Title	アジャイル開発の要求管理履歴データに基づく開発遅延予測モデル
Sub Title	Development delay prediction models based on requirements management history data of agile development
Author	岩本, 佑太(Iwamoto, Yūta) 砂原, 秀樹(Sunahara, Hideki)
Publisher	慶應義塾大学大学院メディアデザイン研究科
Publication year	2019
Jtitle	
JaLC DOI	
Abstract	
Notes	修士学位論文. 2019年度メディアデザイン学 第734号
Genre	Thesis or Dissertation
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=KO40001001-00002019-0734

修士論文 2019 年度

アジャイル開発の要求管理履歴データに基づく開発遅延予測モデル

慶應義塾大学
大学院メディアデザイン研究科

岩本　佑太

本論文は慶應義塾大学大学院メディアデザイン研究科に
修士 (メディアデザイン学) 授与の要件として提出した修士論文である。

岩本　佑太

研究指導コミッティ：

砂原 秀樹 教授　(主指導教員)
加藤 朗 教授　(副指導教員)

論文審査委員会：

砂原　秀樹 教授　(主査)
加藤　朗 教授　(副査)
大川　恵子 教授　(副査)

修士論文 2019 年度

アジャイル開発の要求管理履歴データに基づく開発遅延予測モデル

カテゴリ：サイエンス / エンジニアリング

論文要旨

昨今では，より柔軟かつスピードの早いソフトウェア開発が求められるようになりアジャイル開発の重要性が高まってきている．しかし，アジャイル開発も万能ではなく，アジャイル開発プロジェクトの成功確率は 40%程度となっており，不適切な要求管理がソフトウェア開発の納期遅延を引き起こすことはこれまでに様々な研究によって指摘されている．しかし，要求管理上のどのような要因が開発遅延を引き起こすのかは，定量的な観点での調査がされていない．

そこで本研究では，アジャイル開発の要求管理履歴データと各要求の開発遅延の有無を統計的に分析して，要求管理上の要因と開発遅延の関係性を統計モデルの構築によって評価した．ロジスティック回帰モデルを用いて，各要求の要求管理履歴データを説明変数，各要求に対する開発遅延の有無を目的変数とした開発遅延予測モデルを構築し分析を行った結果，開発遅延につながる要求管理上の要因としてステークホルダー数，要求調整回数の 2 要因が統計的に有意であった．また Hosmer-Lemeshow 検定によってモデルの適合度の高さが確認できた．これらの結果によって，要求管理履歴の定量的な計測データから信頼性のあるモデルの構築ができることを確認した．

キーワード：

要求管理, アジャイル開発, ロジスティック回帰, 開発遅延予測モデル

慶應義塾大学大学院メディアデザイン研究科

岩本　佑太

Abstract of Master's Thesis of Academic Year 2019

Development Delay Prediction Models Based on Requirements Management History Data of Agile Development

Category: Science / Engineering

Summary

In recent years, Agile Development is high importance so that more flexible and faster software development has been required. However, this is not panacea. The success rate of agile development project is only 40%. Various researches pointed out that the delay in delivery on projects and development tasks are caused by irrelevant requirement specification. On the other hand, this issue has not been studied what factors in requirements management cause the delay of development in quantity basis.

In this project, we analyzed statistically the request management history data of Agile development and the occurrence of development delay for each request. After that, the relation between requirements management factors and development delay by constructing a statistical model is evaluated.

With a logistic regression model, we have constructed the "development delay prediction model", based on a explanatory variable used as request management history data for each request and an objective variable used as the presence of development delay for each request. As a result of analysis of this model, as a factor on requirements management leading to development delays, two factors were statistically significant, which were a number of stakeholders and a number of required adjustments. In addition, the fidelity of the model is confirmed with

hosmer-lemeshow test. Therefore it is possible to develop a dependable model from the quantitative measurement data of requirements management history.

Keywords:

Requirements Management, Agile Development, Logistic Regression, Development Delay Prediction Models

Keio University Graduate School of Media Design

Yuta Iwamoto

目　　次

図　目　次

表　目　次

第 1 章
序　　論

1.1. 研究背景

2018 年度の企業の IT 投資額は過去 10 年で最高水準の伸び幅を記録した [1]．デジタルトランスフォーメーションという概念の浸透と共に，業種を問わず IT を用いて企業競争力を高めようとしている企業が増え，その流れで IT 部門を内製化している企業が増えている．そのような背景の中，より柔軟かつスピードの早いソフトウェア開発が求められるようになり，アジャイル開発の重要性が高まってきている．

Bertrand Meyer(2018) *Ajaile Introduction*（邦訳『アジャイルイントロダクション』）[2] では，アジャイル開発とは，単一の開発手法を示すものではなく，エクストリーム・プログラミング（XP），スクラム，リーンソフトウェア開発，クリスタルといった独立した開発手法があり，多くの場合はそれぞれの手法を完全には理解せずに，それらの断片的な要素を組み合わせて適用している開発手法の総称である. と定義しており，初めから全ての厳密な仕様は決めずに，ニーズや要求の変化に適応的に開発を行っていく点が特徴である．

しかしアジャイル開発も万能ではなく，2015 年の Standish Group の調査によるとプロジェクトの成功品質を測る指標である QCD の観点から，一切問題なくアジャイル開発プロジェクトが成功する確率は 40%程度となっており，半分以上のプロジェクトで機能の不足，予算の超過，納期の遅延といった問題が発生し，プロジェクトが失敗していることが明らかとなっている [3]．IPA の調査では上記の失敗の要因として，アジャイル開発上の各反復期間において，管理者やプロダクトオーナーが開発項目に対する優先順位決定・仕様策定といった要求管理を適応

的に行っていくことの難しさが原因であると指摘し，それを上手く行うことがプロジェクト成功のポイントだと指摘している [4].

このような要求仕様の曖昧さや要求管理の煩雑さが，プロジェクトや開発タスクの納期遅延や品質低下を引き起こすことはこれまでに様々な研究によって指摘されている [5]．しかし，要求管理上のどのような要因が開発遅延を引き起こすのかという点については言及がされていない．またこれらの先行研究の多くは，間隔尺度等を用いたアンケートによって得られた評価値を元に分析されており，主観によって評価に揺れが生じるため，分析結果の信頼性に乏しいという指摘もある [6]．そこで私はアジャイル開発における要求管理履歴に着目し，各要求に対する要求管理履歴のログデータとその要求の開発項目の開発遅延の有無をデータとして収集することで，要求管理における各要因と開発遅延を統計的な手法によってモデル化することができるのではないかと考えた．またこの方法を用いることで，ログデータを用いた定量的な分析により主観を排除できること，進行中のプロジェクトにも適用できることから汎用的かつ信頼性の高い分析結果が得られるのではないかと考えた．

また要求に着目した先行研究では，藤原による PRINCE モデル[1]を用いたプロジェクトにおける要求成熟過程の分析がある．この研究では要求種別毎に要求成熟過程が違う点が指摘されている．しかし，要求種別毎に違う要求獲得過程を経た要求項目が，開発工程にどのような影響を及ぼしたのかは調査されていない．ここから，要求の種別によっても開発遅延への影響度合いに差があるかもしれないと推測し，各要求種別毎にも分析を行うべきだと考えた．

以上を踏まえて，本研究ではアジャイル開発における要求管理者の要求管理履歴のログデータを収集し，統計的な手法を用いてモデル化することで，要求管理履歴に基づく開発遅延予測モデルを構築し，要求管理工程上の要因と開発遅延の関係性を評価することを研究目的とした．

1 Nakatani (2009) 『PRINCEMODEL [7] で提案されている要求獲得モデル

1.2. 研究の目的

本研究ではアジャイル開発における要求の開発遅延につながる要求管理工程上の要因を定量的な測定項目から明らかにしたい．そのために要求管理履歴のログデータに着目し，各要求の要求管理上のログデータとその要求の開発遅延の有無を統計的な手法を用いてモデル化することで，要求管理履歴に基づく開発遅延予測モデルを構築できるのではないかと考えた．開発遅延予測モデルを構築し，要求管理工程上の要因と開発遅延の関係性をモデルを用いて評価すること，また要求種別毎に要因や影響度合いに変化があるのかを明らかにすることを研究の目的とする．

1.3. 研究の範囲

本研究ではアジャイル開発の要求管理工程の要求管理履歴データから開発遅延予測の統計モデルを構築し評価することを目的としているため，エンジニアのスキルやプログラミング手法の良し悪しといった開発工程における開発遅延要因は本研究の研究範囲外とし，要求管理工程を研究対象とする．また世の中にある様々な開発手法のうち，アジャイル開発プロジェクトを研究対象としている．

第 2 章
関 連 研 究

本章では，本研究を行う上で重要な概念であるアジャイル開発や要求管理について整理し，それらの分野の先行研究について述べる．その上で本研究で明らかにしたい課題を先行研究を踏まえた問題提起として簡潔に述べる．

2.1. アジャイル開発

2.1.1 アジャイル開発の定義

アジャイル開発とは，要求仕様や設計の変更があるという前提に立ち，初めから全ての厳密な仕様は決めず，イテレーションと呼ばれる 1〜4 週間程度の短い反復期間の中で「要求仕様の決定」「設計」「開発」「テスト」「リリース」という工程を経て，定めた範囲の機能を開発しリリースする．プロジェクト全体の状況やビジネス要求を考慮しつつ，最も優先度高く作るべき範囲を決定し，イテレーション内でその範囲の要求を満たすソフトウェアを開発しリリースする．これを反復的に繰り返すことで，各イテレーションの開始時に最新のプロジェクト状況を鑑みて要求仕様の決定を行うことで，要求の追加や変更に迅速かつ適応的に開発を行っていく開発手法である．イテレーションを繰り返す毎に，要求の追加や変更に適応しつつ，新たな機能を開発していく「反復増加型」の開発プロセスでアジャイル開発は行われていく．

2001 年，当時は個別の手法として提唱されていた様々なアジャイルソフトウェア開発手法を用いていた 17 名のエンジニアが一同に介し，それまで個別な手法として展開されていたアジャイルソフトウェア開発手法に関して議論を行い，重要

点を整理し，アジャイルソフトウェア開発の考え方の規範をまとめたものとして「アジャイルソフトウェア開発宣言」がある．そこに示されている規範を引用する．

アジャイルソフトウェア開発宣言

私たちは，ソフトウェア開発の実践
あるいは実践を手助けをする活動を通じて，
よりよい開発方法を見つけだそうとしている．
この活動を通して，私たちは以下の価値に至った．

プロセスやツールよりも個人と対話を，
包括的なドキュメントよりも動くソフトウェアを，
契約交渉よりも顧客との協調を，
計画に従うことよりも変化への対応を，

価値とする．すなわち，左記のことがらに価値があることを
認めながらも，私たちは右記のことがらにより価値をおく．

『アジャイルソフトウェア開発宣言 [8]』

このような価値規範に則って様々なプラクティスを取り入れながら適応的にソフトウェア開発を行っていくのがアジャイル開発という手法である．

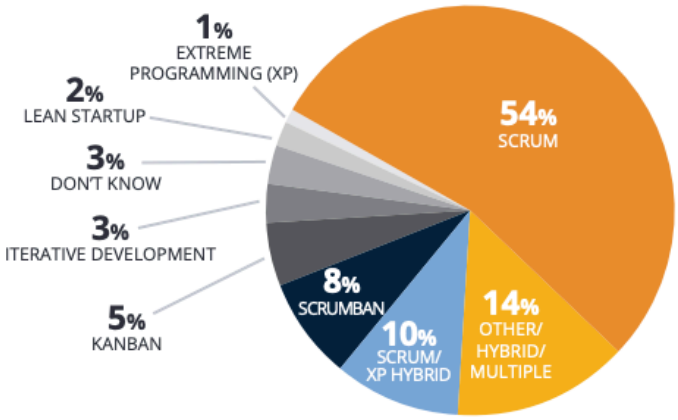
2.1.2 アジャイル開発の種類

アジャイル開発は，おおまかにいくつかの開発手法に分けられる．名称とそれぞれの特徴を以下の表 2.1 に示す．それぞれの手法ごとに特徴を持っており，XP のように開発工程に重心を置いたプラクティスを多く持つ手法もあれば，スクラムのようにプロジェクト運用や組織的な側面に対するプラクティスも内包した手法も存在する．また，具体的なプラクティスを多く持つ手法もあれば，クリスタルのように具体的なプラクティスはあまり持たず，抽象的な価値観や行動規範のみに留めた手法も存在している．

表 2.1 アジャイル開発手法の種類と特徴

開発手法名	作者	特徴
XP	ケント・ベックら	5 つの価値と 19 の具体的なプラクティス によって定義されている． 開発工程中心のプラクティスが多い．
スクラム	サザーランド	アジャイル開発手法のなかで最も 流行している．XP の技術的な側面に加えて プロジェクト・組織的な側面の プラクティスを踏まえた包括的な手法である．
リーン	ポッペンディーク	トヨタ生産方式の流れを汲む リーン生産方式を ソフトウェア開発に適用した方法論である．
クリスタル	アリステア	7 つの規律によって形作られている． XP のような技術的な側面や スクラムのような組織的な側面の プラクティスを包括的に持つ手法ではない．

またアジャイル開発自体が様々なプラクティスやフレームワーク群の総称であり，表 2.1 に示した開発手法をかけ合わせたハイブリット手法や，会社独自のアジャイル開発手法を用いて開発を行っている現場もある．アジャイルレポートでは，毎年アジャイル開発の導入事例や開発手法の導入割合が調査報告されている．最新の 2019 年の調査によると，図 2.1 に示すようにアジャイル開発の半分はスクラムという手法が用いられており，スクラムのプラクティスを何らかの形で取り入れているプロジェクトも含めると全体の 72%ものプロジェクトがスクラムの手法を取り入れたプロジェクトとなっている．



13th Annual State Of Agile Report [9] より引用

図 2.1 アジャイル開発手法の利用割合

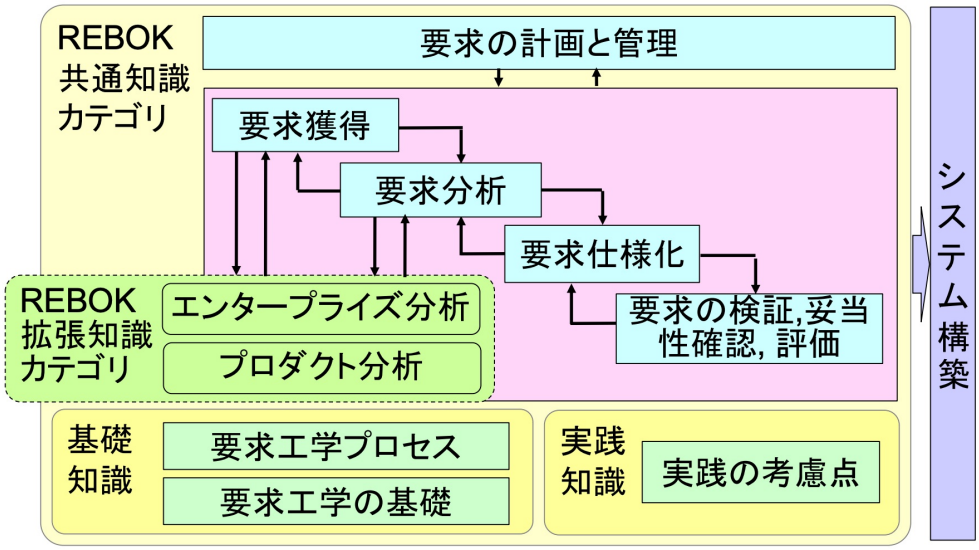
2.2. 要求管理

2.2.1 要求工学における要求管理

古山らの研究にあるように，要求仕様の曖昧さや要求管理の煩雑さが，プロジェクトや開発タスクの納期遅延や品質低下を引き起こすことはこれまでに様々な研究によって指摘されている [5]．そのような背景もあって，要求工学と呼ばれる分野が生まれ，徐々に注目度が上がってきている．

要求工学 (Requirements Engineering) は，システムやコンピュータを組み込んだサービスへの顧客要求の獲得，分析，仕様化，妥当性の確認などを組織的に行うための技術体系である [10]．ソフトウェア開発における最も上流にある工程として要求定義が確立され，要件定義のための技術やプロセスとして，ソフトウェア工学の一分野として研究が進められてきた．IPA の調査結果による指摘にもある通り，プロジェクトの失敗原因の多くは要求に起因しており，システム開発を成功に導くために欠かせない技術となっている．要求工学によって明らかにされた知見や，産業界の現場で培われた技術などを元に，要求定義を正しく行うガイドライン要求工学知識体系 (REBOK) が存在している．

REBOK における要求工学プロセスは図 2.2 のような構成となっている．プロセス知識領域が要求工学プロセスを構成しており,「要求獲得」,「要求分析」,「要求仕様化」,「要求の検証，妥当性の確認，評価」の 4 知識領域は，要求工学で一般に用いられている繰返し型プロセスとして構成されており，要求を獲得し，分析して仕様化し，要求の妥当性を評価するフローを繰り返し行う一連のプロセスを計画し管理するという流れになっている．要求工学知識体系でまとめられている知識やプロセスは特定の開発手法に特化した内容ではないため，アジャイル開発手法における要求管理とどのように対応付けしていくか，という点は課題である．



要求工学の動向と要求工学知識体系 REBOK [11] より引用

図 2.2 REBOK における要求工学プロセス

2.2.2 要求獲得と PRINCE モデル

中谷らは要求変更を避けるのは不可能であるという前提の元，要求を獲得，観測，適切な管理を目指す要求獲得モデルである PRINCE モデルを提唱している．PRINCE モデルでは表 2.2 のような形で要求種別を分類している．藤原はこの PRINCE モデルの要求分類を一部変更しつつ，PRINCE モデルに則ったプロジェ

表 2.2 PRINCE モデルにおける要求種別分類

ITC による REBOK(要求工学知識体系) 適用研究報告書 [12] より引用

No.	種別	説明	種別	要求種別
1	TypeSt（戦略的要求）	事業環境の変化，事業の優位性を保つ・確保するための要求である．たとえば，ビジネスに負けないために，他社や自社の競合製品と同等の製品，あるいはそれらを超えるために求められる要求が相当する．	F	TyoeStF 戦略的機能要求
2			NF	TypeStNF 戦略的非機能要求
3	TypeBz（ビジネス支援要求）	業務の効率，生産性，信頼性を向上させるための要求である．	F	TyoeStF ビジネス支援機能要求
4			NF	TypeStNF ビジネス支援非機能要求

クト分析を行い，要求種別によって要求成熟過程が異なること，特定条件下における要求変更率の高さが要求成熟の早さにつながることを示している [13]．しかし，これらの先行研究では成熟した要求がその後の開発工程において問題なく開発ができたのか，また要求種別毎の開発工程への影響は考慮されておらず，あくまで要求成熟過程における考察となっている．また，後述するアジャイル開発における要求記述手法においては，PRINCE モデルの要求種別がそのまま利用できないという問題もある．

2.2.3 アジャイル開発における要求記述手法

アジャイル開発における要求記述手法としてユーザーストーリーと呼ばれる手法がある．ユーザーストーリーとは，様々な種類の要求項目に期待されるビジネス価値を表現する形式で，ビジネス側と技術者側の両者が理解しやすいように作られている [14]．ユーザーストーリーでの要求記述の例を図 2.3 に示す．ユーザーストーリーでは要求の概要を**誰が（対象・役割），なにができる（達成したいゴール，サービス），その理由（課題，目的，価値)** というような形で記述する．開発するソフトウェアの対象者が価値を理解できる文章で表現し，そのユーザーストーリーを実現するための詳細な仕様，画面遷移フローや画面のデザイン，受け入れ基準等の項目を策定し，それらが適切な形で記述されて，一つのユーザーストーリーとして完成する．画面イメージや詳細な技術仕様は別ドキュメントで記載される場合もある．要求が大きすぎると一つのユーザーストーリーに必要な画面や機能が膨大になり，仕様設計やテスト項目に抜け漏れが発生するため，適切な粒度でユーザーストーリーを分割する必要がある．アジャイル開発ではユーザーストーリー単位で開発が行われる．

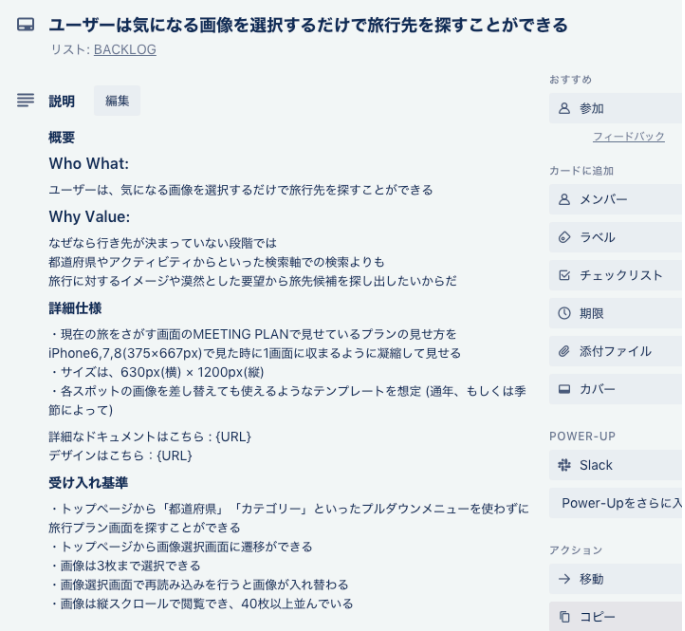


図 2.3 ユーザーストーリーでの要求記述例

2.2.4 アジャイル開発における要求管理手法と管理フロー

アジャイル開発における要求管理手法としてプロダクトバックログという手法がある．プロダクトバックログとは 2.2.3 で説明したユーザーストーリーによって記述された要求項目を開発優先順に並べたリストのことである．チームメンバーやステークホルダーはこのプロダクトバックログを見ることで，なにをどのような優先順位で開発しようとしているかを把握できる．またこのプロダクトバックログに適切な内容のユーザーストーリーを策定し，適切な優先順位で並べるのは要求管理者の責務である．要求管理者は日々プロジェクトが進行する中，様々な要求を適切に汲み取り要求に準じた仕様策定や優先順位決定，またはそれらの変更といった業務を行い，適切にプロダクトバックログを更新し続けなければいけない．

このプロダクトバックログを基点とする要求獲得から開発完了までの一般的な要求管理フローを図 2.4 に示す．図 2.4 の左側にある IDEA リストは，要求の形をなしていない要望やアイディアを入れておく場所であり，様々なステークホルダー

から収集した要望やユーザーからのフィードバックで獲得した要望，アイディアが随時入れられる．IDEA リストは要求管理者だけではなく，チームメンバーやその他ステークホルダーも起票することができる．

図 2.4 の左から 2 番目にある PRODUCT BACKLOG リストは，前述したプロダクトバックログを指している．要求管理者は IDEA リストにある様々なアイディアや要望を整理し，プロジェクトの目的や状況を鑑みて，適切な要求単位にまとめあげ，それを満たす詳細な仕様や受け入れ基準を策定し，ユーザーストーリーとして完成させる．完成させたユーザーストーリーを適切な開発優先順位でプロダクトバックログに並べる．図 2.4 の右側にある TODO，DOING，DONE リストはまとめてスプリントバックログと呼ばれる．スプリントバックログには各イテレーション内で開発すべきユーザーストーリーが状態ごとに並べられている．

TODO リストにはイテレーション内で開発を終える予定で開発未着手状態の要求が並んでいる．イテレーションの開始時に，前期間の残タスクやプロダクトバックログにある優先順位の高い要求から最新のプロジェクト状況を考慮して，見積もり数に則り，イテレーション内で開発を終えることができる量を TODO リストに移動する．その際に，TODO リストにある要求をイテレーション内で開発を終えてリリースすることを，チームメンバーで合意する．

次に DOING リストには，イテレーション内で開発を終える予定で開発着手中の要求が並んでいる．開発メンバーは TODO リストから優先順位や開発メンバーのスキル等を鑑みて開発に着する要求を選定し DOING リストに移行する．

最後に DONE リストには開発が完了した要求が並んでいる．イテレーション内で順次 DOING リストにある要求の開発が終わり，要求管理者のレビューやテストを経てリリースが完了すると，要求は DONE リストに移行される．イテレーション終了時には，TODO にあった要求がすべて DONE に移行されている状態が理想の状態で何も問題なく開発が進捗した証である．要求管理者はイテレーション開始時に，最新の状況を鑑みてイテレーション内で開発すべき要求の選定を行い，チームメンバーと合意する．チームメンバーはイテレーション内で開発すべき要求を分担し開発に取りかかる．イテレーションの最中に要求管理者は新たな要望の獲得や，既存の要望やアイディアの整理，詳細仕様や受け入れ基準を含め

たユーザーストーリーの策定，要求変更に準じた既存要求の変更，要求の優先順位の更新を行い，次のイテレーションに向けて開発すべき要求の整理を行う．このような要求管理フローをイテレーション毎に繰り返すことで，要求の変更や追加に適応的に対処し，ソフトウェア開発が円滑に進むよう務める．

昨今のアジャイル開発では上記のようなプロダクトバックログを用いた要求管理をタスク管理ツールや専用のツールを用いて行うことが多い．そのようなツールの上で行われている要求管理では各種要求管理上の履歴をログデータとして収集しやすく，かつ定量的に分析できるのではないかと考え，要求管理フローにおける要求管理履歴に着目しデータを収集することとした．

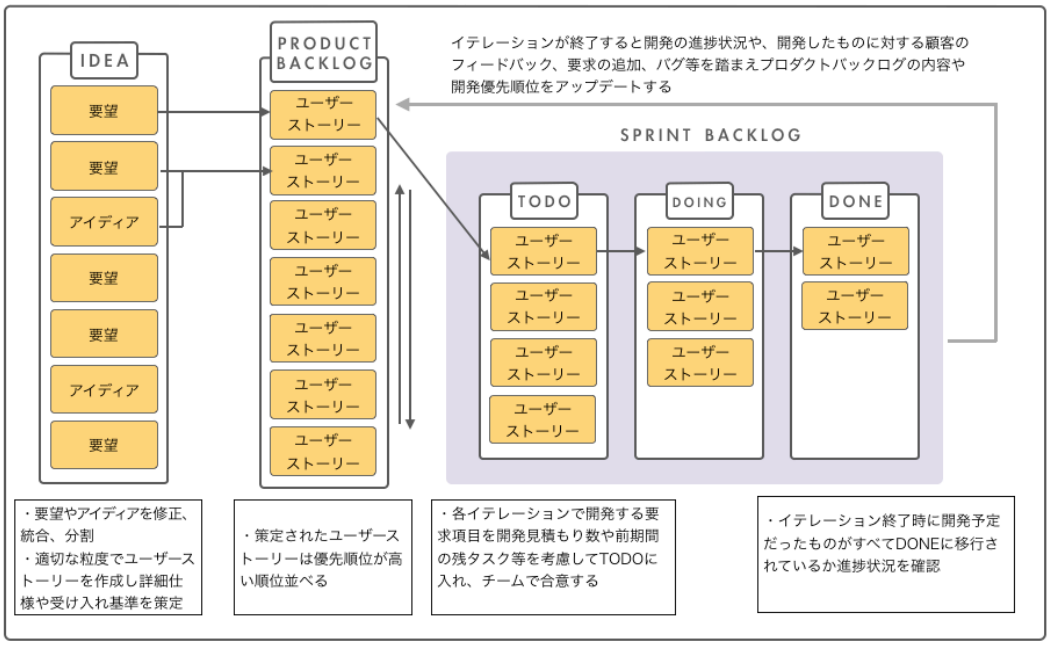


図 2.4 アジャイル開発におけるプロダクトバックログを用いた要求管理フロー

2.3. IT プロジェクトの成否要因に関する先行研究

要求仕様の曖昧さや要求管理の煩雑さが，プロジェクトや開発タスクの納期遅延や品質低下を引き起こすことはこれまでに様々な研究によって指摘されている [5] [15] が，これらの研究では要求仕様の曖昧さや要求管理の煩雑さがなぜ起きるのか，また要求管理上のどのような要因の影響度が大きいのかは明らかにされていない．さらにアンケートによる分析が多く，アンケート作成者の能力や，アンケート内容に研究結果が大きく依存してしまっている問題も指摘されている．このような指摘から，アンケート以外の定量的な評価手法や測定項目を用いた分析が必要であると考えた．

また高橋らの研究では，プロジェクト失敗の一因である納期遅延につながる要因の分析を行っている [16] が，開発工程のみを対象としており，その上流工程である要求管理工程は分析対象から除外されてしまっており，上記の要求仕様の曖昧さや要求管理の複雑さと納期遅延の関係性は明らかにされていない．また大宮らによる納期遅延が保守工程に及ぼす影響を調査した研究では，納期遅延が発生したプロジェクトの保守工程の問題はそれ以前の工程の影響を受けており，上流工程に遡って工程横断で納期遅延を発生させない根本的な研究が必要であることを指摘している [17].

2.4. 先行研究を踏まえた問題意識

アジャイル開発において要求管理者は大きな役割と責任を持っており，要望やアイディアの獲得，ユーザーストーリーによる要求定義，要求を満たす詳細仕様の策定・受け入れ基準の設計，要求の開発優先順位決定，要求追加や変更に準じた要求内容の更新といった業務をプロジェクト進捗状況に応じて適応的に行わなければいけない．しかしこのような要求管理を曖昧に行ったり，煩雑に行ったりすると納期遅延などのプロジェクト失敗をもたらすことが先行研究によって明らかにされている．細かい反復を繰り返すアジャイル開発にとって，要求の開発遅延は次の反復期間に残タスクとして大きな影響をもたらし，それが悪循環となってプロジェクト全体が破綻してしまう．しかし，アジャイル開発において，どの

ような要求管理が開発遅延を引き起こすのか，要求管理と開発遅延の関係性は明らかにされていない．また別の先行研究が示すように，評価手法に依存するような評価項目ではない定量的なアプローチでの分析で，要求管理工程と開発遅延を分析する必要がある．これらの制約を考慮した上でどのような要求管理を行うと開発遅延につながるのか，要求管理工程において開発遅延につながる要因はなにかを工程横断的に分析する必要があると考えた．

第 3 章
研究アプローチと分析手法の提案

本章では，前章での問題提起を踏まえて本研究の目的を達成するための研究アプローチと分析手法について述べる．

3.1. 研究のアプローチと手順

本研究では1.2節で述べた研究目的を達成するために下記の手順で研究を進める．

1. 分析対象選定とデータ収集
2. 計測データに対する前処理・予備分析
3. 開発遅延予測モデルの構築と分析
4. 分析結果の評価と考察

まず分析対象となるプロジェクトを選定すること及びプロジェクトにおける要求管理状況を調査し，定量的な観点であること，測定可能なこと，開発遅延に影響を及ぼす可能性があることなどを考慮して測定するデータ項目とデータ収集手段を決定し，データの収集を行う．次に計測したデータに対して前処理を行い，データの信頼性を高め，予備分析として各測定項目の蛍光や基本統計量を明らかにし，開発遅延の有無に寄与すると思われる要因の当たりをつける．予備分析の結果を踏まえて，統計モデルを用いて開発遅延予測モデルを構築し，モデルの評価と改善，各要因の評価を行う．最後にモデルを用いた分析結果に対する評価と考察を行う．

3.2. 分析手法の提案

本研究では分析対象とするソフトウェア開発プロジェクトの要求管理履歴の計測データと各要求の開発遅延の有無を統計的にモデル化することで，要求管理工程における各要因と開発遅延の有無の関係性を評価する．具体的な手法としては，各要求の成熟過程における計測データを説明変数，各要求の開発遅延の有無を目的変数とした統計モデルを構築する．今回の分析対象において，目的変数は 2 値をとるためロジスティックモデルを用いてモデルを構築し，評価回帰方程式の各変数に計測データを評価値として用いてロジスティック回帰分析を行い，各評価項目の影響度を求める．また検定や予測推定精度を用いたモデルの評価を行う．

3.3. 計測データの収集と前処理

3.3.1 変数として計測するデータ

実際にアジャイル開発で自社事業における web サービス開発を行っているプロジェクトを対象としてデータの収集を行う．アジャイル開発プロジェクト全体で使われている複数のツールから，要求単位毎に各種ログデータを収集することで工程横断的に各要求項目に対する量的な計測データが収集できる．今回対象とするツールは，要求管理ツール，オンラインコミュニケーションツール，ドキュメント共有ツール，仕様調整ツールであるデザインツールの 4 つである．各ツールに残るログデータから，要求が獲得されて成熟していく過程及び開発工程に渡ったあとの計測データ及び開発遅延の有無が計測できる．本研究においては，要求項目に対して量的データで計測が可能であるという観点と各種ツールにおける制約を考慮して表 3.1 の項目のデータを分析に用いる変数として計測する．

表 3.1 計測する変数の説明

名称	分類	定義
開発遅延の有無	質的	要求を予定通りに開発することができたか否か
要求カテゴリー	質的	ユーザーストーリー単位でのカテゴリ分類
要求調整回数	量的	獲得した要求が，要求仕様として定義されるまでの間に更新された回数
要求成熟日数	量的	要求獲得から要求定義完了までかかった日数
開発待機日数	量的	要求定義日から開発開始日までの日数
ステークホルダー数	量的	要求定義に関わった人数

3.3.2 データに対する前処理

3.3.1 で述べたデータに対して下記の 3 つの前処理を行う．

1. 欠損値があるデータの除外
2. 要求管理工程外の要因によって開発遅延となったデータの除外
3. 質的データに対するダミー変数の設定

できるだけ信頼性の高いモデルを構築するために，ツールの不具合等で欠損がある計測データを取り除く．また本研究は要求管理工程における要求管理履歴と，開発工程における開発遅延の関係性をモデル化すること及びその要因を明らかにすることが目的であるので，要求管理工程外にある要因によって開発遅延が引き起こされた計測データも可能な限り分析対象から除外する．例えば，開発担当者の病欠による開発遅延や，社内都合によって急に別プロジェクトのタスクをアサインされたために，大幅なリソース削減を求められた事によって発生した遅延などが該当する．これらの要因によって発生した遅延を含むデータを分析対象から除外することで，より信頼性の高い分析を行う．

3.4. 予備分析

前処理を用いて信頼性を低下させる計測データを除外した上で，各計測データに対して予備分析を行う．各計測項目の基本統計量や，各計測データの開発遅延の有無による差のグラフ化といった手段を用いて，モデル構築の際に大きく寄与するであろう変数を特定する．

3.5. モデルの構築と評価方法

予備分析の結果を踏まえて，各要求の成熟過程における計測データを説明変数，各要求の開発遅延の有無を目的変数として統計モデルを構築する．今回の分析対象において，目的変数は 2 値をとるため二項分布に従う質的変数であると仮定す

る．目的変数が質的変数で，説明変数の目的変数に対する貢献度を明らかにする統計解析手法としてロジスティック回帰モデルを用いる．ロジスティック回帰モデルを用いて統計モデル式を構築し，モデルの適合度・信頼性を評価する．またモデル式のそれぞれの変数に計測データを評価値として当てはめることでロジスティック回帰分析を行い，目的変数に対する各説明変数の貢献度を求める．モデルは下記の (3.1) の式で表される．

$$\log \frac{p}{1-p} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n \tag{3.1}$$

P : 目的変数が 1 となる確率,　b_0 : 定数項,　b_n : 偏回帰係数,　x_n : 説明変数

モデル式は目的変数が 1 となる確率（成功確率）を p と置いたときの $logit(p) = log(p/(1-p))$ に対する線形回帰で表現される．本研究では目的変数が「開発遅延の有無」で P は開発遅延が起きる確率を表し，x は各説明変数であるデータ収集項目を表す．

3.6. 予測推定精度によるモデルのチューニングと評価

構築できたモデルの分析結果を元に，テストデータに対する予測推定精度を用いてモデルの評価を行う．また予測精度を指標として当てはまりの良いモデルの再構築を行い，予測精度の高いモデルの再構築を行う．

第 4 章
分析対象とデータ収集

4.1. 分析対象の概要

4.1.1 分析対象のプロジェクト

本研究における分析対象プロジェクトは，T 社における自社事業としての Web サービス開発プロジェクトである．プロジェクト概要を表 4.1 に示す．プロジェクトは 2018 年に始まり 2019 年 12 月時点で進行中のプロジェクトである．自社事業としてソフトウェア開発を行っており，明確なプロジェクト終了日を持たず，事業撤退の判断がなされない限りプロジェクトが終了する予定はない．プロジェクトで開発しているソフトウェアは個人向け web サービス及びそれの管理システムである．開発規模は業務委託等の契約形態のメンバーを含めると 20〜30 人規模となっており，アジャイルソフトウェア開発手法としてスクラムを用いている．イテレーション期間は原則 1 週間を採用しており，大型連休等の変則的なタイミング以外は基本的に 1 週間単位となっている．要求管理者はエンジニアリングに知見のあるプロダクトオーナーで，1 人で要求の管理を行っている．要求記述手法としてユーザーストーリー，要求管理手法としてプロダクトバックログを用いた要求管理を行っており，要求管理ツールとして Trello[1]を利用している．尚，本プロジェクトにおいては，ユーザーストーリーを粒度別の階層に分けての運用は行っていない．その他サブツールとして，Google Sheets や Docbase[2]といったドキュメントツールも併用しており，要求項目によっては詳細仕様をドキュメントツー

1　ATLASSIAN 社が提供する視覚的にタスク管理が可能なカンバン方式のタスク管理ツール [18]

2　株式会社クレイが提供するドキュメント共有ツール [19]

ルに記入し，ドキュメントツールを Trello とリンクさせて運用しているパターンもある．また，仕様調整・デザイン確認は，オンライン・オフラインそれぞれで行われており，オンライン上ではデザインツールやチャットツールで，オフラインとしては定例の会議内で行われている．要求獲得方法としては，ステークホルダーへのヒアリング，チームの定例会議，ユーザーからのフィードバックやバグ報告，チームメンバーの Trello への起票，ステークホルダーからの依頼等が主な手段である．

表 4.1 分析対象プロジェクトの概要

項目	内容
サービス概要	個人向け web サービス
プロジェクト期間	2018 年 9 月-2019 年 12 月進行中
開発規模	20-30 人規模（内プロダクトチーム約 14 名)
アジャイル開発手法	スクラム
イテレーション期間	基本的に 1 週間（一部変則あり）
ユーザー数	月間 10-15 万 UU/50 万 SS[3]
要求管理者	プロダクトオーナー（エンジニア経験あり）
要求管理者数	1 人
要求管理手法	プロダクトバックログ
要求記述形式	ユーザーストーリー
要求管理ツール	メインツール：Trello サブツール：Google Sheets,Zeplin,Docbase
その他関連ツール	Slack,Github,screenfull,XD
要求獲得方法	ヒアリング, 会議,Trello への起票, ユーザーからのフィードバック, ステークホルダーからの依頼, バグ報告

4.1.2 プロジェクトの期間

対象プロジェクトは自社 IT 事業を行う部署設立と共に動き出したプロジェクトであり，2019 年 12 月時点でプロジェクトを開始して 1 年 5ヶ月である．

4.2. 計測データ収集

4.2.1 データ収集期間

データ収集期間は 2019 年 4 月 1 日〜12 月までの約 9ヶ月間である．

4.2.2 評価に用いる変数とその定義

本研究において，要求項目に対して量的データで計測が可能であるという観点とプロジェクトや各種ツールにおける制約を考慮して表 4.2 の項目を分析に用いる変数として計測した．

表 4.2 評価に用いる変数群

名称	分類	定義
開発遅延の有無 (CH)	質的	要求を予定通りに開発することができたか否か
要求カテゴリー (CA)	質的	ユーザーストーリー単位でのカテゴリ分類
要求調整回数 (RCN)	量的	獲得した要求が，要求仕様として定義されるまでの間に更新された回数
要求成熟日数 (RCD)	量的	要求獲得から要求定義完了までの経過日数
開発待機日数 (RDD)	量的	要求定義日から開発開始日までの経過日数
ステークホルダー数 (SHN)	量的	要求定義に関わった人数

3 UU = ユニークユーザー数, SS = セッション数 [20]

要求獲得からその要求の開発が完了する一連の流れの中で，表 4.2 で示した計測項目がどのように計測されるかを図 4.1 に示す．赤字で表記されているものが計測項目である．時系列に沿って説明すると，まず要求を獲得した日から要件定義を完了するまでの経過日数を要求成熟日数として計測する．これは，2.2.4 で説明したプロダクトバックログを用いた要求管理における，IDEA リストに要望が入った日から，仕様が策定され，BACKLOG リストに移行されるまでの日数である．またこの期間において，要件定義に関わった人数をステークホルダー数，要求がどのカテゴリーに分類されるかを要求カテゴリー，要件を定義するまでに要求内容を更新した回数を要求調整回数として計測する．次に，要件定義完了日から開発が開始されるまでの経過日数を開発待機日数として計測する．これはプロダクトバックログにおける，BACKLOG リストに入った日から，TODO リストに移行されるまでの日数である．最後に，開発完了予定日に開発完了しているか否かを，開発遅延の有無として計測する．プロダクトバックログにおいて，イテレーション終了時に TODO リストに入っていた要求が DONE リストに移行されているかを計測している．

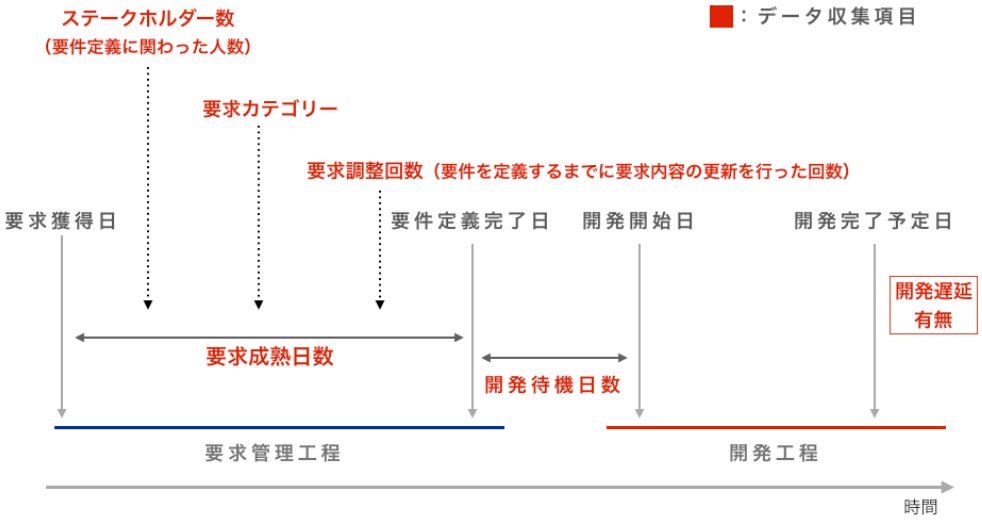


図 4.1 要求のライフタイムサイクルとデータ収集項目のマッピング

要求カテゴリーは表 4.3 に従って分類した．2.2.2 で，PRINCE モデルという要求種別分類手法が先行研究で用いられていることを説明したが，PRINCE モデルは開発項目に対する分類手法で粒度の細かい分類手法である．そのためユーザーストーリーのような要求粒度の分類には適しておらず，本研究では独自の分類手法を用いて要求を分類した．

表 4.3 要求カテゴリーの分類

カテゴリー名	定義
Business Requests	ビジネスサイドの要求対応に起因する項目
New Functions	新機能開発に関する項目
Bugfix	バグ対応に関する項目
Technical Depts	技術的負債に関する項目
Renewal Tasks	サービスの新規リニューアルに関する項目
Others	その他

4.2.3 データ収集の手段

要求管理のメインツールである Trello のログデータからユーザーストーリー単位で要求管理工程における管理履歴データを収集した．Trello Webhook API を用いて Trello 上での各要求に対する処理（追加，更新，削除，カードの移動等）をトリガーとして，Google App Scripts で作成された API に処理の内容を HTTP 通信を用いて POST することで，Google Sheets に各要求の管理履歴ログデータを収集した．収集されたログデータを，Google App Scripts を用いてユーザーストーリー単位で集計することで各要求に対する要求管理履歴をデータとして集計した．またメインツールである Trello では計測できない部分の要求管理履歴を，ドキュメントツールに記録されている会議の議事録や，Zeplin 上のコメント機能の履歴から収集し，ユーザーストーリー単位で集計した．

4.2.4 データ収集と前処理

データの計測期間である 2019 年 4 月 1 日〜12 月の間で 251 件のユーザーストーリーを収集した．そこからなんらかの欠損があるデータを取り除き，216 件のデータとし，さらに要求管理工程外の要因によって開発遅延となったデータを除外し 205 件のデータを分析対象とした．そして表 4.2 にある変数のうち，開発遅延の有無を目的変数，それ以外を説明変数とおいたときに，質的変数となる開発遅延の有無，要求カテゴリーの 2 つに対してダミー変数を設定した．

第 5 章
収集データに対する予備分析

本章では前章によって収集し，前処理が行われた分析対象データに対して，開発遅延に寄与しうる要因の推定，統計モデルを構築するための各データの傾向の把握や，分布の違いを明らかにするため予備分析を行う．

5.1. 分析方法と手順

本予備分析では，下記の手順に従って分析を行う．分析手段として統計処理に強いプログラミング言語である R 及び Python を用いる．

1. 計測データの開発遅延の有無の件数の把握
2. 量的データの開発遅延の有無による傾向の分析
3. 要求カテゴリー毎の要求件数の把握
4. 分析結果の考察

まず，開発遅延の有無の割合から全体の傾向を把握する．次に，量的変数群に対して，開発遅延の有無でヒストグラムを作成し，開発遅延の有無によってデータの傾向に差があるかを分析する．次に質的データである要求カテゴリー毎の要求件数を把握する．最後にこれらの予備分析結果の考察を行う．

5.2. 計測データの開発遅延の有無の件数

本研究で分析対象とする 205 件のデータから，開発遅延の有無の各要求件数を把握するためグラフを作成した．作成したグラフを図 5.1 に示す．

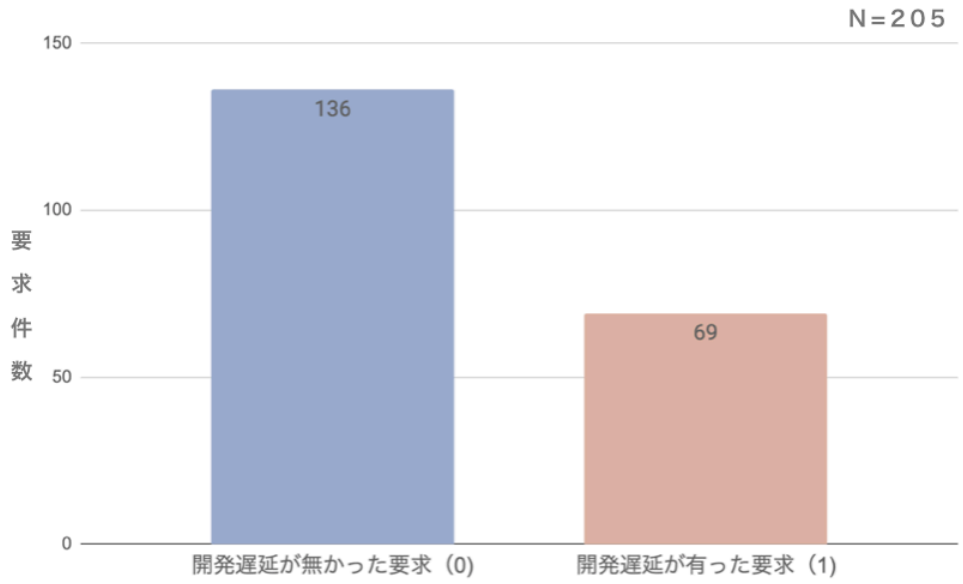


図 5.1　開発遅延の有無の各要求件数

205 件の分析対象データの内，開発遅延がなく無事開発できた要求は 136 件で全体の 3 分の 2，開発遅延があった要求は 69 件で全体の 3 分の 1 であることがわかる．

5.3. 量的データの開発遅延の有無による傾向の分析

次に分析対象データの説明変数にあたる変数の内，量的変数に属するステークホルダー数，要求調整回数，開発待機日数，要求成熟日数の 4 変数に対して，開発遅延の有無で分類してヒストグラムを作成し，それぞれの変数が開発遅延の有無によってデータの傾向に差があるかを分析した．作成したヒストグラムを図 5.2〜図 5.5 に示す．

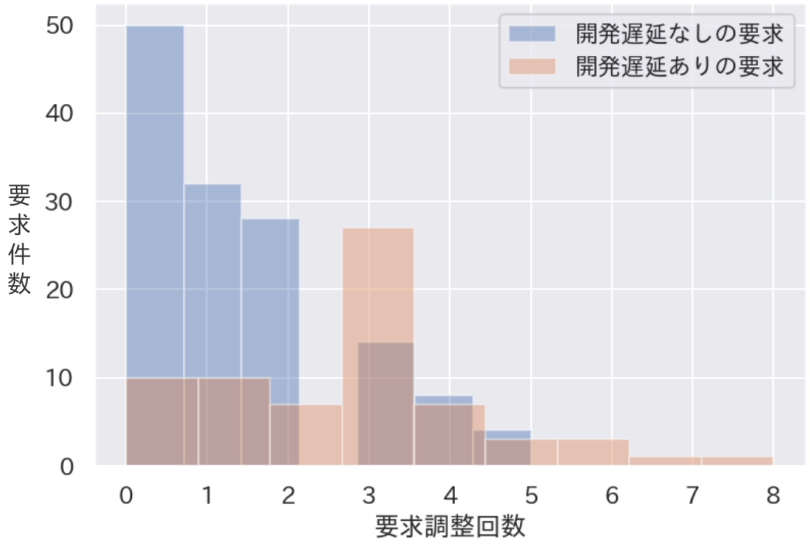


図 5.2　開発遅延の有無によるヒストグラム（要求調整回数）

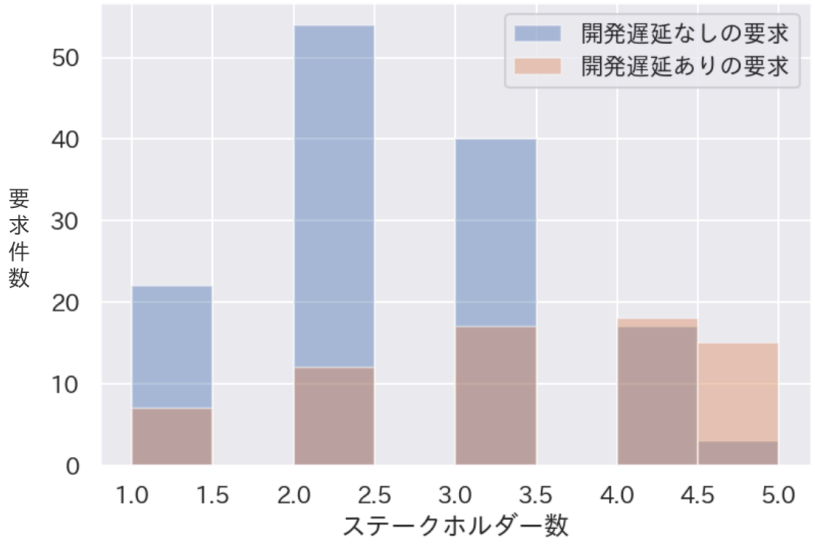


図 5.3　開発遅延の有無によるヒストグラム（ステークホルダー数）

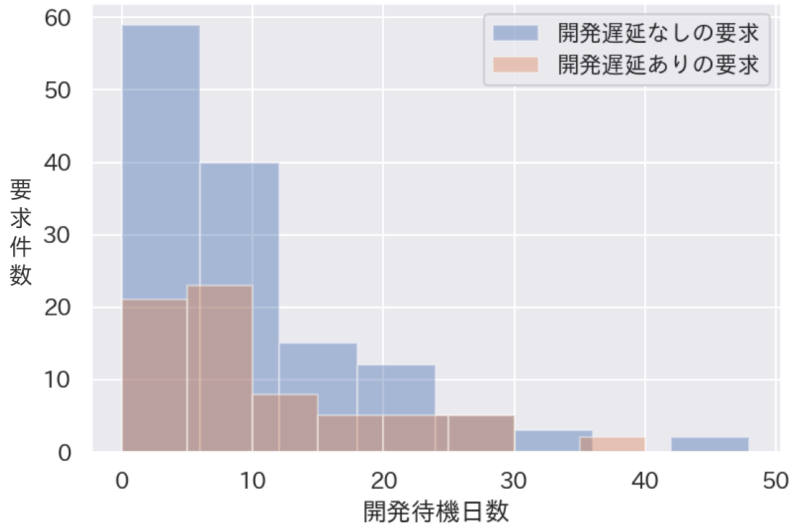


図 5.4　開発遅延の有無によるヒストグラム（開発待機日数）

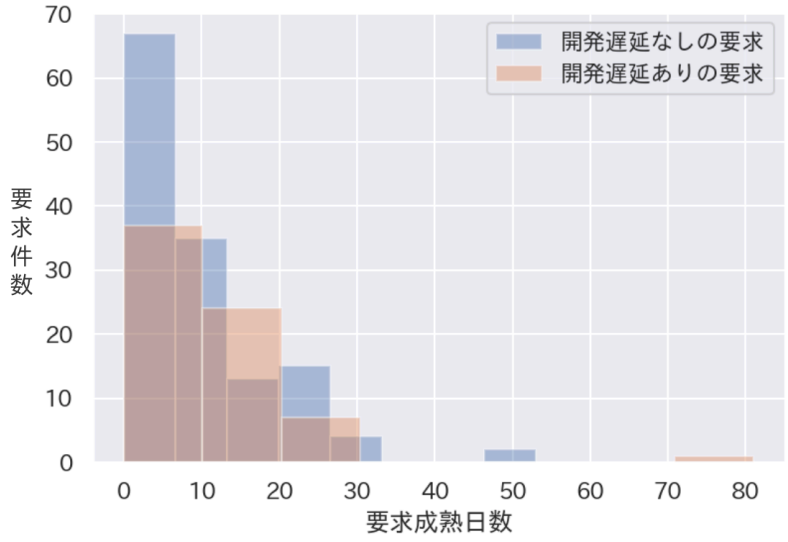


図 5.5　開発遅延の有無によるヒストグラム（要求成熟日数）

要求調整回数のヒストグラムである図 5.2，ステークホルダー数のヒストグラムである図 5.3 を見ると，開発遅延の有無によってヒストグラムに差があるように見受けられる．これは，この 2 要因が開発遅延の有無に影響する可能性を示唆している．一方で，開発待機日数のヒストグラムである図 5.4，要求成熟日数のヒストグラムである図 5.5 を見ると，開発遅延の有無で大きな差はないように見える．よって開発遅延予測モデル構築の際に要求調整回数，ステークホルダー数に着目してモデルの構築を行うべきだと考えた．

5.4. 要求カテゴリー毎の要求件数の把握

次に要求カテゴリー毎の要求件数の把握を行うためにグラフを作成した．作成したグラフを図 5.6 に示す．

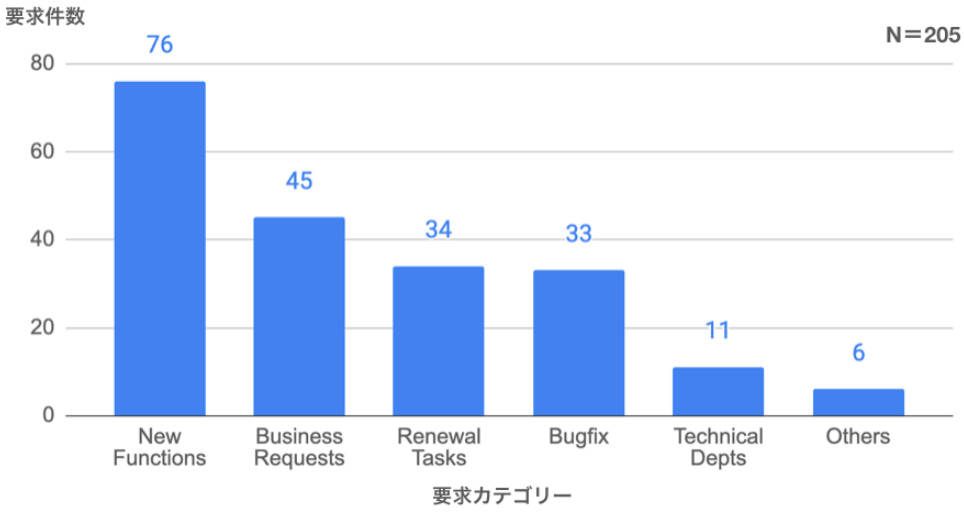


図 5.6　要求カテゴリー毎の要求件数

要求カテゴリー毎に大きく件数が違うことがグラフから読み取れる．特に New Functions と Others では 10 倍以上の件数の差があることがわかる．これだけ件数にばらつきがあると Renewal Tasks，Bugfix，Technical Depts，Others といった件数が少ない要求カテゴリーではサンプルサイズの小ささにより，統計的に有意

な分析結果が得られないのではないかといったことが推察できた．

5.5. 分析結果の考察

予備分析を経てデータ全体の傾向として，開発遅延無しと開発遅延ありの件数が 2 : 1 の割合であることが理解できた．開発遅延なしの件数が多いため，開発遅延予測モデルの構築やその後の予測推定精度によるモデルの評価・チューニングを行う際に，開発遅延なしのデータを過剰評価しすぎないよう注意して分析を進めるべきだと考えられる．また量的データの開発遅延の有無によるヒストグラムから，要求調整回数とステークホルダー数の 2 要因が開発遅延に影響を及ぼしているのではないかということが示唆されたため，2 要因に着目して予測モデルの構築を進めるべきだと考えた．要求カテゴリーに関しては，カテゴリー毎に件数が大幅に違うため，件数が特に小さい，Renewal Tasks，Bugfix，Technical Depts，Others といった要求カテゴリーでは統計的に有意な分析結果を得にくいのではないかということが推察された．これらの分析結果を踏まえて，開発遅延予測モデルの構築を進める．

第 6 章
モデルの構築による分析結果と考察

本章では，前章の予備分析の結果を踏まえて開発遅延予測モデルを構築する．構築したモデルに対して多重共線性の確認，モデルの適合度の検定によって評価を行い考察する．またモデルを用いたロジスティック回帰分析を行い，各要因と開発遅延の関係性を考察する．そしてモデルの予測推定精度による評価を行う．今回計測したデータを訓練データとテストデータに分割し訓練データのみでモデルを再構築し，テストデータに対する正解率を評価指標として予測推定精度の評価を行う．モデルの構築・分析には R と Python を利用する．

6.1. 開発遅延予測モデル

はじめに収集したデータを用いて開発遅延予測モデルの構築を行い，モデルの評価と考察を行う．

6.1.1 本研究における開発遅延予測モデルの構築

4 章で説明した計測項目を各変数として設定し，3 章で提示したロジスティック回帰モデルを用いて本分析で使用するモデルを構築する．3 章で提示したロジスティック回帰モデルをあらためて (6.1) 式に示す．この回帰式に下記の計測項目をそれぞれ目的変数，説明変数として設定する．

- 目的変数 : 開発遅延の有無 (CH) … 有=1, 無=0
- 説明変数 : 要求カテゴリー (CA), 要求調整回数 (RCN), 要求成熟日数 (RCD), 開発待機日数 (RDD), ステークホルダー数 (SHN)

$$\log \frac{p}{1-p} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n \tag{6.1}$$

P : 目的変数が 1 となる確率, b_0 : 定数項, b_n : 偏回帰係数, x_n : 説明変数

(6.1) 式に目的変数である開発遅延の有無 (CH) が起きる確率を P, 各説明変数を $X_n(n = 1..5)$ とおくと (6.2) 式となる.

$$\log \frac{p}{1-p} = b_0 + b_{1i} CA_i + b_{2i} RCN_i + b_{3i} RCD_i + b_{4i} RDD_i + b_{5i} SHN_i \tag{6.2}$$

(6.2) 式のロジスティック回帰モデルを収集したデータを元に構築した. (6.2) の回帰式によって導かれた開発遅延予測モデルを (6.3) 式に示す. 尚, 要求カテゴリーは Bugfix カテゴリーを説明変数の参照カテゴリーとして設定し, (6.3) 式では Business Requests の偏回帰係数を表記している.

$$\begin{aligned}\log \frac{p}{1-p} = {} & -2.7844 + (0.6882 \times CA_i) + (0.3931 \times RCN_i) + (-0.0136 \times RCD_i) \\ & + (-0.0020 \times RDD_i) + (0.4119 \times SHN_i)\end{aligned} \tag{6.3}$$

これによって (6.3) 式の各説明変数に計測データを代入すると, 要求の開発遅延が起きる確率を予測する開発遅延予測モデルが構築できた.

6.1.2 VIF を用いた多重共線性の評価

次に構築できた開発遅延予測モデルの多重共線性の評価を行った. 多重共線性とは, モデルに用いた説明変数同士が相関関係にあるときに起こる状態のことを指す. 多重共線性があると, 分析結果の係数の標準誤差や決定係数が大きな値となってしまい, 信頼性の低いモデルとなってしまう. その多重共線性を評価するため, VIF という指標を用いる. VIF とは分散拡大係数のことを指しており, ある特定の説明変数が, その他の説明変数によってどれほど説明されているかという相関度合いを表しており, この VIF の値が高いと多重共線性の疑われる. 一般

的には VIF が 10 を超えると多重共線性を疑う．VIF を用いた多重共線性の評価結果を表 6.1 にまとめた．今回はすべての説明変数の GVIF 値が 2 以下となっており，多重共線性はないと判断できた．

表 6.1 VIF を用いた多重共線性の評価

説明変数	GVIF	DF:自由度
ステークホルダー数	1.303635	1
要求調整回数	1.718577	1
要求成熟日数	1.358229	1
要求待機日数	1.059031	1
要求カテゴリー	1.129179	5

6.1.3 Hosmer-Lemeshow 検定を用いたモデル適合度の評価

次にモデル全体の適合度を Hosmer-Lemeshow 検定を用いて評価した．Hosmer-Lemeshow 検定はロジスティック回帰モデル全体の有意性（適合度）を検定する手法である．現在のモデルの当てはまりが正しいという帰無仮説の元，予測値と観測値の適合度の評価を行う．Hosmer-Lemeshow 検定の検定結果を表 6.2 に示す．

表 6.2 Hosmer-Lemeshow 検定の検定結果

X-squared	df	p-value
13.81	8	0.08684

p 値は 0.08684 となっており，5%の棄却域に入っていない．Hosmer-Lemeshow 検定は現在のモデルの当てはまりは正しい（適合度が良い）という帰無仮説の元検定を行っているため，帰無仮説は棄却されず，現在当てはめているモデルは正しく適合度は高いという結果が得られた．

6.1.4 開発遅延予測モデルに対する考察

今回構築した開発遅延予測モデルは，多重共線性も無いと判断でき，Hosmer-Lemeshow 検定によっても適合度は高いと判断できた．そのため今回の分析に用いた説明変数である各要因と開発遅延の有無の関係性を表すモデルとして一定の評価があると考えられる．ただ今回のモデルには定量的に評価しにくい変数である各要求の仕様量といった要因を考慮できておらず，それらの要因を変数として組み込むことでより高度なモデル化ができるのではないかと考えられる．また目的変数を開発遅延のみに絞って分析を行ったが，他の目的変数と要求管理履歴の関係性も同じようにモデル化できるのではないかと考えている．プロジェクトの良し悪しを評価する QCD の観点であるソフトウェアの機能品質やコストと要求管理の関係性をモデル化することができ，それらのモデルを複合的にモデル化することで，プロジェクトの成否と要求管理の関係性をより包括的なモデルにより明らかにすることができるのではないかと考察する．

また今回の分析ではプロダクトバックログにおける DONE リストに，要求が開発完了予定通りに移行されたかどうかを判断基準として開発遅延の有無を計測しており，開発遅延の大きさは考慮できていない．開発遅延量も測定項目としてデータ収集を行い，その遅延がもたらす影響を前述の QCD におけるコスト，品質の観点から評価することを考慮したモデルを構築することにより，影響度の高い開発遅延につながる要求管理上の要因を特定できるのではないかと考えられ，プロジェクトに大きな影響を与える開発遅延を防ぐといった対策機構の考案につながるのではないかと考える．

6.2. ロジスティック回帰分析によるモデルの分析

次に構築したモデルを用いてロジスティック回帰分析を行い，各要因の統計的な評価を行った．要因毎の影響度の大きさ及び要求カテゴリーでの差を分析結果をもとに評価した．そして明らかになった要因が開発遅延を引き起こす理由，要求管理工程でその要因を上手く制御する対策について考察する．

6.2.1 ロジスティック回帰分析による分析結果

6.1 で示したモデルを参考にロジスティック回帰分析を行って，モデルの完成度と開発遅延につながる要因を評価する．ロジスティック回帰分析は統計ソフト R を用いて行った．その結果を表 6.3 に示す．

表 6.3 ロジスティック回帰分析による分析結果

	Estimate	Std. Error	z value	P>|z|
(Intercept)	-2.7844	0.626	-4.447	0.000
要求カテゴリー Business Requests	0.6882	0.585	1.177	0.239
要求カテゴリー New Functions	0.5045	0.548	0.922	0.357
要求カテゴリー Others	-0.4959	1.218	-0.407	0.684
要求カテゴリー Renewal Tasks	-0.1784	0.662	-0.269	0.788
要求カテゴリー Technical Depts	0.0865	0.901	0.096	0.924
要求成熟日数	-0.0136	0.018	-0.761	0.447
要求調整回数	0.3931	0.138	2.855	0.004
ステークホルダー数	-0.4119	0.167	2.472	0.013
開発待機日数	-0.0020	0.019	-0.103	0.918

まず各変数の p 値を確認し，統計的に有意である要因を特定した．その結果，ステークホルダー数が有意水準 5%（p=0.013）で統計的有意，要求調整回数が有意水準 1%（p=0.004）で統計的有意であった．その他の説明変数はどれも p 値が 0.2 を超えており，統計的に有意な結果は得られなかった．よって要求カテゴリー毎に開発遅延に及ぼす影響度合いの差は不明であった．また赤池情報量基準（AIC）によるモデル適合度は 240.66 であった．

次に各要因の影響度合いを調べるために各要因の対数オッズを指数変換してオッズ比を算出した結果が表6.4である．この表からは，要求カテゴリーBusiness Requestsと要求カテゴリーNew Functions，ステークホルダー数，要求調整回数の開発遅延に対する影響度合いが大きいことが読み取れる．しかし，要求カテゴリーBusiness Requestsと要求カテゴリーNew Functionsの2つに関しては統計的に有意な要因ではないため，継続した調査が必要である．これらの分析結果から，本プロジェクトにおいては下記2点の統計的に有意な要因とその影響度合いを明らかにできた．

1. 要求調整回数が1回増えるとき，開発遅延が起きる確率を1.48倍引き上げる．
2. ステークホルダー数が1人増えるとき，開発遅延が起きる確率を1.50倍引き上げる．

表 6.4　説明変数のオッズ比

説明変数	オッズ比
要求カテゴリー Business Requests	1.99
要求カテゴリー New Functions	1.656
要求カテゴリー Others	0.609
要求カテゴリー Renewal Tasks	0.836
要求カテゴリー Technical Depts	1.09
要求成熟日数	0.986
開発待機日数	0.998
要求調整回数	1.481
ステークホルダー数	1.509

6.2.2 開発遅延につながる要因の考察

まずステークホルダー数，要求調整回数という 2 つの開発遅延につながる要因を特定できたこと，その影響度合いを係数によって示せたことは，ソフトウェア開発に関わる人間が実現場で感じている違和感を定量的な評価方法によって明らかにできたこととなり，一定の評価が得られるのではないかと考えている．

しかし，今回明らかにできた要因と係数は，分析対象プロジェクトにおいての結果であり，すべての IT プロジェクトに共通してそのような傾向があるとは言えない．だが 2 章の先行研究で示されているように，プロジェクト終了後のアンケート調査では信頼性のある分析ができないという指摘に対して，本研究では，進行中のプロジェクトにおいても適用可能であること，汎用的かつ定量的な計測項目で評価が可能であること，の 2 点を満たす分析手法を提案できた．この点に関しては評価ができるのではないかと考える．この分析手法を元に，多くのプロジェクトを分析していくことで，より一般化された開発遅延につながる要因と係数を明らかにしていくことができるのではないか．

さらに今回の分析結果である要求調整回数とステークホルダー数の 2 要因の増加の背景にある要求管理工程上の要因を考察した．それはユーザーストーリーの粒度と，要求管理者の要求カテゴリー毎の得手不得手の 2 つではないかと考えた．

まずユーザーストーリーの粒度という観点で今回の分析結果を考察すると，ステークホルダー数と要求調整回数という 2 つの要因が開発遅延に影響を及ぼすことは，それに該当する要求の単位であるユーザーストーリーの粒度が大きいことが示唆できるのではないかと考えられる．1 つの要求を定義するために，多くのステークホルダーを巻き込まなれけばいけないということは，多面的な要望をひとまとまりの要求にしてしまったためではないか，と考えられる．要求調整回数に関しては，調整を行う回数が多いほど，要求で定義しなければいけない範囲が広いといった関係性があるのではないか．

要求管理者の要求カテゴリーに対する得手不得手という観点では，本分析においては統計的に有意な結果には至らなかったが，要求カテゴリー毎に要求管理者によって要求の定義が早い，遅いものがあるのではないか，という点である．本研究の分析対象プロジェクトの要求管理者はエンジニアリングに知見のある要求

管理者であり，技術要素の強い要求に対する要求定義には知見が生きることが推測される．例えば，エンジニアリングに知見のない要求管理者が技術要素の強い要求を定義する際はエンジニアを巻き込んで要求を定義しなければいけないため，要求調整回数もステークホルダー数も増加するのではないかといったことが考えれる．このようにユーザーストーリーの粒度が大きいこと，要求管理者の苦手な領域の要求であること，といった背景によってステークホルダー数，要求調整回数が増加しているのではないかと考察した．

6.3. 予測推定精度によるモデルの評価

本節では，6.1 で構築した開発遅延予測モデルを元に，テストデータに対する予測推定精度によってモデルの評価･考察を行う．次にステップワイズ法を用いて，当てはまりの良いモデルの再構築を行い，予測推定精度の向上を図る．それらを踏まえて考察を行う．本分析では python 及び python 機械学習ライブラリ scikit-learn を用いて分析を行った．

6.3.1 分析方法と手順

まず本分析では 205 件の要求管理履歴データを訓練データ，テストデータの 2 つに分割し，訓練データを元にモデルを構築し，テストデータに対する予測推定精度によって評価を行う．評価方法としてはクロスバリデーション法を用いてテストデータを 10 個に分割し，10 グループの未知のデータに対する正解率の平均によって評価を行う．尚，今回の分析では計測データの内，80%を訓練データ，20%をテストデータの割合で分割して利用した．分析の手順を以下に示す．

1. 計測データを分割（訓練データ 80%：テストデータ 20%）
2. 訓練データを元にモデルを構築
3. モデルに対してテストデータを用いたクロスバリデーション法による評価
4. ステップワイズ法によって当てはまりの良いモデルを再構築

5. 再構築したモデルをテストデータに対する正解率によって再評価

6. 結果の考察

6.3.2 テストデータに対する予測推定結果

まず計測データを訓練データ，テストデータに分割した上で訓練データを元に開発遅延予測モデルを再構築した．そしてテストデータに対して 10 分割したクロスバリデーション法による正解率の平均を算出した．ここまでのソースコードと算出された平均正解率を図 6.1 に示す．

```
In [53]: #クロスバリデーション法と正解率を用いた予測推定精度の評価

         X_train, X_test, Y_train, Y_test = train_test_split(selected_X, data_final[Y], test_size = 0.2, random_state = 0)

In [56]: kfold = StratifiedKFold(n_splits=10)
         model_1 = LogisticRegression()
         results = model_selection.cross_val_score(model_1, selected_X, data_final[Y], cv = kfold, scoring = 'accuracy')
         print("10-fold Cross Validation 平均正解率: %.2f" % (results.mean()),"%")

         10-fold Cross Validation 平均正解率: 0.71 %
```

図 6.1　クロスバリデーション法を用いたテストデータに対する平均正解率の算出

最初に構築したモデルでの平均正解率は 71%となった．

6.3.3 ステップワイズ法を用いた当てはまりの良いモデルの再構築

次にステップワイズ法を用いて当てはまりの良いモデルの再構築を行った．ステップワイズ法とは赤池情報基準（AIC) などの基準を元に，各説明変数をモデルに 1 つずつ採択したり取り除いたりしながら，最も当てはまりの良いモデルを求める方法である．今回は AIC の小ささを最も当てはまりの良いモデルの尺度としてモデルの探索を行った．ステップワイズ法によるモデル構築の結果を表 6.5 に示す．

モデル名「1st」がすべての説明変数を採択したモデルであり，最初の予測推定精度の評価に用いたモデルである．そこから AIC が小さくなるよう採択する変数を調整し，一番 AIC が小さい当てはまりの良いモデルとしてモデル名 「4th」が構築された．モデル「4th」で採択されている説明変数は要求調整回数とステーク

表 6.5 ステップワイズ法によるモデルの構築結果

モデル名	AIC	採択された説明変数
1st	240.66	要求カテゴリー + 要求成熟日数 + 要求調整回数 + ステークホルダー数 + 開発待機日数
2nd	234.86	要求成熟日数 + 要求調整回数 + ステークホルダー数 + 開発待機日数
3rd	232.98	要求成熟日数 + 要求調整回数 + ステークホルダー数
4th	231.39	要求調整回数 + ステークホルダー数

ホルダー数の 2 変数である．モデル「4th」のロジスティック回帰分析による分析結果を表 6.6 に示す．

表 6.6 ロジスティック回帰分析による分析結果

	Estimate	Std. Error	z value	P>|z|
(Intercept)	-2.6340	0.461	-5.712	0.000
要求調整回数	0.3564	0.118	3.027	0.002
ステークホルダー数	0.4416	0.164	2.691	0.007

またモデル「4th」によって導き出された回帰方程式を (6.4) 式に示す．

$$\log \frac{p}{1-p} = -2.6340 + (0.3564 \times x_1) + (0.4416 \times x_2) \quad (6.4)$$

P : 目的変数が 1 となる確率，　x_1 : 要求調整回数，　x_2 : ステークホルダー数

この (6.4) 式に示す回帰方程式がステップワイズ法を用いて AIC を基準に当てはまりの良いモデルを構築した結果の回帰式である．

次にモデル「4th」を用いて再度，予測推定精度の評価を行った．手順は 6.3.2 で示す手順と同様である．テストデータに対する平均正解率を用いた予測精度の評価結果を図 6.2 に示す．

```
In [170]: #クロスバリデーション法と正解率を用いた予測推定精度の評価

          X_train, X_test, Y_train, Y_test = train_test_split(selected_X, data_final[Y], test_size = 0.2, random_state = 0)

In [171]: kfold = StratifiedKFold(n_splits=10)
          model_1 = LogisticRegression()
          results = model_selection.cross_val_score(model_1, selected_X, data_final[Y], cv = kfold, scoring = 'accuracy')
          print("10-fold Cross Validation 平均正解率: %.2f" % (results.mean()),"%")

          10-fold Cross Validation 平均正解率: 0.74 %
```

図 6.2　モデル「4th」のテストデータに対する平均正解率

モデル「4th」のテストデータに対する予測推定精度は74%となり，モデル「1st」と比較して 3%の予測推定精度の向上を図ることができた．

6.3.4　予測推定精度による評価の考察

今回構築した開発遅延予測モデルは，クロスバリデーション法を用いたテストデータに対する平均正解率を用いて予測推定精度の評価を行った．さらにステップワイズ法を用いて最適なモデルを探索することで導き出されたモデル「4th」では，74%の予測推定精度となり，3%の予測推定精度の向上を図ることができた．しかし，大幅な精度向上につながるモデルの構築には至らなかった．予備分析にて示したように，本分析対象の計測データの内，3 分の 2 が開発遅延なし，3 分の 1 が開発遅延ありのため，全て開発遅延なしと予測した場合でも予測推定精度は 66%程度になる．そのため，本モデルが予測推定精度という観点で大きな成果を出せたとは言えない．しかし，ステークホルダー数，要求調整回数という定量的で計測しやすい 2 要因のみのシンプルなモデルで 74%の予測推定精度を出せたことは一定の評価が得られるのではないかと考えている．一般的に今回のように目的変数と説明変数を用いて統計モデルを構築する際には，多くの説明変数を用いることでより予測推定精度の高いモデルが構築できる．しかし，それは同時に過学習のリスクを高める要因にもなる．そのため，できるだけシンプルで変数の少ないモデルで性能の向上を図る方が望ましい．今回はシンプルな 2 変数でモデルを構築できたため，その点は評価できると考察する．

また予測推定精度を向上させる方法としては，今回考慮できていない仕様量の大きさ，該当要求の開発難易度といった要因も考えられるが，今回明らかにでき

た 2 要因の詳細な分析による細分化という手段も推定精度向上につながるのではないかと考えている．例えば，ステークホルダーとして要求の定義に関わる人の属性や能力値といった細分化要因として考えられる．要件定義に関わるステークホルダーとして，容易にコミュニケーションが取れるチームメンバーが関わる場合と，経営層や提携企業先の担当者といった普段から密にコミュニケーションができないメンバーが関わる場合とで違いはないのか，またエンジニアが要件定義に関わる場合でもエンジニアリングマネージャーのように技術力もコミュニケーション能力も高いエンジニアと，まだ経験が浅く，コミュニケーションもおぼつかないようなエンジニアが関わる場合で違いはあるのか，といった点である．要求調整回数の細分化でいくと，要求調整回数の調整間隔や調整内容の考慮といった点が考えられる．要求獲得日から要件定義完了までに，5 回の要求調整が行われた時に，等間隔で 5 回行われた場合と，要件定義完了間際に駆け足で 5 回の調整が行われ要件定義が完了した要求で開発遅延に差は出ないのか，また同じ 5 回の要求調整回数でも，一回あたりの更新量が多い要求と，1 回あたりの更新量が小さい要求で差はあるのか．このように今回の 2 要因を様々な観点で分析していくことで，より開発遅延に大きな影響をもたらす要因を特定でき，予測推定精度向上につながると考えている．

6.4. インタビューによる定性評価

最後に実現場で要求管理者として働く人に，今回の分析結果を共有してインタビュー調査を行った．インタビューに協力していただいたのは，今回の分析対象プロジェクトとは別のプロジェクトでプロダクトマネージャーとして働く山田さんである（図 6.3）．利用ツールやプラクティスは違うが，山田さんが所属するプロジェクトにおいてもアジャイル開発を採用してソフトウェア開発を行っている．山田さんはエンジニアリングの知見はなく，ビジネスサイドのバックグラウンドを持つプロダクトマネージャーである．複数回のインタビューを通してビジネスバックグラウンドの要求管理者の意見を収集した．下記にインタビュー時のフィードバック内容をまとめた．

図 6.3 IT 企業勤務 PM 山田真由美さん

- ステークホルダー数，要求調整回数の 2 要因が開発遅延につながることは実現場で働く感覚的にも同意
- 開発遅延が起きる要求は，大きな粒度のまま曖昧な部分を残して開発工程に引き渡してしまっているのではないか
- ステークホルダー数が増えるような要求は要求管理が非常に難しい．私の場合はエンジニアリングに知見がないので，技術要素が強い要求は誰に相談して決めたら良いのかわからず，様々な人を巻き込んで要求定義をしてしまっている．適切な人を巻き込めるような仕組みがあると良い
- 要求獲得に関しては苦手な分野の要求を適切に拾い上げられているか自信がない．そういった部分も気付けるような仕組みがあると嬉しい
- 小・中規模のプロジェクトにおいては要求管理者は 1 人の場合が多く，自身の仕事ぶりを誰かに相談もできず，評価されることもなく，客観的に認識することが難しい．自身の要求管理状況を客観的に可視化し，良くない部分を気づかせてくれる仕組みがあると嬉しい

インタビュー内容からも，今回構築した開発遅延予測モデルに対する納得度合

いに一定の評価が得られた．また，3,4 のフィードバックで示すような，今回の分析結果を踏まえた要求管理を適切に行う仕組みにつながるような提案を頂いた．

第 7 章
結　　論

本論文では，要求管理履歴に基づく開発遅延予測モデルと題して，研究の背景，研究目的，研究方法，予備分析，モデルの構築とその分析結果・考察を論じてきた．本章では研究目的に対する本研究の結論と今後の展望を述べる．

7.1. 結論

本研究ではアジャイル開発における要求管理履歴データに基づいて開発遅延予測モデルを構築し，要求管理工程上の要因と開発遅延の関係性を統計モデルにより評価することを目的として研究を進めてきた．前章での分析結果・考察を踏まえて本研究の結論を述べる．

7.1.1 本研究で構築した開発遅延予測モデル

本研究では要求管理工程履歴から計測できる定量的なデータから，開発遅延予測モデルを構築し評価を行った．計測項目は，下記の 6 項目である．

開発遅延の有無 要求を予定通りに開発することができたか否か

要求カテゴリー ユーザーストーリー単位でのカテゴリー分類

要求調整回数 獲得した要求が，要求定義されるまでに内容が更新された回数

要求成熟日数 要求獲得から要求定義完了までの経過日数

開発待機日数 要求定義日から開発開始日までの経過日数

ステークホルダー数 要求定義に関わった人数

開発遅延の有無を目的変数とおき，その他の項目を説明変数としてモデルの構築を行った．

開発遅延予測モデルの回帰方程式は，p を「開発遅延が起きる確率」，x_1 を「要求カテゴリー」，x_2 を「要求調整回数（回）」，x_3 を「要求成熟日数（日）」，x_4 を「開発待機日数（日）」，x_5 を「ステークホルダー数（人）」とした時に次の (7.1) 式で表すことができる．そして (7.1) 式を元に本研究で収集したデータを用いて開発遅延予測モデルを構築し，(7.2) の回帰方程式を導き出すことができた．

$$\log \frac{p}{1-p} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n \tag{7.1}$$

$$\log \frac{p}{1-p} = -2.7844 + (0.6882 \times x_1) + (0.3931 \times x_2) + (-0.0136 \times x_3) + (-0.0020 \times x_4) + (0.4119 \times x_5) \tag{7.2}$$

構築したモデルを 2 つの方法を用いて評価した．VIF を用いた多重共線性の評価では，多重共線性は確認されず説明変数同士の相関による影響はないことが確認できた．さらに Hosmer-Lemeshow 検定によってモデル全体の適合度は良いと判断できた．

また構築したモデルの予測推定精度を，クロスバリデーション法を用いたテストデータに対する平均正解率によって評価したところ，71%の予測推定精度であった．そこでステップワイズ法を用いて，AIC を基準とした当てはまりの良いモデルの再構築を行った結果，予測推定精度を 3%向上させることができた．再構築後の予測モデルでは，要求調整回数とステークホルダー数の 2 要因のみが説明変数として採択され，シンプルで汎用性の高い要因のみで構築されたモデルとなった．再構築された予測モデルの回帰方程式を，p が「開発遅延が起きる確率」，x_1 が「要求調整回数（回）」，x_2 が「ステークホルダー数（人）」とした時に，次の (7.3) 式で示す．

$$\log \frac{p}{1-p} = -2.6340 + (0.3564 \times x_1) + (0.4416 \times x_2) \tag{7.3}$$

このように要求管理工程履歴で計測できる定量的なデータから，要求管理上の要因と開発遅延の有無の関係性を開発遅延予測モデルとしてモデル化することができた．構築できたモデルを 2 つの評価方法によって評価し，モデルの信頼性・適合度の高さを確認した．

またモデルの予測推定精度をテストデータに対する平均正解率を評価指標として評価した．そしてステップワイズ法を用いて最適なモデルを探索することで，予測推定精度を向上させつつ，モデルに用いる説明変数を減少させ，シンプルで汎用性の高い要因でモデルを構築することができた．また実現場で働く要求管理者に対するインタビュー評価によっても，予測モデルに対して納得度の高いインタビュー結果を得ることができた．

さらにモデルの構築手法という観点でも，本モデルは評価に値すると考える．2 章で示した先行研究では，プロジェクト終了後のアンケート調査では信頼性のある分析ができないという指摘がされていた．それに対して本研究では，進行中のプロジェクトにおいても適用可能であること，汎用的かつ定量的な計測項目で評価が可能であること，の 2 点を満たす分析手法を提案できた．

以上をもって，本研究で構築した要求管理履歴データに基づく開発遅延予測モデルは正しいと言えそうである，と結論づけた．

7.1.2 本研究で明らかにできなかった点

前節において本研究で構築した要求管理履歴に基づく開発遅延予測モデルは一定の評価が得られると結論づけた．一方で本研究で明らかにできなかった部分として，下記の点が挙げられる．

1. 仕様量の大きさ，要求管理者の熟練度といった，本研究では未考慮である要因を含めた分析
2. 開発遅延につながる要因の詳細な分析
3. 要求カテゴリーによる影響
4. 開発遅延の有無ではなく遅延量を考慮した分析

最初にデータ収集の難しさや分析対象プロジェクトの制約といった理由で考慮できていない要因を含めた分析が挙げられる．要求の仕様量や要求管理者の熟練度といった要因である．このような要因を含めた多面的な分析を行うには至らなかった．

次に開発遅延につながる要因の詳細な分析である．今回明らかになった要求調整回数，ステークホルダー数という 2 つの要因と係数は，分析対象プロジェクトにおいての結果であり，すべてのアジャイル開発プロジェクトに共通してそのような傾向があるとは言えない．そのため，本研究で提案した分析手法を元に，より多くのアジャイル開発プロジェクトを分析していく必要性がある．また，この 2 要因のどのような点が開発遅延に影響を及ぼしているのかという要因の詳細な分析を行うまでには至らなかった．

そして，要求カテゴリーの影響度も本研究では明らかにすることはできなかった．要求カテゴリー毎の件数のばらつきが大きくサンプルサイズが小さいことが原因だと考えられるため，さらなる追加調査が必要である．

最後に，開発遅延の有無ではなく遅延量の大きさという点に着目した分析である．本研究では遅延量の小さなものも大きなものも等しく扱って分析を行ったが，遅延量の大小によって，要求管理上の要因に変化はあるのかを明らかにすることはできなかった．これらの本研究で明らかにできなかった点に対する今後の研究アプローチを次節で述べる．

7.2. 今後の展望

今後の展望として大きく 2 つの側面があると考えられる．1 つ目は，未考慮要因の検討，要因の詳細な分析という部分から，要求管理工程上の要因と開発遅延の関係性を多面的に分析していくことで，より当てはまりの良い予測モデルの構築ができると考える．さらに，本研究のアプローチを用いて，開発遅延のみならず，ソフトウェアの機能品質，コストといったプロジェクトの成否を評価する重要な要因と要求管理工程の関係性を明らかにしていくことが考えられる．

2 つ目は，研究を進める上で明らかにできたモデルや要因を元に，適切な要求

管理を支援するルールや仕組みの設計である．適切な要求分割基準の策定，要求管理者の要求管理スキル向上を促す仕組みの構築が挙げられる．以上 2 点をいくつかのトピックに分けて述べていく．

7.2.1 未考慮要因を含めた多面的な分析

本研究ではプロジェクト上の制約や計測の難しさといった観点から，考慮できていない要因がいくつかある．例えば仕様量の大きさ，要求管理者の熟練度といった要因である．これら未考慮の要因も計測できる手段を考案し，より多面的な分析を行うことで，精度の高いモデル化や要因の特定ができるのではないかと考えられる．

ステークホルダー数，要求調整回数の 2 要因の詳細な分析

本研究が分析対象としたプロジェクトにおいては，ステークホルダー数，要求調整回数の 2 要因が開発遅延に影響を及ぼす結果となった．この 2 要因が他のプロジェクトにおいても同様の結果をもたらすのかを調査していくことで，2 要因の影響度を一般化された形で証明していくことが望まれる．また 2 要因の詳細な分析を行うことで，それぞれの要因のどのような部分が開発遅延に影響を及ぼしているのかを明らかにすることができる．本研究では 2 要因の詳細な分析点として，ステークホルダー数であればステークホルダーの属性とスキル，要求調整回数においては調整間隔と調整内容を分析する必要があると推察している．それぞれの要因に対して詳細な分析を行うことで，どのような関係性やスキルを持つステークホルダーが関わったとき，どのような要求調整回数の増え方や内容の変化があったときに開発遅延が起きるのかを明らかにできるのではないだろうかと考える．

他のプロジェクト成否要因と要求管理の関係性の分析

本研究では，要求管理が及ぼす影響とその関係性を開発遅延という要因のみに絞って研究を進めてきた．今回の研究アプローチを用いて，他のプロジェクト成

否要因である機能品質・コストといった要因に対して要求管理が及ぼす影響を分析できるのではないかと考える．開発後の機能品質が低くなる要求と高くなる要求の要求管理履歴の差，開発に伴うコストが低い要求と高い要求の要求管理履歴の差といった点を，本研究のアプローチを用いることで同じように分析することができる．そしてそれぞれのモデルが構築できたとき，それらのモデルを複合的にモデル化することで，プロジェクトの成否と要求管理の関係性を包括的なモデルにより明らかにすることができるのではないかと考える．

遅延量を考慮した分析

今回の分析では，開発が開始された要求が，開発完了予定日に開発が完了しているか否かによって開発遅延の有無を計測しており，遅延量の大きさは考慮していない．開発遅延量は測定項目としてデータ収集することが可能である．そこから開発遅延量の小さい要求管理歴と開発遅延量の大きい要求管理履歴の差分を分析してくことで，遅延量の大きさに影響する要因を分析できると考えられる．

また開発遅延量の影響度合いをコストといった観点から評価する複合的なモデルを構築することによって，プロジェクトに対して影響度の高い開発遅延を引き起こす要求管理上の要因を特定できるのではないかと考えられ，プロジェクトに大きな影響を与える開発遅延を防ぐといった対策機構の考案につながるのではないかと考える．

要求管理者のスキルを考慮した分析

分析対象としたプロジェクトの要求管理者は１人であり，エンジニアリングに知見がある要求管理者であった．要求管理者に求められるスキルの幅は広く，前章のインタビューの山田さんのようにエンジニアリングに知見がない要求管理者は，技術要素の強い要求に対して苦手意識を抱えている．このように要求管理者の熟練度や得意・不得意領域といった要因によって要求管理に差がでるのか，という観点での分析も非常に大事なのではないかと考えている．そのような観点で

研究を進めることで，要求管理者の熟練度や得意・不得意領域の傾向を可視化し，要求管理者のスキル向上につながる仕組みを考案できるのではないかと推察する．

7.2.2 要求管理向上につながる対策機構の設計

本研究で得た結論を踏まえて，適切な要求管理基準の策定や，適切な要求管理を促す仕組み，要求管理者のスキル向上につながる仕組みについて述べる．

要求の適切な分割基準の抽出

本研究の分析結果から要求の適切な分割基準に関して，2つのアプローチがあると考えた．一つはステークホルダー数，要求調整回数による絶対的な基準，もう一つは要求管理者のスキルに応じた相対的な基準である．ユーザーストーリーの粒度という観点で今回の分析結果を考察すると，ステークホルダー数と要求調整回数が多くなる要求は，ユーザーストーリーの粒度が大きく曖昧なのではないかということが示唆される．1つの要求を定義するために，多くのステークホルダーを巻き込まなれけばいけかったり，要求調整回数がたくさん必要になるほど，要求で定義しなければいけない範囲が広いといった背景があると推察する．そのためこれらに対して，一つの要求に対して，ステークホルダー数がN人を超えたら，要求調整回数がN回を超えたら要求を分割する，といったような分割基準を設定できるのではないか．エンジニアリングのオブジェクト指向設計やクラス設計において，クラスの肥大化はアンチパターンであり，単一責任原則といったルールに則ってシンプルな責務の設計が望まれる．そのための一つの基準として1クラスは最大200行まで，1メソッドは30行まで，というように行数によって分割基準を設けている場合もある．そのような分割基準として，要求調整回数，ステークホルダー数，もしくはその組み合わせが利用できるのではないかと考えている．

また相対的な基準としての分割基準を設定するというアプローチもある．様々なバックグラウンドを持つ要求管理者の要求管理履歴を収集できると，スキルやバックグラウンドによって，要求カテゴリー毎に要求管理の得意・不得意が傾向として抽出できるのはないかと推察する．それが傾向として抽出できると先程の

絶対的な基準に加えて，相対的な得意・不得意度合いを加味した要求分割基準を，要求管理者毎に策定できる仕組みを構築できるのではないか．

分割基準に基づく要求管理支援システムの構築

前述した要求の適切な分割水準を元に要求管理を支援するシステムの構築も今後の展望として考えられる．特にアジャイル開発において，要求管理者は広い範囲の様々な要求を適切に管理し続けなければならず，業務量も多大である．要求の適切な分割水準やプロジェクト成否に関わる要求管理の要因といった今後の展望で明らかにされるであろう要因と，要求管理者のスキルを鑑みて，適切な粒度での要求分割の提案，適切なタイミングでの要求更新，要求獲得を促す要求管理支援システムを構築できると考える．

謝　　辞

本研究の指導教員であり，幅広い知見から的確な指導と暖かい励ましや厳しいご指摘をしていただきました慶應義塾大学大学院メディアデザイン研究科の砂原秀樹教授に心から感謝いたします．少し特殊なルートを辿った卒業となったため長期間籍を残すことになり，たくさんご迷惑をおかけしましたが，時には厳しく，時には温かいご指導をいただき，自身でも成長を感じることができました．長期間砂原教授の元で学ばせて頂いたことを大変誇りに思います．本当にありがとうございました．

研究の方向性について様々な助言や指導をいただきました慶應義塾大学大学院メディアデザイン研究科の加藤朗教授に心から感謝いたします．研究の方向性に迷ったときや研究室の定例の進捗報告においてもいつも丁寧なご指導をいただき，諦めずに研究を遂行することができました．本当にありがとうございました．

また副査として研究の方向性や内容に関して助言をいただきました慶應義塾大学大学院メディアデザイン研究科の大川恵子教授に心から感謝いたします．

修士課程入学時の同期でもあり，研究活動全般において数多くの助言をいただきました慶應義塾大学大学院メディアデザイン研究科博士課程の岡田光代氏に心から感謝いたします．また，複数回のインタビューによって，研究内容に関して助言を頂きました山田真由美氏に心から感謝いたします．

最後に大学院進学や休学といった意思決定を心から応援してくださり，経済面・精神面においてご支援頂きました両親と家族に感謝と御礼を申し上げます．

参 考 文 献

[1] 一般社団法人日本情報システム・ユーザー協会. 企業it動向調査2019（2018年度調査）. https://juas.or.jp/cms/media/2017/02/it19_ppt.pdf, 2019. [Online; accessed 23-Jan-2020].

[2] Bertrand Meyer. アジャイルイントロダクション. 近代科学社, 2018.

[3] Standish Group. Chaos report 2015. https://www.standishgroup.com/sample_research_files/CHAOSReport2015-Final.pdf, 2015. [Online; accessed 23-Jan-2020].

[4] IPA 技術本部ソフトウェア高信頼化センター山下博之. アジャイル開発の現状と課題. https://www.ipa.go.jp/files/000035262.pdf, 2013. [Online; accessed 23-Jan-2020].

[5] 古山恒夫, 菊池奈穂美, 安田守, 鶴保征城. ソフトウェア開発プロジェクトの遂行に影響を与える要因の分析. 情報処理学会論文誌, Vol. 48, No. 8, pp. 15–22, 2007.

[6] 江崎和博. ソフトウェア開発プロジェクトの成功に関係する要因の研究. 電子情報通信学会論文誌 A, Vol. 98-A, No. 9, pp. 571–579, 2015.

[7] 古山恒夫, 菊池奈穂美, 安田守, 鶴保征城. ソフトウェア開発プロジェクトの遂行に影響を与える要因の分析. 情報処理学会論文誌, Vol. 48, No. 8, pp. 15–22, 2007.

[8] Mike Beedle. アジャイルソフトウェア開発宣言. https://agilemanifesto.org/iso/ja/manifesto.html, 2001. [Online; accessed 20-Dec-2019].

[9] StateOfAgile. 13th annual state of agile report. p. 9, 2019.

[10] 情報サービス産業協会 REBOK 企画 WG. 要求工学知識体系. 近代科学社, 2011.

[11] 青山幹雄, 中谷多哉子, 斎藤忍, 鈴木三紀夫, 中崎博明, 藤田和明, 鈴木律郎. 要求工学の動向と要求工学知識体系 rebok. 情報システム学会誌, Vol. 6, No. 1, p. 56, 2010.

[12] ITC 大阪城 REBOK 研究会. Itc による rebok(要求工学知識体系) 適用研究報告書. http://www.itc-osakajo.jp/upload/report/S11004.pdf, 2012. [Online; accessed 20-Dec-2019].

[13] 藤原由希子. Prince モデルを用いたプロジェクト分析と要求成熟過程の推定に関する考察. ソフトウェアエンジニアリングシンポジウム 2011 論文集, pp. 1–6, 2011.

[14] Kenneth S. Rubin. エッセンシャルスクラム. 翔泳社, 2011.

[15] 川端薫, 千種実, 横田毅. ソフトウェア開発プロジェクトの成否に影響を与える要因の統計的分析. プロジェクトマネジメント学会 2012 年度春季研究発表大会予稿集, Vol. 2509, pp. 370–375, 2012.

[16] 高橋圭一. 学生による実ソフトウェア開発における納期遅延要因に関する分析. プロジェクトマネジメント学会 2007 年度秋季研究発表大会予稿集, Vol. 2212, pp. 183–188, 2007.

[17] 大宮望, 大葉充晶, 山本久志, 丸山友希夫, 中邸良樹. 納期遅延プロジェクトが及ぼす保守工程への影響. 一般社団法人経営情報学会, 2011.

[18] Atlassian. Trello. https://trello.com/, 2011. [Online; accessed 22-Dec-2019].

[19] 株式会社クレイ / KRAYinc. 情報共有ツール docbase(ドックベース) — ストレスフリーなドキュメント共有ツール. https://docbase.io/, 2016. [Online; accessed 30-Jan-2020].

[20] 株式会社ベーシック. Uu とページビューとセッションとの違いを理解しよう | ferret. https://ferret-plus.com/592, 2019. [Online; accessed 30-Jan-2020].