応する様々な生成AIサービスがリリースされている。
- いくつかのユースケースをグループ化し、次頁以降で**代表的なサービスをピックアップして紹介**する。

企画・要件定義	設計・実装	テスト	保守・運用

対話型マルチモーダル生成AI (p.8)
ChatGPT, Gemini, Claude, Llama 3

- アイデア創出
- 要求・ニーズ分析
- システム要件作成

エンジニアアシスタント型生成AI (pp.9-10)
GitHub Copilot, GitLab Duo, Tabnine, Cursor

- ソースコード・SQL自動生成
- リファクタリング
- コード変換
- コード説明
- コードレビュー
- テストケース生成
- テストコード生成
- バグ検出・修正
- インフラ・リソース管理
- 脆弱性検知・修正
- 運用効率化

デザイン作成型生成AI (p.12) : Canva, Adobe Firefly, Visual Copilot, FigmaAI, v0, LlamaCoder

- UIデザイン作成

Web検索型生成AI (p.8): Perplexity AI, GenSpark, AI Overview, SearchGPT

- 情報調査

ドキュメント作成型生成AI : Copilot for Microsoft 365, Gemini for Google Workspace

- ドキュメント作成支援

プロジェクト管理支援型生成AI (p.11) : Atlassian Intelligence, AsanaAI, FigJamAI

- PJ管理支援

汎用的な生成AIサービス

●プロンプト次第で多くのユースケースに対応できる生成AIサービス。システム開発でも、**幅広いユースケースで利用**できる。

	サービス名 (企業)	特徴
対話型マルチモーダル生成AI	**ChatGPT** (OpenAI)	• サービスリリース後から、2か月でユーザ数が1億人を超えるなど、**生成AIブームの火付け役**である。 • 最新モデル「OpenAI o1」*1は、プログラミングコンテストで上位10%程の性能を見せ、**コーディング能力が向上**。
	Gemini (Google)	• 2023年に「Google Bard」という名称でリリースし、2024年2月にAIモデルの名称と統一され現名称へ変更。 • Google検索と連動してWeb上のコンテンツ情報を利用しており、**リアルタイム性の高い回答を生成**できる。
	Microsoft Copilot (Microsoft)	• 当初はBing検索にLLMを組み込んだサービスとして提供開始、以降に一般提供および現名称へ変更。 • Microsoft Office製品との統合やデータ保護機能を提供するなど、**企業での活用を念頭にした機能が強い**。
	Claude (Anthropic)	• アメリカのスタートアップ企業が開発。リリース後、**回答精度・コスト・プロンプト長の観点で人気が急上昇**した。 • 最新モデル「Claude3.5 Sonnet」が**GPT-4oやGeminiを上回る性能**を見せた(2024年9月時点)。
	Llama 3 (Meta)	• ChatGPTやClaudeの最新モデルの性能に引けをとらない、**オープンソースによるLLM**。 • 2024年9月にリリースされたモデル(Llama 3.2)により画像処理が可能になり、**マルチモーダル化**した。
Web検索型生成AI	**Perplexity AI** (Perplexity)	• Googleの元研究者らによって設立されたスタートアップが提供。**検索型の生成AIサービスの先駆者的存在**。 • 回答生成時にソースが出力されることが特徴であり、**回答の正確性をチェックすることも容易**である。
	GenSpark (MainFunc.ai)	• 2024年6月にリリースされた、Perplexity AIに似たサービス。2024年9月時点ではベータ版。 • AIエージェントにより複数の視点でWebページを検索、要約できる。また、生成したページをリンクで共有可能*2。
	AI Overview (Google)	• ユーザのGoogle検索でのクエリに従って、生成AIが**検索結果の概要説明などを出力**する点が特徴。 • 2023年から一部ユーザ向けに試験運用。2024年8月に日本でも正式提供を開始した。
	Search GPT (OpenAI)	• 2024年7月に発表された、生成AIを利用してWeb検索を行う機能。 • 2024年9月時点ではテスト段階であるが、将来的にChatGPTへ統合される見込み。

*1：2024年9月時点はPreview版として提供, *2：2024年9月時点では、生成したページはリンクを知っている人であれば誰でも見れてしまうため、情報保護の観点で注意が必要。

エンジニアアシスタント型生成AI（1/2）

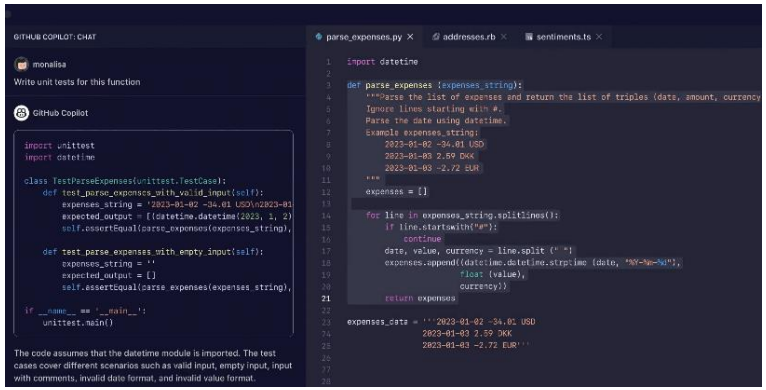
- プログラミングなどに利用する**エディタ/IDEと統合され、効率的にコーディングするための生成AIサービス**が増加。
- 作業効率向上には利用する**ツールのユーザビリティが重要**となる。また、一部ツールでは**開発ライフサイクル全体を効率化する機能**があり、**自社の効率化したい業務や周辺環境(プロジェクト管理ツールなど)に応じたサービス選定**が重要。

GitHub Copilot (GitHub社)

- **コーディングを補助する生成AIの代表的サービス**。Visual Studio Code*1等のエディタ・IDEと統合されている。
- GitHub上のコードを学習したAIモデルであり、様々な言語のコード生成に対応。**コミットメッセージ生成、プルリクエスト生成、脆弱性修正**など、開発ライフサイクル全体の効率化に寄与。
- ライセンス侵害防止のため、**一定の長さ以上公開コードと一致する場合にはコードの提案を抑える機能**なども提供している。
- 自然言語でコードに関する計画や、ビルド、テスト、デバッグができる「GitHub Copilot Workspace」も提供*2 。

*1：以降、「VSCode」と表記　*2：2024年4月からテクニカルプレビュー版として提供開始

利用例：コーディング中の関数(右)について、
単体テスト作成の指示を行い、結果を確認(左)



[出所] GitHub Copilot · Your AI pair programmer · GitHub, https://github.com/features/copilot (参照：2024/9/3)

Cursor (Anysphere社)

- VS Codeを複製して開発された**生成AI搭載のコードエディタ**。
- 直感的なUIで使いやすく、継続的に機能改良されている。
- コードやライブラリのドキュメントに関するQ&Aも可能。
- Composer機能により、複数ファイルの編集もでき、ユースケースが広がった*3。

*3 :2024年9月時点では、ベータ版として提供

利用例：API処理の並列化

自然言語による入力(下図赤枠)により、生成AIが既存コードの変換を自動で提案する。利用者は「Accept」ボタンにより提案を受け入れる。

[出所] Cursor - Built Software Faster, https://www.cursor.com/ (参照：2024/9/4)

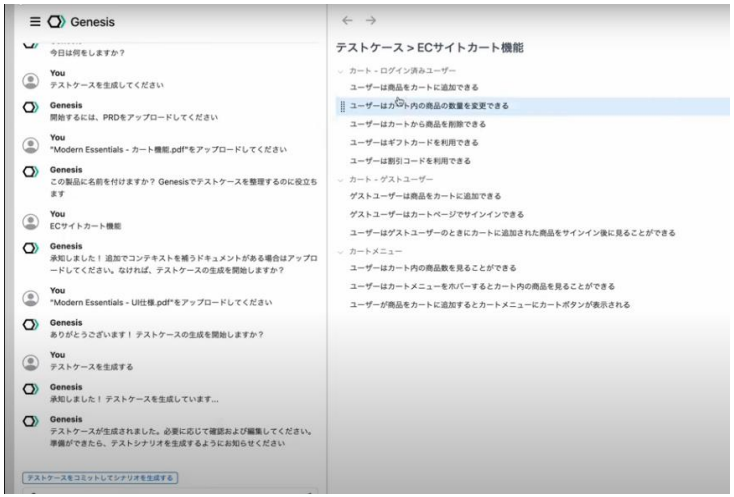
エンジニアアシスタント型生成AI（2/2）

●コーディング全般の作業を補助するのではなく、**開発プロセスにおける特定のユースケースの補助に特化したサービス**もある。例えば、テストケースの自動生成やSQL文の自動生成、コードの脆弱性診断・自動修正などがある。

Autify（オーティファイ社）

- AIによるノーコードテスト自動化ツールに生成AIを搭載した機能「Autify Genesis」をβ版として公開。
- プロダクトの仕様書をアップロードし、**対話形式でテストケース・テストシナリオを自動生成**。

利用例：仕様書についてLLMと対話し(左)、テストケースを生成(右)



[出所] Autify Genesis ｜生成AIによるテストケース・テストシナリオ自動生成 ｜Autify（オーティファイ）, https://autify.jp/products/genesis (参照：2024/9/3)

SQLAI.ai（SQLAI.ai社）

- SQLの取り扱いに特化し、**自然言語からのSQL文の自動生成や修正、説明、最適化**が可能。MySQLやPostgreSQL, SQL Serverなど20種以上のDBエンジンに対応。
- **自社のDBスキーマでAIを学習する機能も提供**している。

[出所] Generate SQL Queries in Seconds for Free - SQLAI.ai, https://www.sqlai.ai/ (参照：2024/9/3)

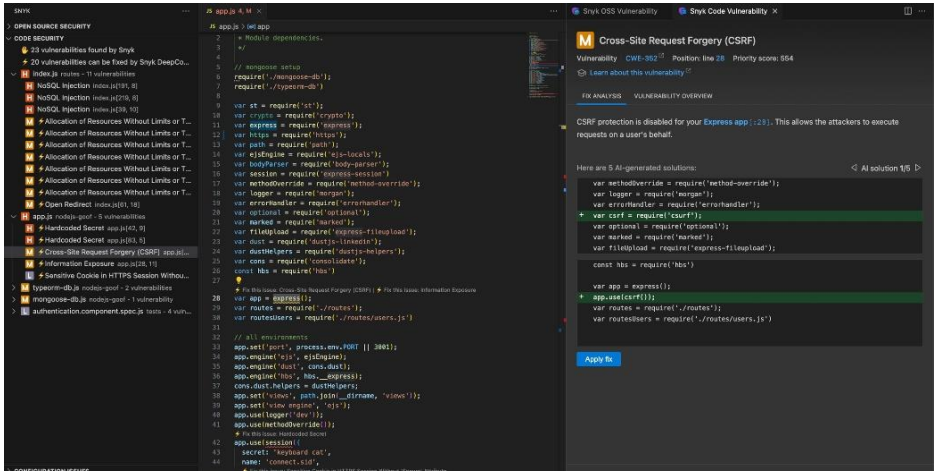
Snyk DeepCodeAI（Snyk社）

- 「Snyk Code」サービスで提供されている、**コードの脆弱性診断と修正案を自動で提示**するサービス。
- AIコーディングツールと組み合わせるユースケースを想定し、生成AIにより出力されたコードの診断も可能。
- 生成AIと複数の機械学習手法の両方を活用して、**ハルシネーションを抑えた高精度なスキャンと修正案生成**が可能。

DeepCodeAIによるCSRF*検知・修正の例

実装したコード(左)からCSRFの脆弱性を検知し、
コード修正案(右)を提示

*クロスサイト・リクエスト・フォージェリ



[出所] Fix code vulnerabilities automatically | Snyk User Docs, https://docs.snyk.io/scan-using-snyk/snyk-code/manage-code-vulnerabilities/fix-code-vulnerabilities-automatically#enable-deepcode-ai-fix (参照：2024/9/3)

プロジェクト管理支援型生成AI

- システム開発の**プロジェクト管理向け**に、生成AIを組み込んだツールが提供され始めている。
- 自社のプロジェクト管理方法に照らし合わせてツールを導入することで、プロジェクト管理を効率化できる。

Atlassian Intelligence (Atlassian社)

- プロジェクト管理向けの同社製品内で使える生成AIサービス。
- チケット管理ツールであるJiraでは、**自然言語でのチケット検索やフォーマットに応じたチケット生成、ユーザーストーリー生成、タスク分解、個人のスキルに応じたタスク割り当て***等が可能。
- サービス運用ツールであるJira Service Managementでは、**課題の説明やコメント生成、課題の要約、製品サポートの自動化、アラートのグループ化、顧客感情分析***などができる。
- ナレッジ共有ツールであるConfluenceでは、**ページの要約やQ&A検索、固有の用語も含めた用語説明***などができる。

Asana AI (Asana社)

- プロジェクト管理ツールであるAsanaの生成AIサービス。
- **プロジェクトやタスクの状況の要約や、Q&A、目標・KPIの提案やダッシュボード作成、オンボーディング支援***などができる。

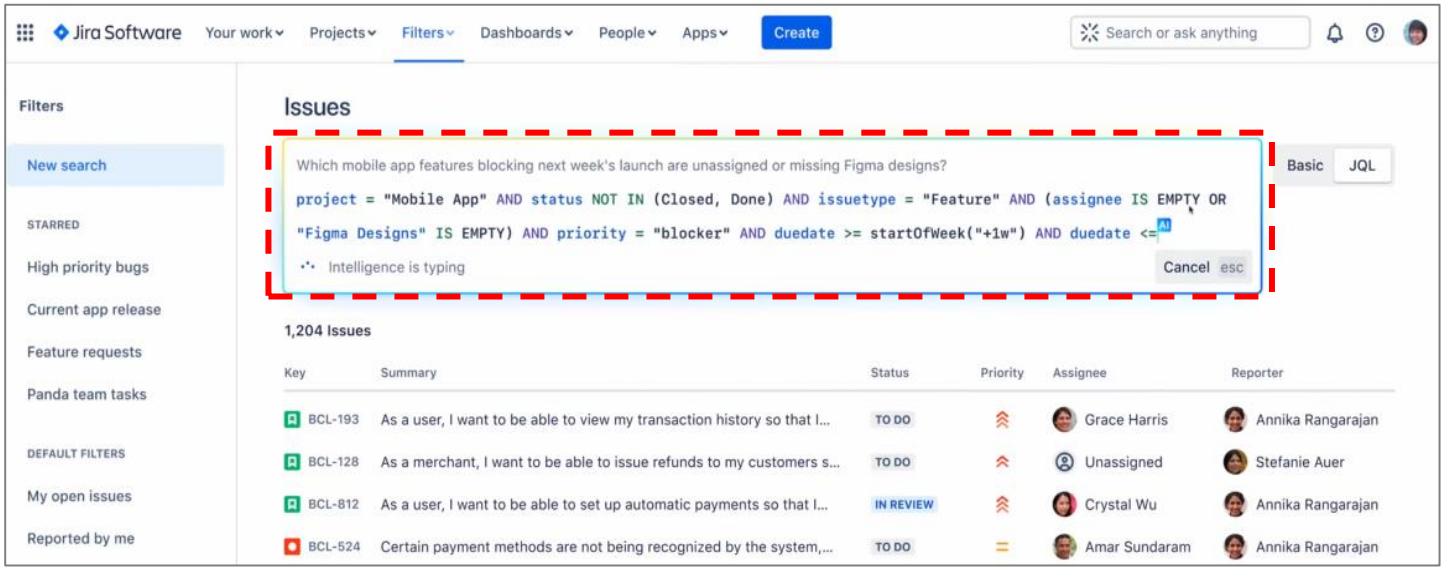
FigJamAI (Figma社)

- チームでのコラボレーションツールであるFigJamに搭載された生成AIサービス。2024年9月時点はベータ版として提供中。
- **ガントチャート・タイムラインなどの計画作成、会議のホワイトボードの内容要約**などができる。

*近日実装予定の機能も含む

Atlassian Intelligenceの例

自然言語でIssuesを検索する例。
自然言語から、Issue検索用の独自言語(JQL)に生成AIが変換する（右図赤枠）。



[出所] Atlassian Intelligence | AI in Atlassian Cloud products, https://www.atlassian.com/platform/artificial-intelligence (参照:2024/9/3)

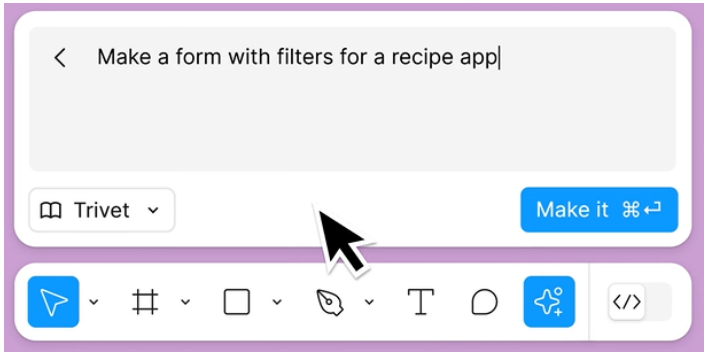
デザイン作成型生成AI

- 画像生成ツールは非常に多くあるが、特にWebアプリ開発で重要な、**UIデザインに生成AIを活用する動き**もある。
- 単なるUIデザインのみならず、v0のように、**プロンプトから簡単なWebアプリを生成できるサービス**も出てきている。

サービス	特徴
Visual Copilot (Builder.io社)	• 生成AI搭載の、Webデザイナー向け支援ツール。 • Webデザインツール**「Figma」のデザインをHTML、React, Vue, Angularなど10種類以上のコードに変換**できる。
Figma AI (Figma社)	• **プロンプトからのUIデザイン生成や、文章のトーンや長さを変換する機能、静的なモックから動的なプロトタイプを作成する機能**などを提供。 • 検索クエリの意味やコンテキストに基づき必要なコンポーネントやアセットを検索して出力する、既存画像の検索によるデザイン再利用など**生成AIによる検索機能**も提供。
v0 (Vercel社)	• **プロンプトから簡単なWebアプリを生成**できるSaaSツール。 • 生成したWebページのコードをダウンロードできる。Reactベースのコードが中心だが、今後UIライブラリを拡充予定。 • **過去のバージョンに戻る機能**や、**リンクを基に生成物を簡単に共有できる機能**などがある*。
LlamaCoder (Together AI社)	• Meta社の「Llama 3.1」を利用した独自の推論技術によりプロンプトから**簡単なWebアプリを生成しデプロイまでできるOSSツール**。 • 生成された**コードに人間が修正を加え、リアルタイムにテストしながら開発**できる。

*今後、オプトアウト機能を提供予定としているが、2024年9月時点では、生成物はAIの学習に用いられる点など、セキュリティ面で注意が必要である。

Figma AIのMake Design機能の例



「Make a form with filters for a recipe app」とプロンプトを入力すると、当該機能を実現するデザイン案が生成される。

[出所] Figma AI: 創造力をFigma AIで解き放つ, https://www.figma.com/ja-jp/ai/ (参照:2024/9/3)

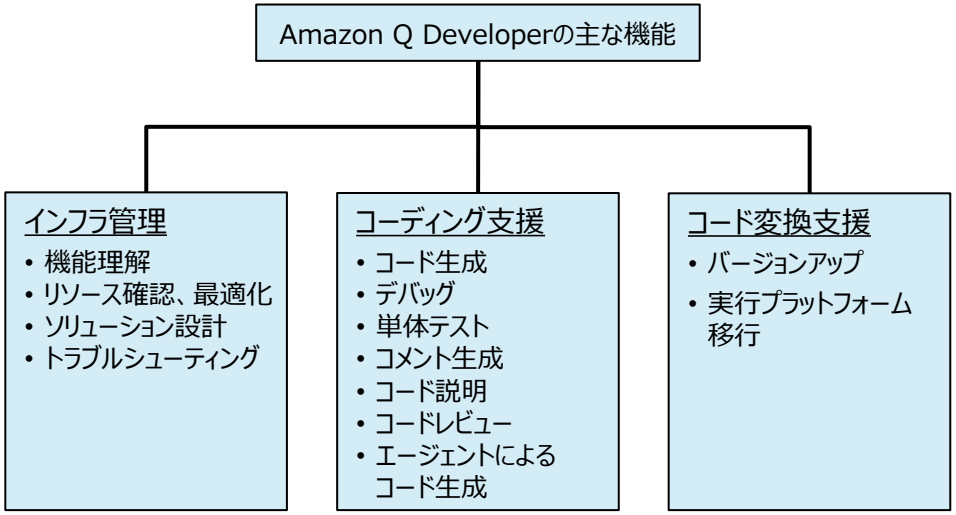
クラウドベンダによる生成AIサービス

- クラウドベンダ各社は生成AIを取り入れて、**クラウド上でのシステム開発を効率化**するためのツール群を発表。
- コーディング支援だけでなく、チャットによるサービス説明やリソース確認、 IaC*1自動生成や脆弱性診断などのインフラ管理支援機能を搭載。こうした機能により、**システム開発にクラウドを活用する流れが加速していく可能性**がある。

*1 IaC (Infrastructure as Code)：サーバなどのインフラ環境をコードにより自動構築する仕組み

Amazon Q Developer (AWS)

- チャットによる**AWSの機能理解やリソース確認・最適化、ソリューション設計、トラブルシューティング**などができる。
- VSCodeなどのIDEと統合し、JavaやPython, C++など12種以上の言語に対応したコード生成・補完ができる。**TerraformなどによるIaC支援**も可能。
- コード変換の支援機能も提供しており、**Javaプロジェクトのバージョンアップ支援***2ができる。**WindowsからLinuxコードへの変換などにも対応**予定。



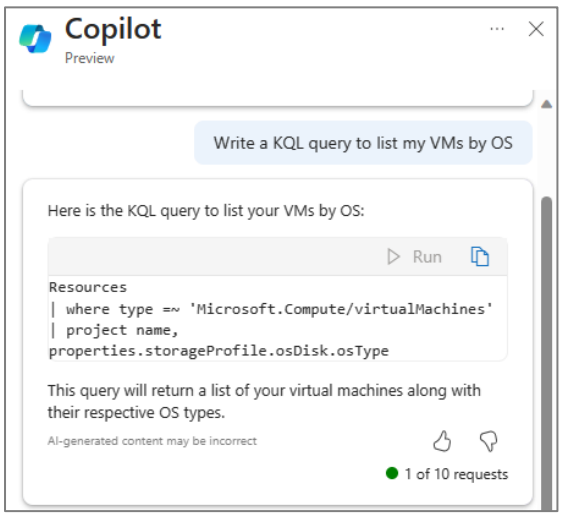
*2 2024年9月時点では2GB以下のサイズ制限あり

Gemini Code Assist (Google)

- Geminiを用いて構築された、ソフトウェア開発者向け支援機能。
- VSCodeやJetBrainsなどのIDEと統合し、JavaやPython, C++など20種以上の言語に対応したコード生成ができる。
- コード全体に対し、機能追加やファイル間依存関係解析、ファイル間を跨ぐ変更、バージョンアップ支援などが可能。*3
- 組織の**既存コードやナレッジを基にコードをカスタマイズ可能**。*3

Microsoft Copilot in Azure (Microsoft)

- チャットを用いて**Azureサービスの機能理解やリソース最適化、トラブルシューティングなどができる機能**が公開されている。*3
- 右図は「OSごとにVMを一覧表示するKQLクエリを作成」する例。



[出所] Microsoft Copilot in Azure (プレビュー) を使用してリソース情報を取得する | Microsoft Learn, https://learn.microsoft.com/ja-jp/azure/copilot/get-information-resource-graph (参照:2024/9/3)

*3 2024年9月時点でプレビューの機能

主要ITベンダ・SIerにおける各社動向（1/2）

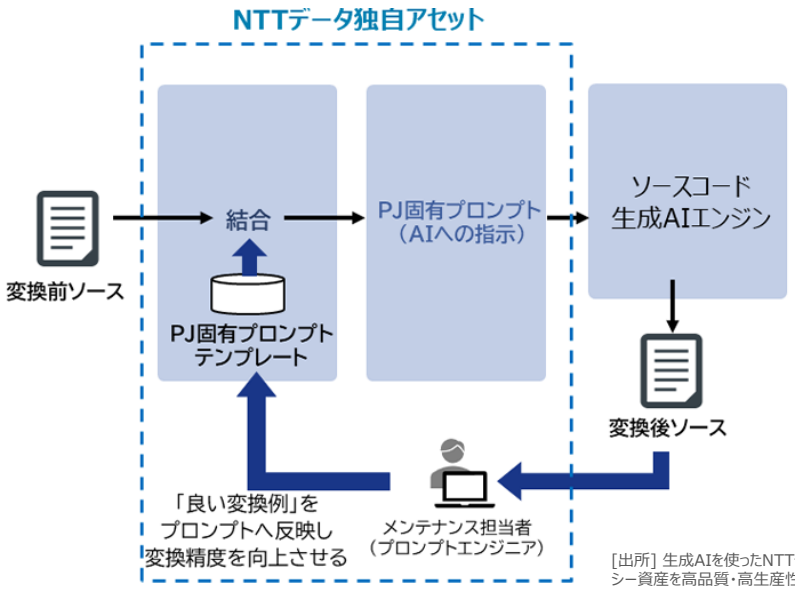
- 精力的に生成AI活用を進めているのはITベンダ・SIerで、**システム開発の幅広い工程で模索**されている。
- 社内外のシステム開発で生成AIによる**開発効率化の事例**や、**開発ガイドラインや独自ツールの整備**などの動きがある。
- SI事業の性質上、実装工程のみならず、**要件定義などの上流工程やモダナイゼーション**における生成AI活用を模索。

IBM

- 製品として「watsonx Code Assistant」を提供。**コード生成**や、COBOLからJavaへの変換による**モダナイゼーション**、Ansible Playbookの生成による**インフラ管理自動化**を実現。

NTT Data

- **ソフトウェア開発の全工程で積極的に生成AIを活用**する方針。
- COBOLからJavaへの変換やデータベースの移行作業など、**マイグレーション作業**でも生成AIによって効率化できると見込んでいる。



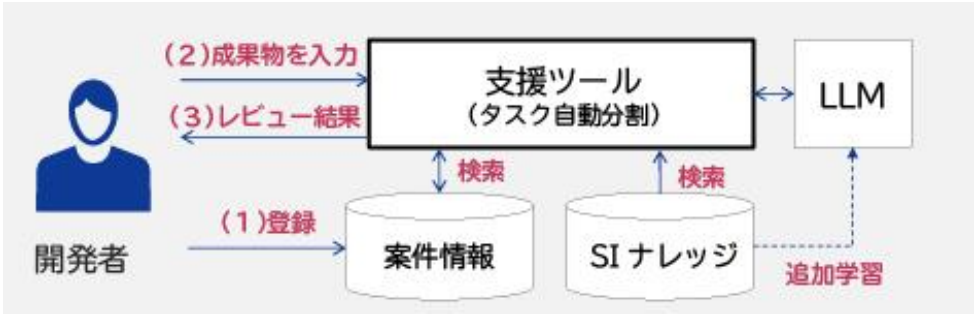
[出所] 生成AIを使ったNTTデータ流「新時代のシステム開発」とは ～グローバルで商用への適用実績拡大中！レガシー資産を高品質・高生産性でモダナイズ～ | DATA INSIGHT | NTTデータ - NTT DATA, 2023/11/16, https://www.nttdata.com/jp/ja/trends/data-insight/2023/1116/ (参照:2024/9/4)

NEC

- **コーディングやテスト工程を中心に活用**。GitHub Copilotを NECグループ向けクラウド型ソフトウェア開発基盤上で活用。

[出所] 2023年度JEITA ソフトウェアエンジニアリング技術ワークショップ 講演資料, NECの生成AIとソフトウェア・システム開発への取り組みについて, 2024/2/9, https://home.jeita.or.jp/system/seminar/pdf/seminar_yanoo.pdf (参照：2024/9/4)

- 要件定義や設計工程など上流工程にも活用。ただし、設計工程では案件固有の知識が必要となることが多く、まだ発展途上と見ており、**RAGや独自LLM開発による検証**を進めている。
- 現行の開発ガイドラインを、生成AIを活用したものに拡充するなど、**開発プロセスやモダナイズプロセスの改革**にも取り組む。



[出所]矢野尾 一男, ソフトウェア・システム開発への生成AIの活用, NEC技法 Vol75 No2, 2024年3月, https://jpn.nec.com/techrep/journal/g23/n02/pdf/230209.pdf (参照：2024/9/4)

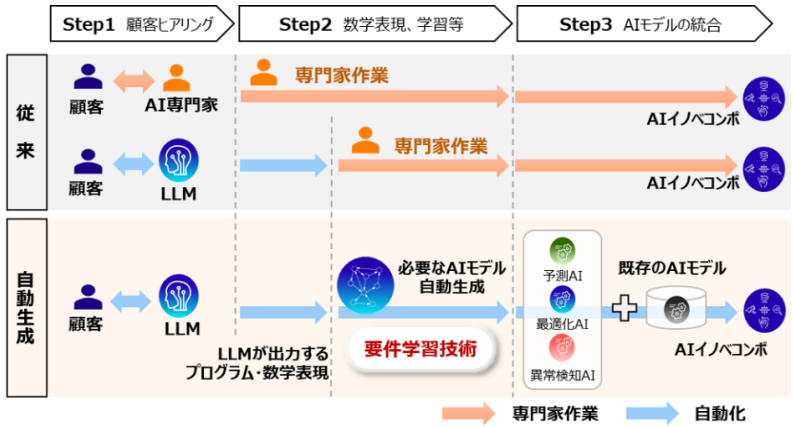
主要ITベンダ・SIerにおける各社動向（2/2）

富士通

- ソフトウェア開発の各工程で様々なユースケースを模索中だが、**「コード生成よりもインパクトがあるのは上流の品質向上」**としている。そのため、設計の抜け漏れチェックなど、設計書レビューでの生成AI活用検証をみずほFGと合同で進めている(P.14)。

[出所] 2023年度JEITA ソフトウェアエンジニアリング技術ワークショップ 講演資料, 富士通の生成AI技術: ソフト開発特化型生成AIと富岳LLM, 2024/2/9, https://home.jeita.or.jp/system/seminar/pdf/seminar_kobayashi.pdf (参照：2024/9/4)

- 「Fujitsu Kozuchi」のプラットフォーム上で、顧客の自然言語による入力から、**要件を満たすAIコンポーネントを生成AIで生成**する技術を開発。主に最適化や予測、異常検知などが対象。



[出所] 富士通プレスリリース, AIイノベーションコンポーネントを自動生成するAI技術を開発, 2023/10/11, https://pr.fujitsu.com/jp/news/2023/10/11.html (参照:2024/9/4)

日立製作所

- ミッションクリティカルなシステムのコードにも生成AIを活用する。システム開発に生成AIを適用するフレームワークとして、**ガイドライン**やVSCodeの**プラグインツールを整備**し、社内やSI事業で活用。
- また、JCB社とともに**生成AIを活用した開発フレームワーク標準化**に向けた検証プロジェクトを実施すると発表。

[出所] 日立ニュースリリース, 生成AIを活用し、システム開発のトランスフォーメーションを加速, 2024/5/21, https://www.hitachi.co.jp/New/cnews/month/2024/05/0521.html (参照:2024/9/4)

- システム運用管理製品である「JP1 Cloud Service」で生成AIアシスタントを搭載した機能を2024年4月から正式リリース。**問い合わせ対応や障害対応、運用設計・自動化支援、運用レポート作成支援**などのユースケースを順次拡大予定。

[出所] 日立ウェブサイト, 生成AIで変わるこれからのIT運用とは？ ～生成AIがIT運用の変革をサポート IT運用の効率化・対応品質の向上～, https://www.hitachi.co.jp/Prod/comp/soft1/jp1/feature/generative_ai/index.html (参照:2024/9/4)

金融業界における各社動向

- 信頼性の高いシステムが求められる金融業界でも、**事例は少ないが、システム開発に生成AIを活用し始めている**。
- 国内では、みずほFGが大手ITベンダと共同でシステム開発や運用に生成AIを活用した事例がある。
- 海外では、GitHub Copilotをエンジニア向けに積極的に取り入れ、システム開発を効率化する動きが見られる。

国内の金融業界における事例

みずほFG・富士通によるシステム開発支援

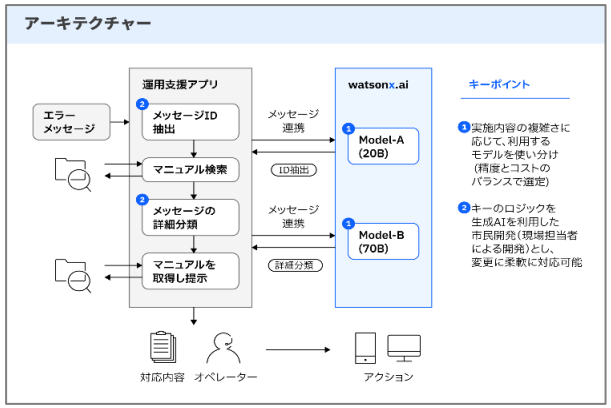
- 勘定系システム「MINORI」を対象した実証実験を実施。
- 既存の設計書とレビュー記録を生成AIのプロンプトに入力し、**設計書の記載漏れや誤りを自動検出することでシステムの品質向上**をめざした。有識者のナレッジ活用などにより**レビュー結果の正確性や網羅性向上**を確認した。

[出所] みずほフィナンシャルグループ ニュースリリース, 〈みずほ〉と富士通、システム開発・保守に生成AIを活用する共同実証実験を開始, 2023/6/19, https://www.mizuho-fg.co.jp/release/20230619release_jp.html (参照:2024/9/4)

みずほFG・IBMによるシステム運用支援

- システム運用支援アプリとIBM社のwatsonxの基盤モデルを連携し、**AIによるエラーメッセージの解釈から対応案を提示する**実証実験を実施。**98%の精度が実現**でき、**2024年度に本番環境への適用**を公表。

[出所] みずほフィナンシャルグループ ニュースリリース, 〈みずほ〉と日本IBM、システム運用に生成AIを活用する実証実験を通じて運用の高度化を実現, 2024/2/1, https://www.mizuho-fg.co.jp/release/20240201release_jp.html (参照:2024/9/4)



[出所] 日本経済新聞, みずほFGがシステム運用監視業務でIBMの生成AIを選択した理由, https://ps.nikkei.com/ibmportal/mizuhofg2405/ （参照：2024/9/4）

海外の金融業界における事例

CitiにおけるGitHub Copilot活用

- 2024年4月半ばまでに**約4万人の全開発者にGitHub Copilotを提供予定**と発表。
- **自社内のコードレポジトリを利用したRAGも実践**し、自社基準を満たすコード生成に利用。また、**レガシーシステムのモダナイゼーション**にも取り組む。

[出所] American Banker, Why Citi is rolling out generative AI to all its developers, 2024/2/27, https://www.americanbanker.com/news/why-citi-is-rolling-out-generative-ai-to-all-its-developers (参照:2024/9/4)

ANZ BankによるGitHub Copilotの効果検証

- 同社エンジニアに対して、GitHub Copilotを利用したグループと、そうでないグループとで効果を比較検証した論文を発表。
- **あらゆる熟練度のエンジニア**で、**GitHub Copilotを利用した場合に開発の生産性向上が確認**できたと報告されている。

Table 3: Productivity increase across different proficiency levels

Python proficiency	Mean Total_Time_Spent (Control Group)/per problem (in minutes)	Mean Total_Time_Spent(Copilot Group)/per problem (in minutes)	Productivity Improvement
Beginner	20.07	9.58	52.27%
Intermediate	28.60	16.70	41.6%
Advanced	39.82	23.70	40.48%

[出所] Sayan Chatterjee et al, The Impact of AI Tool on Engineering at ANZ Bank An Empirical Study on GitHub Copilot within Corporate Environment, arXiv:2402.05636, https://arxiv.org/abs/2402.05636

その他企業における動向

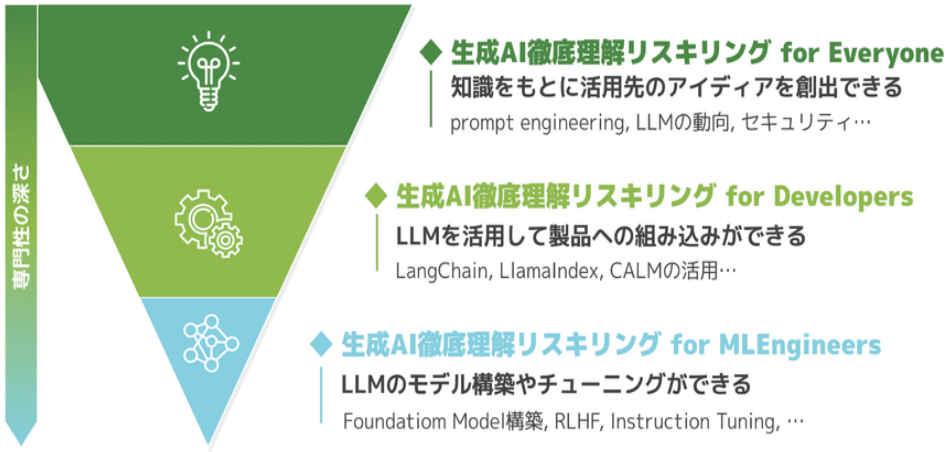
● 自社のソフトウェア開発に生成AIを取り入れ、**コーディングにおける生産性の向上**を発表している企業が複数存在。

サイバーエージェント

- 2023年4月からGitHub Copilotを全社的に導入。2024年4月では、**1000人以上の開発者のうち約8割が業務に活用**。
- GitHub Copilotの利用状況把握にも取り組み、コード送信行数や実装コード数で国内トップの実績とされる。
- 社内アンケートによると、**開発者の約半数が1-2割程度のコーディング業務削減効果**があったとのこと。

[出所] CyberAgentWay, 対象者は1,000名以上、サイバーエージェントが日本一GitHub Copilotを活用している理由, 2024/3/7, https://www.cyberagent.co.jp/way/list/detail/id=29887 (参照:2024/9/4)

- 生成AIのリスキリングプログラムにも取り組んでいる。全社員対象のものだけでなく、LLMをプロダクトに組み込むための開発者向けコンテンツ整備など、先進的な取り組みがある。



[出所] CyberAgentWay, サイバーエージェントの99.6%にあたる社員・全役員が受講した「生成AI徹底理解リスキリング」とは？, 2024/2/1、https://www.cyberagent.co.jp/way/list/detail/id=29775 (参照:2024/9/4)

LINEヤフー

- 2023年10月から**約7000人の全エンジニアを対象にGitHub Copilotを導入**。一部エンジニア向けのテスト導入では、**1人あたり1日約1～2時間のコーディング時間削減と10～30%の生産性向上**が明らかになった。
- GitHub Copilotで生成したコードの信頼性を担保するため、**複数レビューの徹底などのルールを設定**している。

[出所] LINEヤフー ニュース, LINEヤフーの全エンジニア約7,000名を対象にAIペアプログラマー「GitHub Copilot for Business」の導入を開始, 2023/10/13, https://www.lycorp.co.jp/ja/news/release/000862/ (参照:2024/9/4)

内製開発を効率化した企業の例

企業名	内容
住友ゴム工業[*1]	Google CloudとGemini Code Assistを利用して**シミュレーションツールの内製開発の生産性向上**を実現。
TOPPANホールディングス[*2]	プログラム開発業務に特化したOSS-LLMの生成AIを導入。生成AI導入前と比較し、プログラム開発に費やす時間を**最大70%短縮**できたと発表。
東京海上日動システムズ[*3]	生成AIにより詳細設計書からコードを生成する実証実験を実施。既存アプリケーション修正や新規アプリ開発で**平均約40%、最大約90%の生産性向上**を発表。

[出所]
*1: Google Cloud公式ブログ, 住友ゴム工業様：Gemini を活用したクラウドベースの内製開発で、アプリ開発の生産性を大幅に改善, 2024/4/23, https://cloud.google.com/blog/ja/products/application-development/Sumitomo-rubber-industries-cloud-based-in-house-development-using-gemini (参照:2024/9/4)

*2: TOPPANホールディングス, TOPPANホールディングス、生成AIを活用し、社内システムプログラム開発の業務効率が約70%向上, 2023/11/9, https://www.holdings.toppan.com/ja/news/2023/11/newsrelease231109_1.html (参照:2024/9/4)

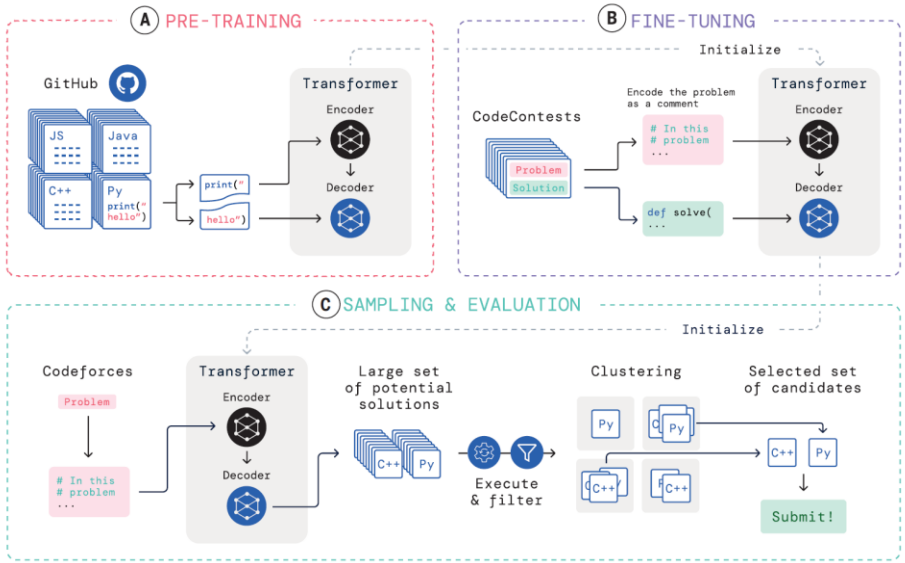
*3: IBM Newsroom, 東京海上日動システムズ、日本IBMと共同開発した生成AIを活用したコード生成ツールの実証実験で、プログラミング工程の生産性を約40%向上, 2024/6/26, https://jp.newsroom.ibm.com/2024-06-26-Tokio-Marine-and-Nichido-Systems-IBM-Japan-a-code-generation-tool-using-generative-AI (参照:2024/9/4)

コード自動生成の研究動向（1/2）

- システム開発に生成AIを活用する研究として、**先行する分野はコード自動生成**である。プログラミングコンテストで人間と同等レベルの性能を達成したAlphaCodeをはじめとして、多くの研究が存在する。
- コード生成では、LLMをコード生成タスク向けにFine-tuningして用いる研究が主流であった。一方、コード生成に特化したOSSベースのLLMが公開されてきたため、**プロンプトエンジニアリングやRAGによるコード生成の研究が増加**。

Fine-tuningを用いたコード生成例：AlphaCode

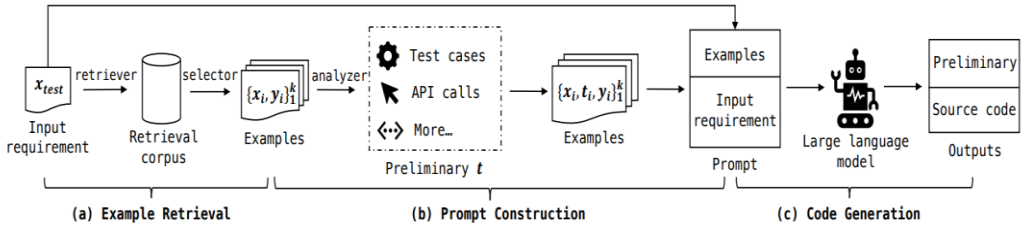
- 2022年にGoogle DeepMindが発表し、プログラミングコンテスト参加者の**上位50%以上**に匹敵する性能を見せた。人間と同等レベルでコードを生成できる初のAIとされ、注目を浴びた。
- GitHubのコードからPre-trainingしたモデルをプログラミングコンテストの問題と解答でFine-tuningして実現されている。
- 後継のAlphaCode2では**上位15%以上**に匹敵する性能を示した[*1]。



[出所] Yujia Li et al., “Competition-level code generation with AlphaCode”, Science378,1092-1097(2022), DOI:10.1126/science.abq1158 のFig.2より抜粋

プロンプトとRAGによるコード生成例：AceCoder

- 入力した要件(①)から、APIや単体テストケースなどの中間生成物(②)を出力させ、検索により類似コード(③)を取得し、①～③の3点をまとめたプロンプトをLLMに入力して、コードを生成する手法。



[出所] Jia Li et al., “AceCoder: Utilizing Existing Code to Enhance Code Generation”, arXiv:2303.17780, https://arxiv.org/pdf/2303.17780 のFigure 3

- 従来のプロンプト手法(Few-Shot)と比較して、**1回のコード生成でテストケースをパスする成功率(Pass@1)が向上**している。

モデル	手法	プログラミング課題 (数値:Pass@1)		
		Python	Java	JavaScript
CodeGeeX -13B	Few-Shot	20.40	16.63	11.16
	AceCoder	↑**26.74**	↑**28.38**	↑**21.03**
CodeGen -6B	Few-Shot	14.60	18.25	9.94
	AceCoder	↑**22.83**	↑**22.45**	↑**16.45**

[出所] Jia Li et al., “AceCoder: Utilizing Existing Code to Enhance Code Generation”, arXiv:2303.17780, https://arxiv.org/pdf/2303.17780 のTable1を基に日本総研作成

*1 “AlphaCode 2 Technical Report”, AlphaCode Team, Google DeepMind, https://storage.googleapis.com/deepmind-media/AlphaCode2/AlphaCode2_Tech_Report.pdf (参照：2024/9/4)

コード自動生成の研究動向（2/2）

● 通常のコード生成だけでなく、**コードのパフォーマンス最適化**や**セキュリティ面に注力した研究**も出てきている。

LLMによるコードのパフォーマンス最適化の研究例

- LLMを使って高速なC++のコードを生成するための研究。
- プログラミングコンテストで、プログラマがパフォーマンス改善のために行った編集を抜粋し、シミュレータによる実行時間とコードをセットにした「PIEデータセット」を構築。
- 複数の手法でコード生成を行ったところ、**PIEデータセットを利用した場合のモデルにおいて性能向上**を確認できた。
- **パフォーマンス最適化のためのデータを集めて利用**できれば、**LLMによるパフォーマンス最適化**を実現できると示唆された。

シナリオ	モデル	最適化率	速度改善率*2	正確率
人間による参考値	-	100%	3.66	100%
プロンプトのみ (CoT*1)	GPT-3.5	43.05%	1.30	78.73%
PIEデータセット利用 (検索・Few-Shot)	GPT-4	**76.07%**	**3.93**	**95.71%**
PIEデータセット利用 (Fine-tuning)	GPT-3.5	**87.63%**	**6.86**	**95.09%**

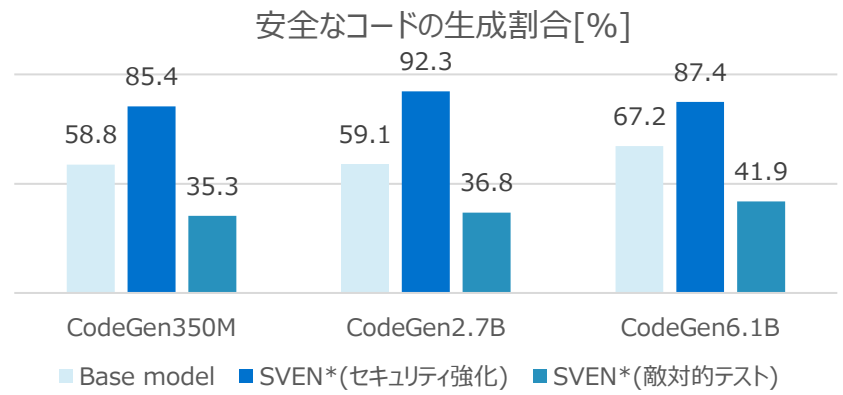
[出所] Alexander Shypula et al., “Learning Performance-Improving Code Edits”, arXiv:2302.07867, https://arxiv.org/pdf/2302.07867 のTable 1を基に日本総研作成

*1：Chain-of-Thought. LLMに中間的な推論ステップを加えさせることで推論能力を向上させるプロンプト手法。

*2：改善前の実行時間が改善後の何倍かを表す指標。

セキュリティ面で安全なコード生成に関する研究例

- LLMによるコード生成のセキュリティ強化と、敵対的テストによるリスク評価の両方を行った研究。
- LLMの重みを変更せず、セキュアなコード生成か否かを表すベクトルを追加し、このベクトル部分のみ学習させることで、少量のデータで効果的に学習できるようにした。
- 既存の脆弱性に関するデータセットを手作業で修正した高品質なデータセットを作成し、上述の手法でコード生成用のLLMを学習させたモデルで生成したコードを評価。
- 評価の結果、提案手法では、コード生成の性能を落とすことなく**安全なコードを生成する割合が大幅に増加**した(逆に、敵対的テストでは安全なコードの生成割合が減少することを確認)。



*SVEN:本研究で提案されている手法の名称

[出所] Jingxuan He et al., “Large Language Models for Code: Security Hardening and Adversarial Testing”, arXiv:2302.05319, https://doi.org/10.48550/arXiv.2302.05319 のFigure 7を基に日本総研作成

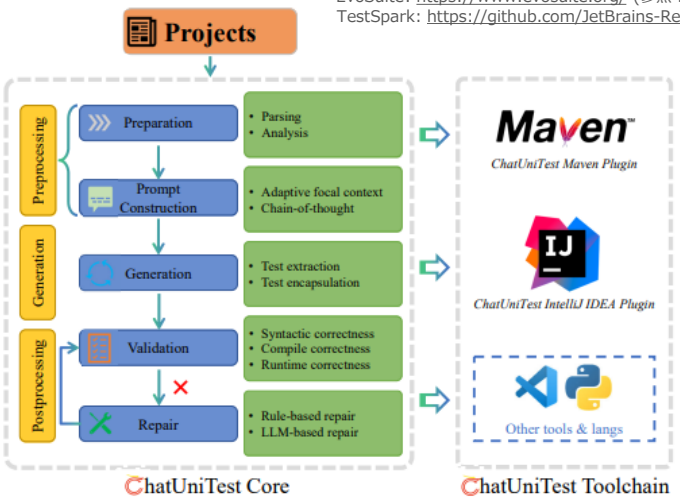
テスト自動生成の研究動向

- コード自動生成の次に研究が盛んな分野として、テスト自動生成技術が挙げられる。
- **既存コードから単体テストを自動生成**する研究が主流だが、**GUIテストを自動で行うツールの研究**も出てきている。

単体テスト自動生成の研究例：ChatUniTest

- LLMを利用した、Javaプログラムの単体テスト自動生成ツール。
- 前処理段階では、コード情報解析や依存関係解析を行い、重要度の高い情報を選定しプロンプトに追加する。
- テスト生成段階では、前段階で生成したプロンプトとユーザが入力したプロンプトに基づき、単体テストコードを自動生成する。
- 後処理段階では、生成したテストの構文解析・コンパイル・実行検証を行う。エラー発生時にはルールベースによる修正とLLMによる修正を加えることで、高品質なテストコードを生成。
- いくつかのプロジェクトで評価した結果、**既存のツール*を上回る性能**を見せた一方で、ラインカバレッジは平均で**59.6%**であり、まだ実用面には課題が見える。

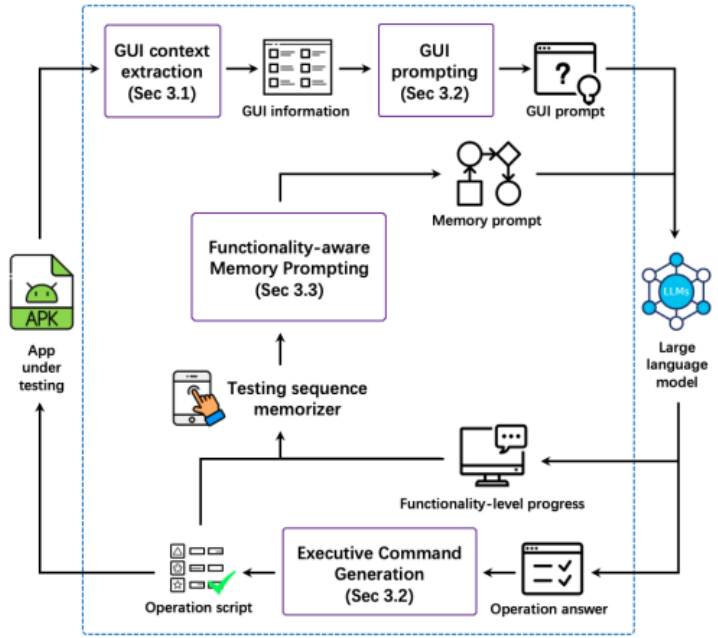
* EvoSuite: https://www.evosuite.org/ (参照：2024/9/4)
TestSpark: https://github.com/JetBrains-Research/TestSpark (参照：2024/9/4)



[出所] Yinghao Chen et.al, "ChatUniTest: A Framework for LLM-Based Test Generation.", arXiv:2305.04764, https://doi.org/10.48550/arXiv.2305.04764 の Figure 1より抜粋

GUIテストツールの研究例：GPTDroid

- LLMをテスターとして利用し、Q&A形式の対話によりスマホアプリのGUIテストを行うツール。
- ウィジェットなどのGUI情報から、Few-shot Learningのプロンプトによりアプリを GUI操作するコマンドを生成、テスト実行。
- Google Playの93個のスマホアプリに対して評価を行い、テスト自動化に用いられる複数のツールと比較したところ、既存ツールよりも短時間で**アクティビティカバレッジにおいて32%、バグ検出率において31%高い値**を示した。



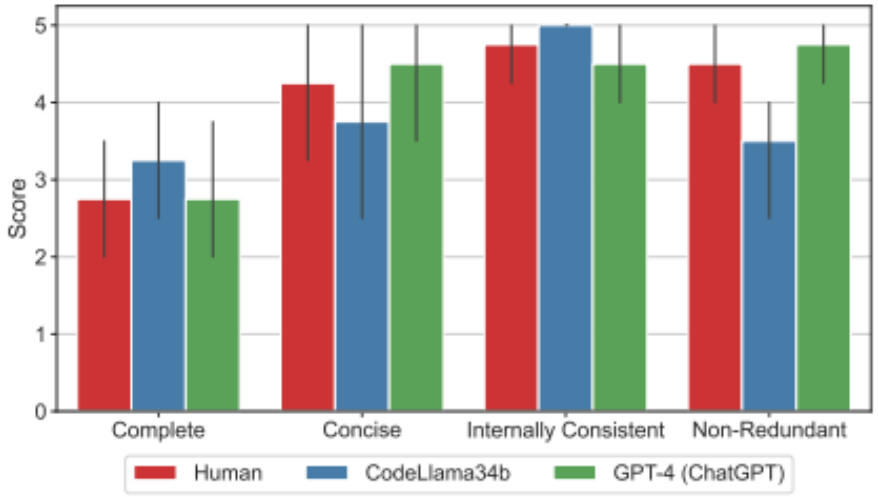
[出所] Zhe Liu et.al, "Make LLM a Testing Expert: Bringing Human-like Interaction to Mobile GUI Testing via Functionality-aware Decisions.", In Proceedings of the IEEE/ACM 46th International Conference on Software Engineering (ICSE '24). Association for Computing Machinery, New York, NY, USA, Article 100, 1–13. https://doi.org/10.1145/3597503.3639180 のFig.2より抜粋

要件定義フェーズにおけるLLM活用の研究動向

●コードやテストの自動生成と比較してまだ多くはないが、LLMによって**要件定義を効率化する研究**が出てきている。

LLMを活用した要件定義の研究例

- 大学のクラブ管理システムなど、**実用的なシステムにおけるLLMを利用した要件定義**に関する実証評価の研究。
- 入門レベルのエンジニア、CodeLlama、GPT-4のそれぞれで要件定義書を作成し、完全性や簡潔さなどの観点で人間のレビュアーによる5段階評価を行った。その結果、LLMによる要件定義書は**入門レベルのエンジニアとほぼ同等レベル**であった。
- また、LLMによる既存要件の検証と修正を正しく行えるかも評価したところ、GPT-4ではほとんどのケースで建設的なフィードバックを行えたが、CodeLlamaはそれほど有用でなかった。
- 4つのシステム開発ケースで要件定義書作成の時間を比較したところ、LLMは人間の**7～47倍**の大幅な時間短縮を実現。



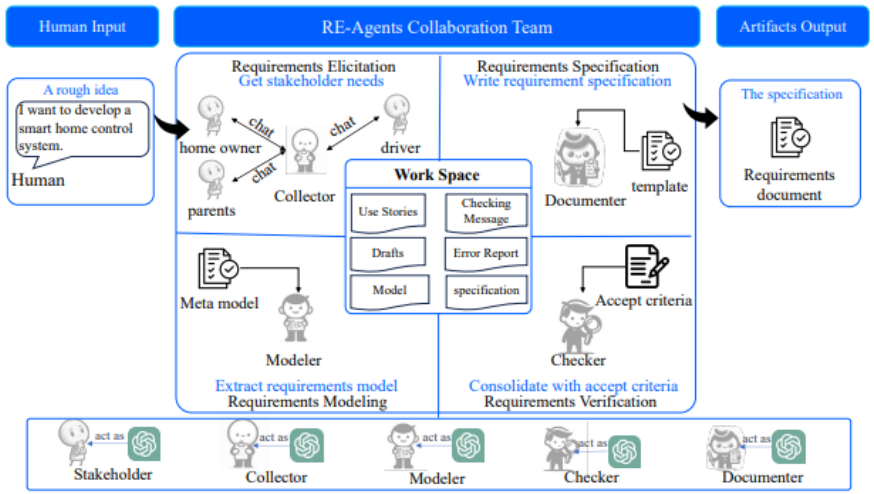
[出所] Madhava Krishna et.al, “Using LLMs in Software Requirements Specifications: An Empirical Evaluation”, arXiv:2404.17842, https://doi.org/10.48550/arXiv.2404.17842 のFig.2より抜粋

自律的な要件定義を行うLLMエージェントの研究例

- 下表のような複数のロールを持つLLMエージェントが協調して、**人間によるラフなアイデアから要件定義を自動化**するツール。
- ATMシステムなど複数の開発ケースで評価、**有用性を示した**。

Agentのロール	アクション
Stakeholder	• 開発システムに関するユーザーストーリー提起 • Collectorからの質問に答える
Collector	• Stakeholdersにニーズについて質問 • 要件のドラフト作成
Modeler	• 要件を示すEntityとRelationのモデルを作成
Checker	• 要件のドラフトやモデルが合格基準を満たすか確認
Documenter	• 要件のドラフトやモデルから仕様書作成 • CheckerがNG判定時にエラーレポート作成

[出所] Dongming Jin et.al, “MARE: Multi-Agents Collaboration Framework for Requirements Engineering”, arXiv:2405.03256, https://doi.org/10.48550/arXiv.2405.03256 のTABLE1を基に日本総研作成



[出所] Dongming Jin et.al, “MARE: Multi-Agents Collaboration Framework for Requirements Engineering”, arXiv:2405.03256, https://doi.org/10.48550/arXiv.2405.03256 のFig.1より抜粋

保守・運用フェーズにおけるLLM活用の研究動向

- まだそれほど多くはないが、**システムの保守・運用を効率化するための研究**も出てきている。
- 既存コードのバグ修正や、セキュリティリスク対応の研究などが主流だが、**自律的なソフトウェア改善の研究**も見られる。

既存コードのバグ修正の研究例：ChatRepair

- テストの失敗情報や以前に生成したパッチ情報を利用したFew-shotプロンプトによって、対話形式で既存コードのバグ修正を行うツール。公開データセット*1を用いて既存ツールと比較したところ、**337個のバグのうち162個**を正しく修正できた。

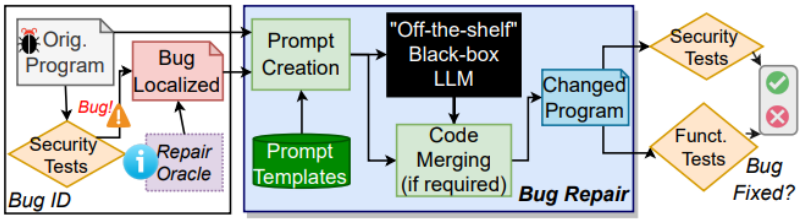
*1 Defects4J(D4J), https://github.com/rjust/defects4j (参照: 2024/9/5)

データセット	正しく修正できたバグの数		
	ChatRepair	ChatGPT	CodexRepair
D4J 1.2	**114**	80	99
D4J 2.0	**48**	25	31

[出所] C. S. Xia, et.al, “Keep the Conversation Going: Fixing 162 out of 337 bugs for $0.42 each using ChatGPT”, arXiv:2304.00385, https://doi.org/10.48550/arXiv.2304.00385 のTable1を基に日本総研作成

セキュリティリスク対応の研究例

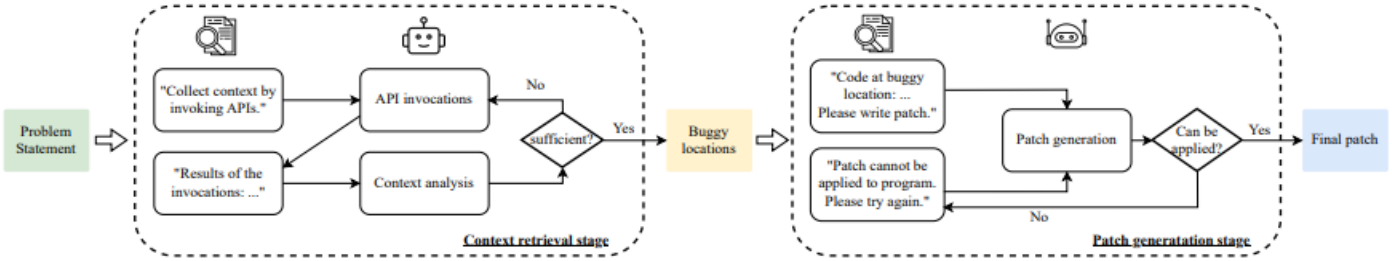
- 脆弱性のあるコードをセキュリティ評価ツールと組み合わせてLLMによって修正できるか検証した研究。
- 結果として正しく脆弱性対応できたケースもあったが、**正確性や複雑なシナリオへの対応などの課題**が示唆された。



[出所] H. Pearce, et.al, "Examining Zero-Shot Vulnerability Repair with Large Language Models“, 2023 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2023, pp. 2339-2356, doi: 10.1109/SP46215.2023.10179324.

自律的なソフトウェア改善の研究例：AutoCodeRover

- 既存のソフトウェアにおいて、機能追加やバグ修正のためのパッチ適用をLLMエージェントにより自律的に行うツール。
- 解決したい問題の説明文(GitHubのイシューなど)とソフトウェアのコードを入力として、コンテキスト検索のLLMエージェントによりコードの修正箇所を特定後、LLMエージェントを利用してパッチ生成と検証を繰り返すことで最終的に問題を解決するパッチを適用する。
- 公開ベンチマーク*2を利用した評価で、GitHubの**57件のイシューをそれぞれ4分未満で解決**でき、人間の開発者の平均である2.68日を大幅に上回る迅速さで修正できた。SWE-benchのフルベンチマーク*3では、**2294件のうち12.4%のイシューを解決**[*4]した。



*2: SWE-bench-lite, https://www.swebench.com/lite.html (参照: 2024/9/5)

*3: SWE-bench, https://www.swebench.com (参照: 2024/9/5)

*4: 記載スコアは論文中の値であるが、SWE-benchのランキングではGPT-4oを利用して18.83%のスコアを達成(参照：2024/9/5)。

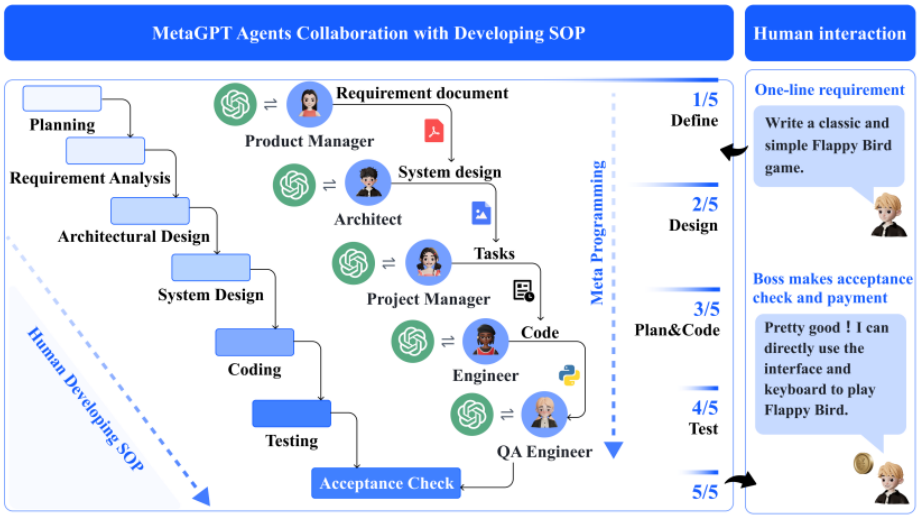
[出所] Yuntong Zhang et.al, “AutoCodeRover: Autonomous Program Improvement”, arXiv:2404.05427, https://doi.org/10.48550/arXiv.2404.05427 のFigure 2より抜粋

AIエージェントによる自律的なシステム開発の研究動向

● 発展途上だが、**AIエージェントが要件定義から実装、テスト、デプロイを自律的に行う研究**もあり、注目されている。

ソフトウェア開発エージェントの研究例：MetaGPT

- 人間が入力した簡単な要件を基に、複数のLLMエージェントが協調して自律的にソフトウェアを開発するツール。
- プロダクトマネージャやアーキテクト、エンジニアなどの役割と作業手順をLLMエージェントに与え、複雑なタスクも解決。
- エージェント間は設計書など定められた成果物をプロトコルに、共有のメッセージプールを介してPub-Sub型で効率的に通信。
- 生成したコードを繰り返し改善するフィードバック・ループを実現し、高品質なソフトウェアを開発する仕組み。
- **公開ベンチマークで良いスコアを出した**他、**人為的な修正コストなど複数のメトリクスの観点でも有用性**が示された。

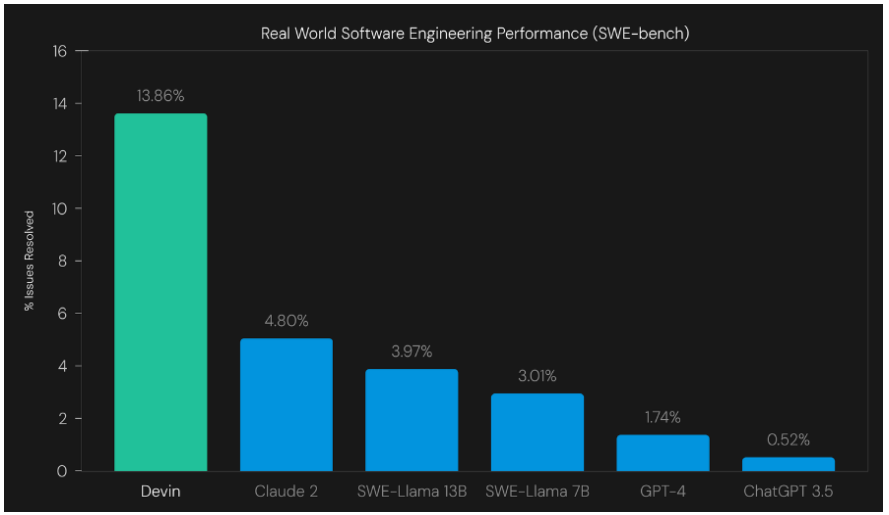


[出所] Sirui Hong et.al, "MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework", arXiv:2308.00352, https://doi.org/10.48550/arXiv.2308.00352 のFigure 1より抜粋

AIソフトウェアエンジニア：Devin AI

- 2024年3月に米国のスタートアップ(Cognition)が発表した、世界初のAIソフトウェアエンジニア。
- 大手のAI企業のコーディング面接に合格したほか、フリーランス向けプラットフォームにおける開発実務タスクを自動で行った。
- GitHub上のオープンソースコードのイシューを基にしたベンチマーク*1では、**2294件のうち13.86%のイシューを解決**し、他モデルと比較して高い値を示した*2。
- Devin AIは現在、アーリーアクセス中。

*1: SWE-bench, https://www.swebench.com (参照: 2024/9/5)
*2: 2024年9月5日時点で、SWE-benchの首位はHoneycombと呼ばれるAIエージェントがマークした22.06%となっている。スコアは短期間で非常に変動している点に注意。Honeycombの詳細は下記参照。https://honeycomb.sh (参照: 2024/9/5)



[出所] Cognition | Introducing Devin, the first AI software engineer, https://www.cognition.ai/blog/introducing-devin (参照：2024/9/5)

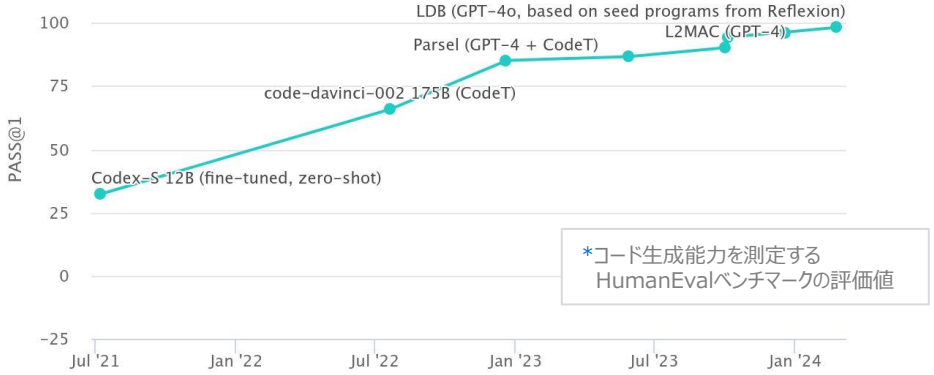
LLMの最新の技術動向

● 今後のシステム開発への生成AI活用に影響を及ぼすと考えられる、LLMの最新技術動向を紹介する。

LLMの継続的な性能改善

最新のLLMが次々と発表され、着実に性能を上げており、**コード生成の性能も向上**してきている。LLMのトークン数制限も徐々に緩和され、**大規模なコードの分析や修正も可能になってきている**。

コード生成の性能変化*



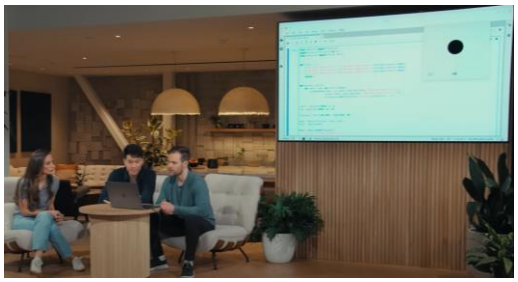
[出所] HumanEval Benchmark (Code Generation) | Papers With Code, https://paperswithcode.com/sota/code-generation-on-humaneval （参照：2024/9/6）

LLMの高速なマルチモーダル処理

テキスト入力のみでなく、**リアルタイムな音声対話が可能**となるなど、AIが**異なる種類のデータをまとめて高速に処理可能**となる。

対話によるプログラミングデモ

OpenAI社は、PCを操作しながら対話によってAI(GPT-4o)からコーディング補助を受けるライブデモ動画を公開した。近い未来に**AIとのペアプログラミングの可能性**を感じさせる。



[出所] Live demo of GPT-4o coding assistant and desktop app - YouTube, https://www.youtube.com/watch?v=mzdvw_euKlk （参照：2024/9/6）より抜粋

LLMの軽量化、ローカルLLMの開発

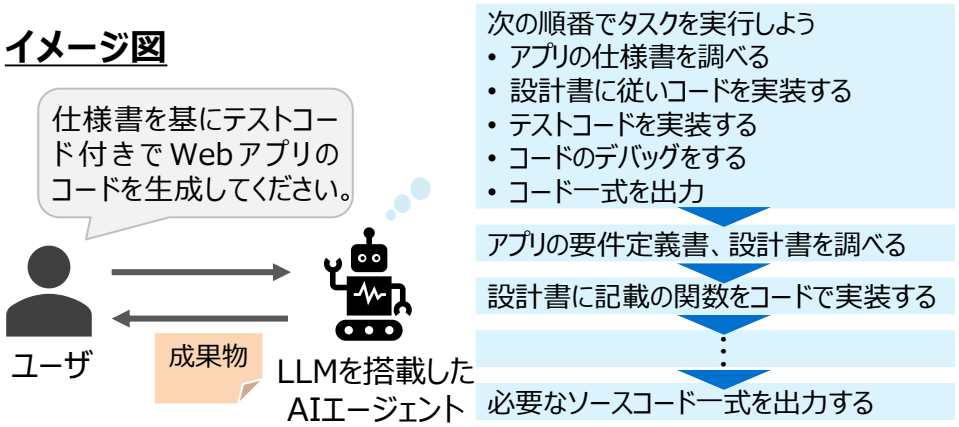
PCやモバイル端末のローカル環境で利用できる軽量なLLMが開発されている。これにより、**情報漏洩のリスクを抑えて生成AIを活用**でき、**生成AIツール導入の後押しとなる可能性**がある。

企業名	ローカルLLM	特徴
Google	Gemma 2	Geminiと同様の技術により学習。 高性能で幅広いハードウェア環境で動作
Microsoft	Phi-3	軽量だが他モデルと同等性能を示す
Meta	Llama 3	ローカルLLMの先駆的存在の後継モデルで、 実用レベルでモデルのカスタマイズが可能。
Apple	OpenELM	非常に軽量でiPhoneやiPadでも利用可

LLMを搭載した自律的なAIエージェントの開発

LLMを搭載した自律的なAIエージェントの開発が進んでおり、**AIによるタスクの自動処理が進む可能性**がある。

イメージ図



最新動向を踏まえた今後のシステム開発の展望

● 今後、システム開発の工程全体で生成AI活用が進み、「人間と生成AIとの協調によるシステム開発効率化」、「生成AIを活用した新たなシステム開発手法の確立」、「IT業界のビジネス構造変革」が実現されることが予想できる。

システム開発向け生成AIサービスの増加、機能拡張	各社における生成AI活用促進	アカデミアの研究活発化	LLMの進化

短期的変化 → 中期的変化 → 長期的変化

人間と生成AIとの協調によるシステム開発効率化

- システム開発の工程全体において**生成AI活用、ツール導入が進み、生産性向上や品質向上が見込める**。
- 生成AIとのリアルタイムな**音声対話によるペアプログラミングやレビューが実現**し、**AIとエンジニアの協業**が進む。
- 自律的な生成AIエージェントによりシステム開発や運用、障害対応など**幅広い工程でタスクの自動化**が進む。
- レガシーシステムのモダナイズなど、**システム開発における実用的な課題への生成AI活用に向けた検討**が進む。

生成AI前提の新たなシステム開発手法の確立

- AIエージェントが中心に実装し、人間が受入テストを行うような、**ローコード・ノーコードな開発手法**が確立される。
- **要件定義や設計の割合が増え**、AIが生成したコードやテストの**レビューが増える**など、人間の**工程の比重が変化**。
- AIの生成物のレビューをしやすいように**システム構成が最適化**され、**マイクロサービス化などが進む**。
- 生成AIで補いきれない要件を捉えた設計能力やコードレビュー能力など、**エンジニアに求められるスキルが変化**する。

IT業界のビジネス構造変革

- 生成AIによるシステム開発支援が本格化し、**自社プロダクト開発やシステム内製化のハードルが下がる**。
- システム内製化の増加や、一部工程の自動化に伴うタスクの変化により、**SIerや受託開発企業には新たなビジネスモデル**が求められる。

生成AI活用におけるリスクと対策

- **生成AIは万能ではなく、リスクも存在**する。生成AI活用のリスクを把握し、必要な対策を講じるべき。特に、**生成AIに頼りきるのでなく、あくまで人間の作業を支援するものという意識**を持ち、それに応じた使い方が重要となる。
- 生成AIは様々なサービスがあり、それぞれ得意・不得意がある。サービス導入時は、**自社状況や開発対象、導入目的などに応じた最適なサービスを選定**すべき。サービス導入は目的でなく、**サービスの有効活用による効率化**が重要。

リスク	システム開発における具体例	対策
ハルシネーションによる誤情報出力	• 実在しないライブラリや関数をコードに記述し、正常に動作しない。 • 誤ったアルゴリズムや不適切な設計を採用し、パフォーマンス低下や意図しない結果を招く。 • コード内に誤ったコメントを挿入し、開発者の誤解を招き、バグの原因となる。	• 生成AIは**不正確な情報を出力する**点を利用者が認識する。特に、**プログラミング言語によってコード生成の性能が変化する**ことに留意する。 • **人間が正誤を確認し、テストするプロセスを必ず含める**。正確な情報を見分けるため、**人間のエンジニアに十分な開発スキルが必要となる**。 • RAGを活用するなど、**ハルシネーションを抑える仕組み**を構築する。
機密情報流出	• 生成AIサービスを個人アカウントで利用してしまい、情報が漏洩する。 • デバッグ時に個人情報を含むログデータを生成AIに入力し、外部に漏洩する。 • 独自アルゴリズムや暗号化手法を生成AIに問い合わせ、特許技術や知的財産が流出する。	• 機密情報や個人情報など、生成AIに入力不可の内容を具体例も含め**ガイドラインとして整備し、常に利用者が確認する**。**個人アカウント利用を禁止**する。 • 利用する生成AIサービスにおいて、**入力データがモデルの学習に利用されないようにオプトアウト**する。オプトアウトできない場合もあるため、**利用規約や仕様を確認し、自社のポリシーに則った設定**をする。 • 必要に応じ、**オンプレミスやプライベートクラウドで使えるサービスを選定**する。
著作権やライセンスの侵害	• AIが出力したコードがOSSのライセンス違反に該当した際、採用すると訴訟等に発展。 • AIが生成したコードが特許技術に基づいた際、知らずにそのまま実装すると特許侵害に繋がる。 • AIが生成したUIデザインが他社のものと類似した際、そのまま採用すると意匠権を侵害する。	• 生成AIの出力が**他者の権利を侵害する可能性**を利用者が認識する。 • 出力内容を人間が精査し、**他者の権利侵害の可能性がないか確認**する。 • **ライセンス面で問題がないデータで学習された生成AIを使用**する。