

### 3. 知財業務の自動化 - ノーコードからの第一歩

## 3. ノーコード自動化 - 概要

この章で学ぶこと

主なトピック:

- 生成 AI によるコード生成とノーコード開発の 2 つのアプローチ
- 特許文献一括処理・検索自動化・明細書作成支援の実装事例
- Dify、n8n、FlowiseAI、LangFlow 等の活用方法
- プログラミング経験がなくても始められる自動化手法
- 効果測定と継続的改善のサイクル

## 3-1. プログラミング不要？生成 AI にコードを書かせる時代

### 生成 AI で解決するプログラミングの壁

#### 従来の課題:

- 学習コスト、専門知識、デバッグの困難、保守負担

#### 生成 AI の解決:

- 自然言語指示: 日本語でコード生成
- 自動デバッグ: AI がエラー特定・修正
- 迅速開発: 数時間で完成度の高いコード
- 業務活用: 定型業務の自動化

## 3-1-2. ノーコード開発の2つのアプローチ

### アプローチ比較と選択指針

#### 1. LLM コード生成パターン

- **特徴:** 自然言語指示で AI がコード自動生成
- **利点:** 高カスタマイズ性、複雑ロジック実装可能
- **適用場面:** 独自ソリューション、高セキュリティ要件

## 3-1-2. ノーコード開発の2つのアプローチ(2)

### アプローチ比較と選択指針

#### 2. ノーコードツール利用

- **特徴:** ビジュアル開発、テンプレート活用
- **利点:** 迅速開発、保守性高、専門知識不要
- **適用場面:** 標準機能、チーム利用、迅速プロトタイピング

**代表ツール:** Dify、n8n、FlowiseAI、LangFlow

## 3-2. 知財業務での自動化事例 - 事例 1: 特許文献の一括処理

### 要件定義

#### 目的

複数の特許文献を一括で処理し、分析結果を自動生成

#### 入力

PDF ファイル（特許明細書）

## 3-2. 知財業務での自動化事例 - 事例 1: 特許文献の一括処理（1）

### 出力・処理内容

#### 出力

CSV ファイル（分析結果）

#### 処理内容

1. PDF からテキスト抽出
2. 技術要素の自動識別
3. 重要度の評価
4. 結果の整理・出力

## 3-2. 知財業務での自動化事例 - 事例 1: 特許文献の一括処理（2）

**参照:** 詳細な生成 AI 指示プロンプトは [code-examples/patent-document-processing-prompt.md](#) を参照してください。

この事例では、特許文献の一括処理システムの開発プロンプトを段階的に構築し、以下の要素を含む包括的な指示を作成します：

- **ロール設定:** Python 開発専門家としての役割定義
- **コンテキスト:** 特許文献処理システムの要件
- **タスク定義:** 具体的な機能要件と技術要件
- **開発手順:** 段階的な実装アプローチ
- **出力形式:** モジュラー設計と制約条件



## 3-2. 知財業務での自動化事例 - 事例 1: 特許文献の一括処理 (3)

### 期待される効果

- **処理時間の短縮:** 手動処理の 1/10 の時間
- **精度の向上:** 一貫した分析基準
- **スケーラビリティ:** 大量文書の一括処理
- **追跡可能性:** 詳細なログとレポート

## 3-3. 事例 2: 特許検索の自動化

### 要件定義

#### 目的

定期的な特許検索を自動化し、新規特許の監視を実現

#### 入力

検索キーワード、検索期間

## 3-3. 事例 2: 特許検索の自動化（1）

### 出力・処理内容

#### 出力

新規特許のリスト、重要度評価

#### 処理内容

1. 特許データベースへの自動アクセス
2. キーワード検索の実行
3. 新規特許の抽出
4. 重要度の自動評価

## 特許検索自動化システムの要件

### 機能要件:

- 特許データベース API 連携
- 定期検索の自動実行
- 新規特許の自動抽出・重要度評価
- 結果の自動通知

### 技術要件:

- Python 3.8 以上、スケジューリング機能、API 連携機能
- データベース連携、通知機能

## 3-3. 事例 2: 特許検索の自動化（2）

### 期待される効果

- **24 時間監視:** 継続的な特許監視
- **見落とし防止:** 自動化による確実な検索
- **効率化:** 手動検索の 90%削減
- **重要度判定:** 自動的な優先度評価

詳細なコード実装については、別資料を参照してください。

→ [特許検索自動化システム - 詳細コード例](#)

## 3-4. 事例 3: 明細書作成支援システム

### 要件定義

#### 目的

明細書作成の効率化と品質向上を実現

#### 入力

発明の概要、技術仕様

## 3-4. 事例 3: 明細書作成支援システム (2)

### 出力・処理内容

#### 出力

明細書ドラフト、品質チェック結果

#### 処理内容

1. 発明内容の分析
2. 明細書構造の生成
3. 各セクションの自動作成
4. 品質チェックの実行

## 3-4. 事例 3: 明細書作成支援システム (3)

### 明細書作成支援システムの要件

#### 機能要件:

- 発明内容の自動分析
- 明細書構造の自動生成
- 各セクションの自動作成
- 品質チェック・修正提案機能

#### 技術要件:

- Python 3.8 以上、自然言語処理ライブラリ
- テンプレート機能、品質評価機能



## 3-4. 事例 3: 明細書作成支援システム（4）

### 期待される効果

- **作成時間短縮:** 従来の 50%の時間で完成
- **品質向上:** 一貫した品質の明細書
- **標準化:** 組織内での書式統一
- **ミス削減:** 自動品質チェックによる確認

詳細なコード実装については、別資料を参照してください。

→ [明細書作成支援システム - 詳細コード例](#)

## 3-5. コードが読めない人でもできるコツ

### 成功のポイント

#### 段階的アプローチ:

1. 小さく始める → テスト → 改善 → 拡張

#### 効果的指示方法:

- 具体的要件、段階的指示、例示提供、制約明示

#### ベストプラクティス:

- バージョン管理、ドキュメント化、テスト、バックアップ

### 3-5. LLM への指示をまとめておく。

- AGENTS.md
  - GEMINI.md
  - cursor.rules
- など。 [参考](#) [48]

## 3-6. 生成 AI を活用したノーコードツール

### ノーコードツールの基本概念

#### ノーコードツールとは

- **プログラミング不要:** コードを書かずにアプリケーション構築
- **ビジュアル開発:** ドラッグ&ドロップで機能を組み合わせ
- **AI 統合:** 生成 AI との連携で高度な機能を実現
- **迅速開発:** 数時間で本格的なアプリケーション構築

## 3-6. 生成 AI を活用したノーコードツール（1）

### 知財業務での活用

- 自動化ワークフロー: 定型業務の自動化
- データ処理: 大量データの効率的処理
- レポート生成: 自動レポート作成システム
- 通知システム: 重要情報の自動通知

## 3-6. 生成 AI を活用したノーコードツール（2）

### 1. Dify

[公式](#) [49] 動画を見てもらおうとイメージがつかめて良いと思う

- **特徴:** AI アプリケーション構築プラットフォーム
- **用途:** チャットボット、AI アシスタント、ワークフロー自動化
- **利点:** 直感的な UI、豊富な AI モデル対応
- **知財業務での活用:** 特許調査アシスタント、明細書作成支援

## 3-6. 生成 AI を活用したノーコードツール（3）

### 2. n8n

[公式](#) [50] 動画を見てもらおうとイメージがつかめて良いと思う

- **特徴:** オープンソースのワークフロー自動化ツール
- **用途:** システム連携、データ処理、API 統合
- **利点:** 高度なカスタマイズ、無料で利用可能
- **知財業務での活用:** 特許データ収集、定期レポート作成

## 3-6. 生成 AI を活用したノーコードツール（4）

### 3. FlowiseAI

[公式](#) [51] 動画を見てもらおうとイメージがつかめて良いと思う

- **特徴:** LangChain ベースの AI アプリケーション構築
- **用途:** RAG システム、AI エージェント、チャットボット
- **利点:** 専門的な AI 機能、柔軟なカスタマイズ
- **知財業務での活用:** 特許文献検索、技術動向分析
- LangChain ベースなのでエコシステムが膨大で自分で書き足す場合も便利でおすすめ。



## 4 LagnFlow

[公式](#) [52] 動画を見てもらおうとイメージがつかめて良いと思う

- **特徴:** LangChain の GUI ツール、ドラッグ&ドロップで開発
- **用途:** RAG システム、AI エージェント、チャットボット
- **利点:** アイデアの試行錯誤が容易
- **知財業務での活用:** 特許文献検索、技術動向分析
- Hugging Face で試せる、活発なコミュニティ

## 3-7. Dify を使った知財業務の自動化

Dify の基本設定

アカウント作成

コード例: `code-examples/patent-document-processor.md` の「Dify 設定」セクションを参照

## 3-7. Dify を使った知財業務の自動化（1）

### 主要機能・活用例

#### 主要機能

- **チャットアプリ**: 対話型 AI アプリケーション
- **ワークフロー**: 自動化フローの構築
- **データセット**: 独自データの学習
- **API 連携**: 外部システムとの連携

#### 特許調査アシスタントの構築

コード例: `code-examples/patent-search-automation.md` の「Dify 特許調査アシスタント」セクションを参照

## 3-8.n8n を使ったワークフロー自動化

n8n の基本設定

インストール・設定

コード例: `code-examples/patent-search-automation.md` の「n8n 設定」セクション  
を参照

## 3-8.n8n を使ったワークフロー自動化（1）

### 主要機能・活用例

#### 主要機能

- ノード: 各種サービスとの連携
- ワークフロー: 処理フローの構築
- スケジューリング: 定期実行の設定
- エラーハンドリング: 異常時の対応

#### 特許データ収集の自動化

コード例: `code-examples/patent-search-automation.md` の「n8n 特許データ収集」セクションを参照

## 3-8.n8n を使ったワークフロー自動化（2）

### ワークフロー例・拡張機能

#### 特許監視システム構成:

1. 定期実行トリガー → 特許庁 API → 新規特許抽出
2. AI 重要度評価 → 結果通知・保存

#### 設定ポイント・拡張機能:

- エラーハンドリング・重複チェック
- Slack 連携・Excel 出力
- 重要度フィルタリング

## 3-9. FlowiseAI を使った RAG システム構築

FlowiseAI の基本設定

インストール・設定

コード例: `code-examples/patent-specification-generator.md` の「FlowiseAI 設定」  
セクションを参照

## 3-9. FlowiseAI を使った RAG システム構築（1）

### 主要機能・活用例

#### 主要機能

- **チャットフロー**: 対話型 AI の構築
- **ベクトルストア**: 文書データの管理
- **ツール連携**: 外部 API との連携
- **カスタマイズ**: 高度なカスタマイズ

#### 特許文献検索システム

**コード例:** `code-examples/patent-specification-generator.md` の「FlowiseAI 特許文献検索」セクションを参照







## 3-10. 追加 1 : Not Another Workflow Builder

### 参照

- 単純なタスク、例えば文章の要約や簡単な質疑応答などは、専門知識が不要なノーコードのエージェントで十分に対応できます。
- しかし、複数のステップや条件分岐が必要な複雑なタスクでは、**視覚的なビルダーは管理が難しく**なり、コードで直接ロジックを記述する方が**柔軟性と制御性**が高まります。
- そのため、LangGraph のようなコードベースのフレームワークが、予測可能で信頼性の高い複雑なワークフローを構築するのに適している。

## 3-10 追加 2: OpenAI Agent builder と Google opal

項目	OpenAI Agent Builder	Google Opal
 基本コンセプト	視覚的なキャンバスで、複雑な AI エージェントのワークフローを構築するための開発者・企業向けプラットフォーム。	自然言語で作りたいアプリを記述するだけで、AI が自動でワークフローを生成する実験的なノーコード・ミニアプリビルダー。
 操作方法	フローチャートのように、ドラッグ & ドロップでノード（機能部品）を繋ぎ合わせてワークフローを設計する。	「こういうアプリが欲しい」と文章で指示すると、AI が自動でワークフローを生成。その後、視覚的な編集も可能。

項目	OpenAI Agent Builder	Google Opal
 主な対象者	<p>開発者、AI エンジニア、企業の IT 部門。本番環境への導入を視野に入れているユーザー。</p>	<p>非技術者、ビジネスユーザー、企画担当者。迅速なプロトタイピングや業務効率化を目指すユーザー。</p>
 主な特徴	<ul style="list-style-type: none"> <li>・複雑な条件分岐や複数エージェント連携が可能</li> <li>・バージョン管理、監査ログなどの企業向け機能が充実</li> <li>・パフォーマンス評価やテストの仕組みが組み込まれている</li> <li>・コードの統合も可能</li> </ul>	<ul style="list-style-type: none"> <li>・専門知識が不要で、アイデアを即座に形にできる</li> <li>・Google Workspace との深い連携（ドキュメント、スプレッドシート、ドライブなど）</li> <li>・テンプレートが豊富で手軽に始められる</li> </ul>

## 3-11 Agent Skills

- [Agent Skills](#)
- Anthropic の新機能「Agent Skills（以下、Skills）」は、指示・スクリプト・リソースをフォルダ単位でまとめ、Claude に“能力パック”として付与できる仕組みです。ノーコード的に始められる既製スキルもありますが、API や Claude Code で本格活用する場合はコード実行ツールの有効化が前提です。部署横断で再利用でき、運用コストの削減と出力の一貫性向上が期待できます。[【速報】Anthropic が発表した「Agent Skills」がいい！](#)