目次

第 1 章　序論 2

第 2 章　関連研究 4

2.1　ドキュメントとソースコードの一元管理 4

2.2　ドキュメント自動生成 4

2.3　本研究との相違点 5

第 1 章

序論

近年のソフトウェア開発では，ソフトウェア開発における複雑性が増し，少人数の開発チームへと分割されていく中で，開発者は短期間でより多くのを作業をしなくてはらならない状況にある．そのため，分割されたチーム間で円滑な連携を行うことや，ソフトウェア品質を保つためにドキュメントを整備することが重要視されている．また，2020 年初旬からの新型コロナウィルス感染症拡大により，ソフトウェア開発の現場ではリモートワークを中心としたテキストベースでのやり取りも増えたため，より一層ドキュメントを充実させる必要性が高まった．このような状況において，開発スピードを求められるソフトウェア開発の現場では，ドキュメントとソースコードの整合性を維持することが困難となってきた．

ソフトウェア開発の現場では，バージョン管理システムを使用したソフトウェアのバージョン管理，CI/CD(継続的インテグレーション/継続的デリバリー) を活用したソフトウェアのビルド・テスト・デプロイなどの自動化，円滑なやりとりを行うためのチャットツールの導入など，開発スピードの向上に力を入れている．また，同じ開発チームであったとしても，フロントエンドの開発，バックエンドの開発，モバイルアプリケーションの開発，デザインの調整など開発者の役割は様々であり，開発チームの内外問わずに円滑なやりとりを行う必要がある．このため，ソフトウェア開発で使用するドキュメントは常に最新のものであることが求められている．

開発者は，ビジネス要求を素早く実現するために，機能の追加や修正を優先してしまい，ドキュメントの更新を後回しにしてしまう事例は少なくない．例えば，誤って古いドキュメントを参照して開発を進めた場合，後の段階で手戻りが発生していしまうため，開発スピードを大幅に落としてしまう原因となってしまう．また，普段と異なる開発環境では開発者は自身の能力を最大限発揮することは難しい．したがって，ライブラリやフレームワーク，エディタに依存せずに，既存のソフトウェア開発プロジェクトに無理なく組み込むことができるツールを作成する必要があると考えた．

以上を踏まえた上で，ドキュメントとソースコードが乖離している可能性のある箇所を特定および開発者に通知をすることができる機能をもった支援ツール milk を開発した．milk はライブラリやフレームワーク，エディタに依存しないツールである．同時に，milk の有用性を検証するために，Git のコミット履歴を使用したシミュレータの開発を行い，実際のソフトウェア開発を想定し

たシミュレーションを行った．シミュレーションで検証できなかった機能については，実際にプログラミング経験が豊富な実験協力者に milk を使用してもらい，その有用性を確かめる評価実験を行った．

本論文の構成は以下のとおりである．2 章で，本研究の関連研究について示し，本研究との相違点を述べる．3 章では本研究で提案するツールについて，4 章ではその設計と実装について述べる．5 章では，提案ツールの有用性検証するためのシミュレーションおよび評価実験について述べる．6 章では，評価実験の結果と考察について述べ，7 章で本研究の結論を述べる．

第 2 章

関連研究

本章では，本研究に関連する研究および技術と，本研究との相違点について述べる．

2.1 ドキュメントとソースコードの一元管理

赤石らの提案するドキュメントとソースコードの一元管理ツールは，開発者が XML 形式のドキュメントを作成した後，それに従ってソースコードを作成することで，ドキュメントとソースコードの整合性を維持することを期待する．作成したドキュメントとソースコードは，ビュワーを通じて閲覧を行うことができる．ビュワーではドキュメントとソースコードが混在した形で近い場所に表示されるため，開発者はドキュメントとソースコードの乖離に気づきやすい．しかし，ドキュメントとソースコードの整合性を維持するためには，赤石らの提案するエディタおよびビュワーを使用しなくてはいけないため，開発者の学習コストや開発スピードの低下などが懸念される．また，ドキュメントとソースコードの対応付けを行う方法や，提案ツールの評価，ビュワーの見た目などが未実施である．

2.2 ドキュメント自動生成

ソースコード中に適切なアノテーションやコメント文を挿入することで，ドキュメントを自動生成することのできるライブラリやフレームワークが存在する．それぞれのライブラリやフレームワークでドキュメントを自動生成するための書き方は異なるが，いずれも適切な箇所で適切なフォーマットでコーディングする必要がある．以下では，ドキュメントを自動生成することができるライブラリおよびフレームワークを紹介する．

2.2.1 Javadoc

Javadoc は，JDK に標準搭載された機能で，適切なフォーマットに従って作成したコメント文やソースコードから HTML 形式のドキュメントを自動生成するツールである．プログラミング言語に Java を使用したソフトウェア開発では，ソースコードの API 仕様書としてに Javadoc が使

用されることも多い．

2.2.2 Spring Boot

Spring Boot は，Web アプリケーション開発を行うことのできる Java の Web フレームワークである．REST API を実装するときに，Swagger と呼ばれるドキュメントを自動生成する機能が存在する．

2.2.3 FastAPI

FastAPI は，API サーバーを構築することのできる Python の Web フレームワークである．FastAPI には Swagger と ReDoc と呼ばれるドキュメントを自動生成する機能が存在する．

2.3 本研究との相違点

赤石らの研究で提案されたドキュメントとソースコードの一元管理を行うツールでは，目的は同じであるものの，プログラミング言語やエディタが限定されるのに対し，本研究では，ライブラリやフレームワーク，エディタに依存せずに，既存のソフトウェア開発プロジェクトに無理なく組み込むことができるといった点で異なる．ドキュメントを自動生成するライブラリやフレームワークでは，技術選定の段階で制限されてしまうのに対し，本研究では，現在進行中のプロジェクトに無理なく組み込むことができるといった点で異なる．

ソースコードに直接関係のないドキュメント，例えば，開発プロジェクトのアーキテクチャの概念図やプロジェクトの方向性，補助的な説明などを記載するドキュメントは，通常ソースコード中には記載せずに，別途ドキュメント専用のファイルなどに記載するため，自動生成がすべてをカバーできないことに留意する．これらのドキュメントも，プロジェクトが進み古くなってしまうと，ドキュメントとソースコードが乖離してしまう可能性がある．

本研究ではドキュメントとソースコードが乖離している可能性のある箇所を特定および開発者に通知をすることができる機能をもったツールの開発を行う．