独立行政法人情報処理推進機構 委託

2012 年度ソフトウェア工学分野の先導的研究支援事業
「モデルを含む設計成果物の集積と
その活用方法に関する研究」
成果報告書

平成 25 年 1 月

国立大学法人九州大学

本報告書は独立行政法人情報処理推進機構　技術本部　ソフトウェア・エンジニアリング・センターが実施した「2012 年度ソフトウェア工学分野の先導的研究支援事業の公募による採択を受けて九州大学システム LSI 研究センター（研究責任者 久住 憲嗣）が実施した研究の成果をとりまとめたものである.

目次

研究成果概要

１．背景

近年，大規模，複雑化し，また，諸外国との激しい競争にさらされている．そのため，ソフトウェアの工数削減や設計品質の向上を目的としてUMLなどのモデルを援用して開発することが，ソフトウェア開発における大きな流れである．モデリング技術を利用することで，正確な設計が可能となり，設計が可視化でき，開発者間でのコミュニケーションが円滑になる.さらにモデルを中心に据えた自動化技術であるモデル駆動開発(Model-Driven Development; MDD)を用いると，開発早期でのシミュレーションや検証を行えるようになるなどの多くのメリットがある．従来の自然言語とプログラミング言語を中心とした開発から，モデルを用いた開発への移行は不可避である．そのためモデリングやMDDに関する研究やケーススタディが盛んに行われている．

しかしながら，研究を実証的に遂行する上で必要不可欠なデータが圧倒的に不足している．研究などに利用できるモデルを中心として開発された第三者が利用可能な設計成果物がほとんど無く，さらに，モデルのバグ情報や編集履歴まで含めて利用できる成果物は皆無である．モデリング言語ではなく，プログラミング言語で記述されたオープンソースソフトウェアを対象として，ソースコード本体とバグ情報，編集履歴などを元に，開発に有用な情報を抽出するリポジトリマイニング[1]の研究が盛んなこととは対照的である．

２．研究課題

第三者が利用可能なモデルを含んだ設計成果物がないことは，以下の2点に起因する．

1) モデルの編集履歴を収集できるツールの不在

既存のモデリングツールはモデルの履歴を保存する機能が無い．また，既存のリポジトリシステムにモデルを単純に登録するだけでは，モデルの履歴に対して横断的に検索をかけることが困難である．モデルとその編集履歴を収集できるツールの開発が必要不可欠である．

2) 第三者に利用可能なデータを収集する場が無い

ソースコードを対象とする研究の場合は，オープンソースプロジェクトの成果物を利用可能であるが，モデルを含む設計成果物は公開されていないことが多い．モデルを対象とした実証的研究を促進するためには，第三者に利用可能な設計成果物の収集と公開が必要である．

また，これらの問題に伴って

3) モデルを含む設計成果物を大局的にとらえるための枠組みが不足している

研究対象とできる第三者が利用可能なモデルを含んだ設計成果物が不足しているため，これらを大局的に分析して開発を支援する手法が十分に発達していない．設計成果物のメトリクスを計測して，大局的視点から開発を支援する手法が必要である．

---

[1] 設計成果物とその履歴を格納するリポジトリに格納されている膨大なデータを元に，開発に有用な知見を導出する手法

そこで本研究では，これらの問題を解決することで，モデルを用いた開発においても実証的に研究を進められるようにすべく,1) モデルを集積するためのモデリングツールを開発し，2) 講義や Project-Based Learning (PBL)等でモデルを集積して，3) 集積できたモデルとその編集履歴を公開する．そして，4) 既存のメトリクスを利用して予備評価する．

本研究課題の目標は，モデルを対象とした実証的研究に利用できる，モデルとその編集履歴などを継続的に集積し,集積したデータを利用するための環境を整備することである．この大目標を実現するためのサブゴールとして以下の 2 点を設定する．

1) モデルとその編集履歴などを集積するための，モデリングツールの開発

モデルを利用した開発において，モデルとその編集履歴，バグ情報などの設計成果物を，自動的に収集し，データベースに格納するツールを開発する．そのツールの実用性を検証するために，本学で実施している講義，PBL などでツールを使用する．

2) 集積された設計成果物を利用したメトリクスの評価

集積されたデータが有用であることを実証すべく，ソースコードを対象として提案されているメトリクスを集積データに適用し，バグ密度の予測手法が適用可能か評価する．

３．研究アプローチ

では以下の手順で研究を進める．まず,

① モデルとその編集履歴を集積するためのツール改変を実施し，モデルとその編集履歴を集積するための環境を整備する．

同時に，モデルとその編集履歴を活用する環境を整備すべく,

② モデルの編集履歴を検索・利用するための API を定義し，動作確認する

次に，その改変したツールを用いて,

③ 講義や PBL での利用を通したツールの有効性評価を実施し,講義やプロジェクトにおいてモデルとその編集履歴を収集できることを確認し，講義，及び，プロジェクトにおけるモデルとその編集履歴を収集する．

最後に③で収集したデータを対象として

④ モデルを対象としたリポジトリマイニングの実証評価を行い，知見を得る．

以下に本研究において実現する環境を図示する．

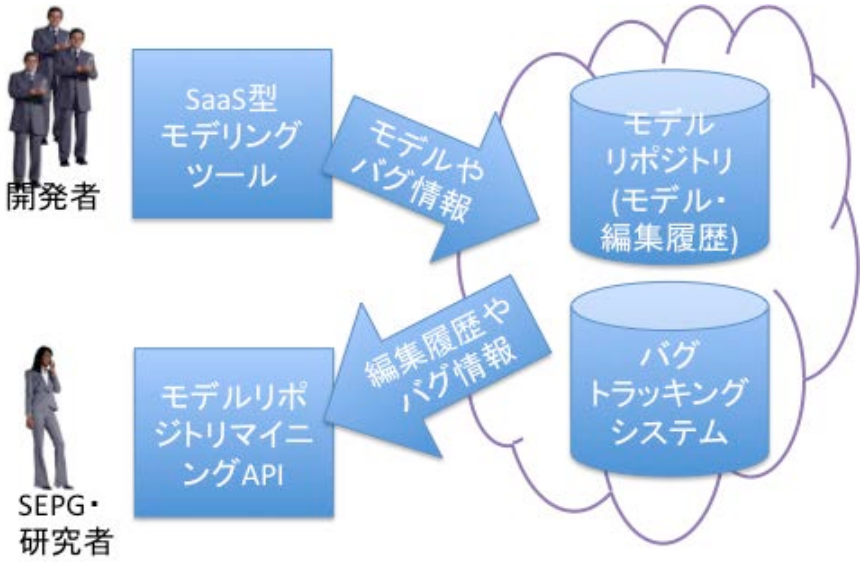

図１　本研究が構築する環境

４．実施内容

それぞれのサブゴールについて実施した結果以下のような結果となった．

① モデルとその編集履歴を集積できるようにツールを改変し，動作確認する

既存のモデリングツールはモデルの履歴を保存する機能が無かった．また，既存のリポジトリシステムにモデルを単純に登録するだけでは，モデルの履歴に対して横断的に検索をかけることが困難である．そのため，モデルとその編集履歴を収集できるツールの開発が必要不可欠であった．

そこで本研究では，下図のように SaaS 型ドメイン特化モデリング言語ツールである clooca を対象として，モデルとその編集履歴を集積し，分散協調開発を行えるようにするためのモデルリポジトリを構築した．本研究の遂行によりモデルとその編集履歴，及び，バグ情報などを，ツールの支援のもとに収集できるようになった．

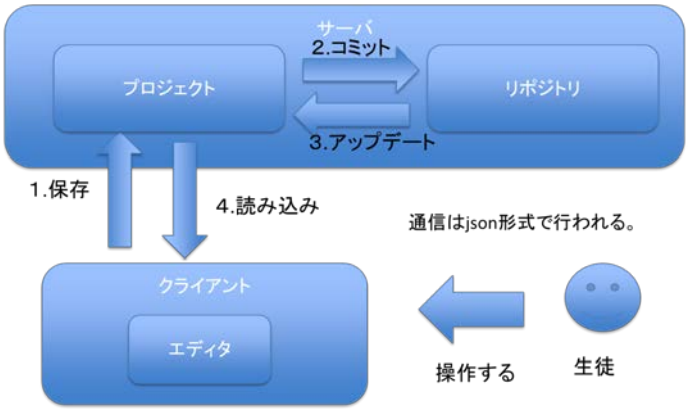

図２　リポジトリシステム概要

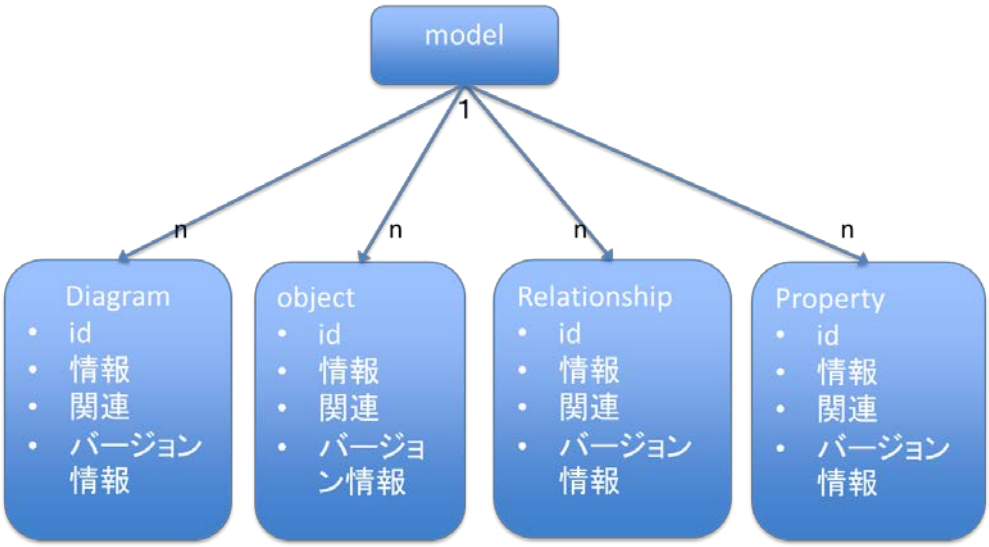

図 3　モデルの構造

さらに，バグ情報とリポジトリ上の変更を対応づけるために，バグトラッキングシステムとの連携機能を開発した．

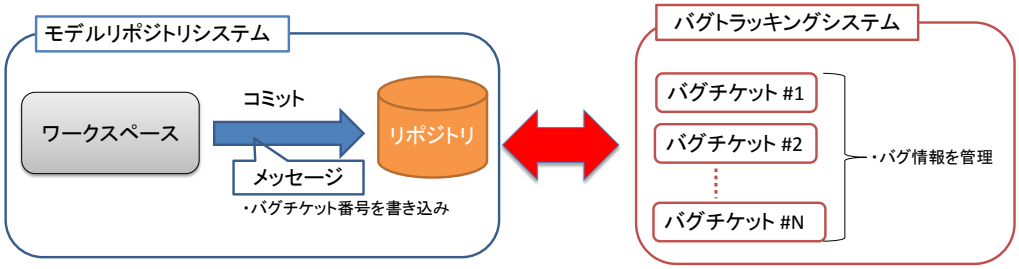

図 4　バグトラッキングシステムとの連携

これらの活動により，モデルとその編集履歴，及び，バグ情報などを，ツールの支援のもとに収集できるようになったが，一部，編集履歴の蓄積に不十分な点がはっ決された．③の活動において収集したモデルとその編集履歴を解析する際に，一部の受講生はリポジトリにコミットする頻度が低く，その受講生が作成した設計成果物の成長過程を詳しく見ることができない事例が発生した．今回，モデルをリポジトリにコミットするタイミングは，1 機能追加，1 バグフィックス，1 課題終了ごと，など，講義の中でインストラクションを与えたが，コミットの実施自体は受講生の主体性に任せる結果となった．対策として，コミットしたタイミングで編集履歴を保存するのではなく，単に受講生がモデルを保存したタイミングですべての編集履歴を残すことを検討中である．

② モデルの編集履歴を検索・利用するための API を定義し，動作確認する

①で集積した設計成果物に対するメトリクスを研究者や教育者の要望に応じて計算，提示することを支援するための API を構築した．計算したいメトリクスは，研究内容，授業支援内容などにより異なるため，できる限り多くのメトリクスを計算できるように開発するのでは無く，成長させやすいアーキテクチャとした（下図）．

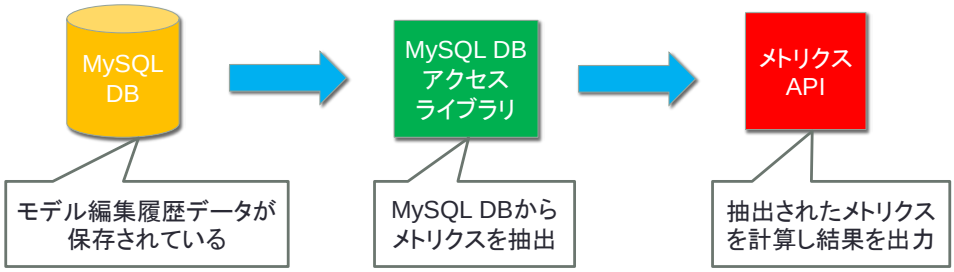

図 5　メトリクス API アーキテクチャ

具体的には，データベースに格納されているモデルとその編集履歴を取り出しメトリクスの計算を支援するレイヤを構築し，さらのその上位にメトリクスを計算するレイヤ，指定したメトリクスを計算し提示する Web インタフェースを開発した（下図）．

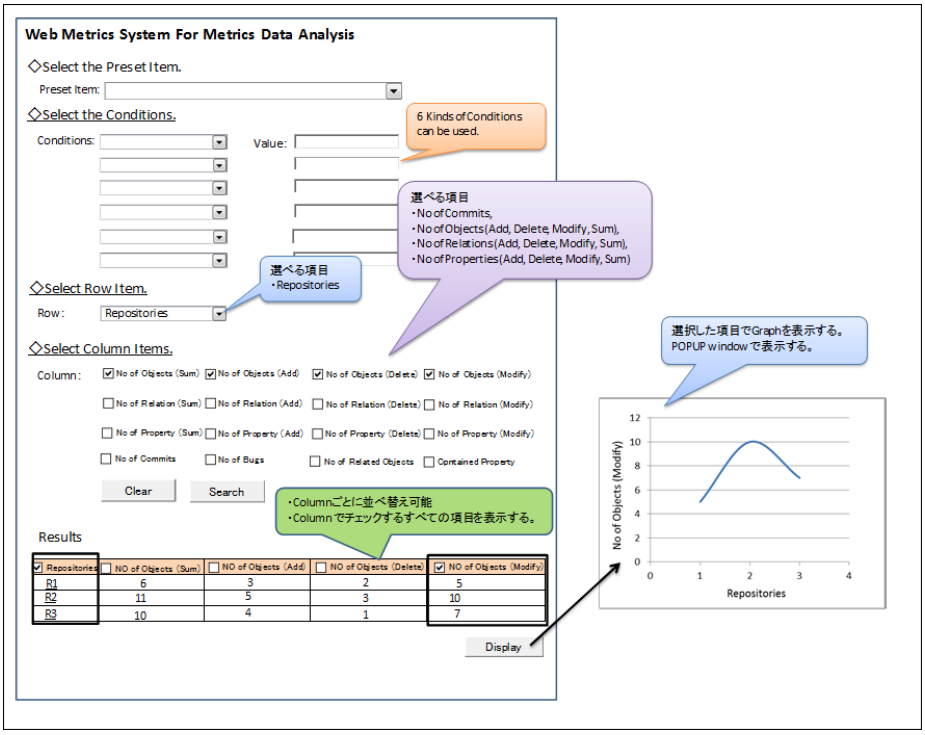

図 6　画面イメージ図

③ 講義，PBL 等でツールを利用して，モデルとその編集履歴を集積する

講義や Project-Based Learning (PBL)において，受講者に開発したツールを利用させ，モデルとその編集履歴の集積を行った．このことによってモデルとその編集履歴を収集し，教育や研究上の知見を得るための材料とする

講義は 2 種類実施することとした．ひとつは UML などのモデルを初めて学ぶ初学者向けの講義，もうひとつは UML を用いたモデル駆動開発を学ぶモデル駆動開発特論における講義である．両者ともにモデリングを必要とするような演習課題を与え，その演習課題を受講生が実施する過程のモデルを収集することとした．初学者向けの講義は徳山工業高等専門学校における特別講義にて実施した．また，モデル駆動開発特論は九州大学大学院システム情報科学府にて実施した．

両講義において共通の総合演習について説明する．

総合演習では架空の運輸会社の自動搬送ロボットの開発業務を請け負うこととする．開発ターゲットは LEGO Mindstorms NXT を用いて開発した自律型車両ロボットである．自律型車両ロボットには，超音波センサ，バンパ，光センサ，タッチセンサが搭載されており，それぞれ，ロボット側面の側壁を検知することによる配送先判定，障害物のあたり判定，ライン判定，荷物判定を行うために利用する．

演習のコース図を以下に示す．自動化する業務は，運搬業務，転送サービス，回送業務の 3 種類であり，これらの業務は荷物や配達先の有無により選択される．課題は光センサでコースの黒いラインをとレスし，配達先もしくは転送先，車庫で停止し，それぞれの地点で適切な動作を行い，所定の位置に荷物を届けることである．

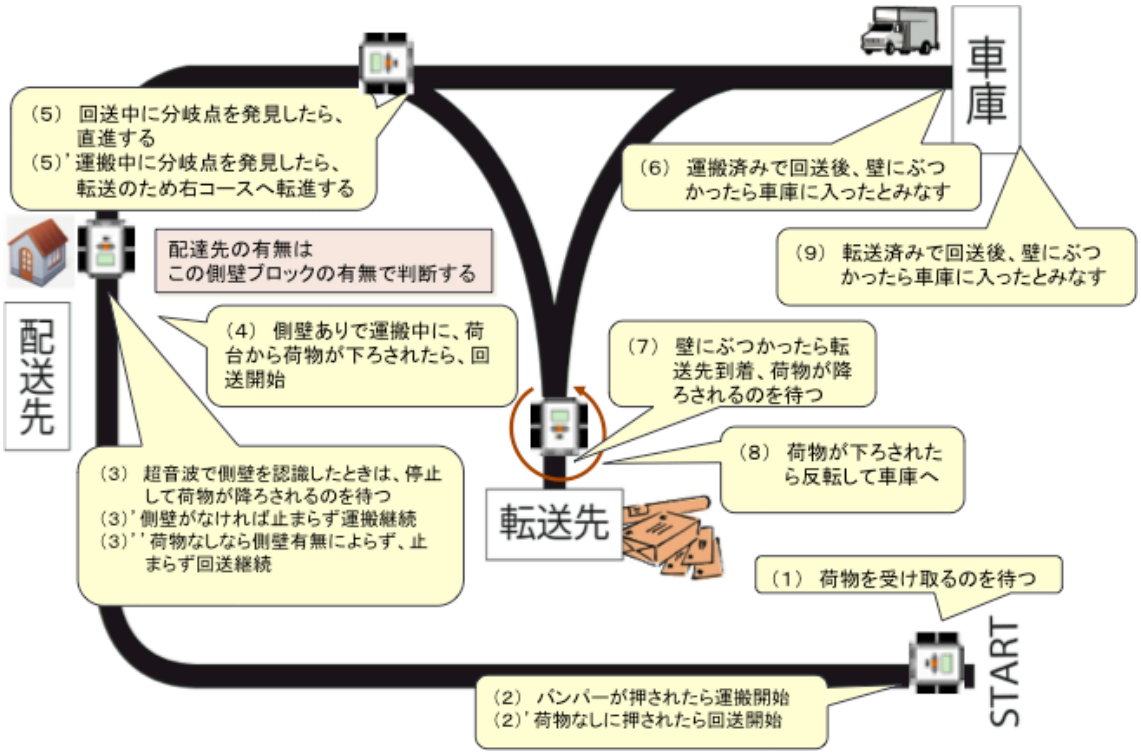

図 7　総合演習課題コース

その結果，改変した clooca を使用することで，当初想定した規模と数のモデルとその編集履歴が収集できることが確認できた．被験者 16 名分のモデルとその編集履歴を収集することができた．その結果，下図に示すようなデータを収集することができた．

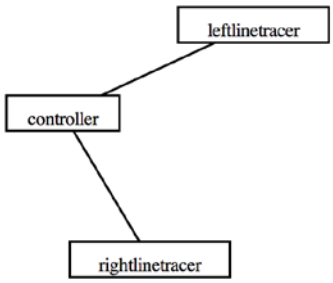

図 8　ある被験者のクラス図

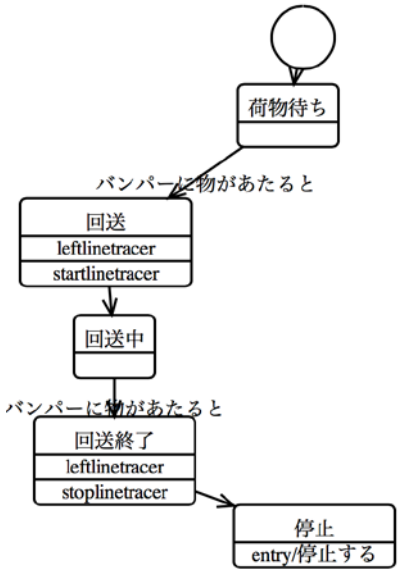

図 9　ある被験者のステートマシン図 1

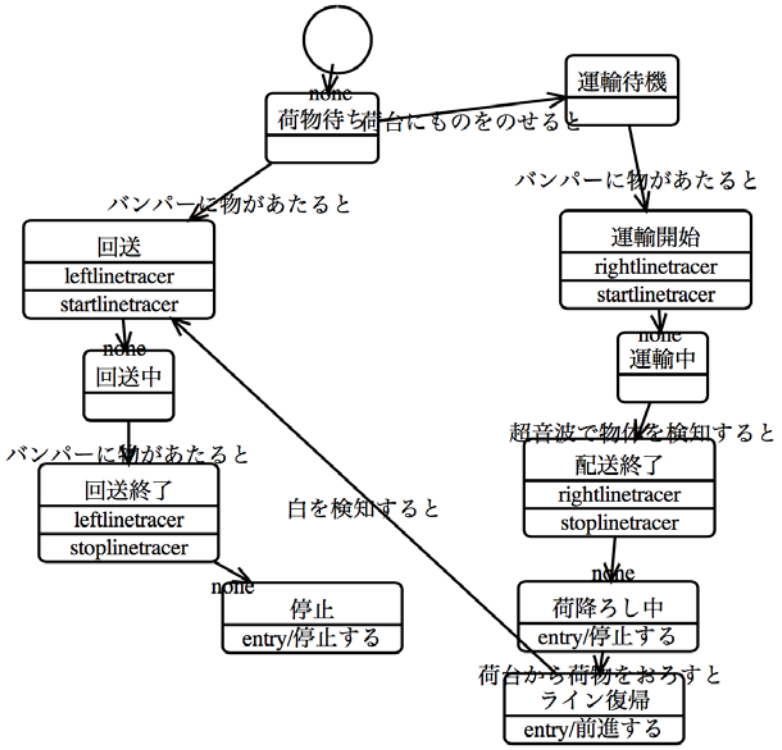

図 10　ある被験者のステートマシン図 2

より大規模なモデルとその編集履歴の収集を目指しPBLを対象とした集積を行った．PBL においては ESS ロボットチャレンジという情報処理学会 組込みシステム研究会 組込みシステムシンポジウムにおける特別企画である．PBL においてはプロジェクト途中で課題の変更を余儀なくされたため，当初想定した規模の設計成果物を得ることができなかった．

これらの活動を通じて，講義，及び，PBL において，改変した clooca を使用することでモデルとその編集履歴が収集できることが確認できた．このことによって，講義の現場，もしくは，開発の現場において同様の仕組みを利用することで，講義，開発支援のために必要なデータが得られるようになるものと考える．

公開可能なモデルとその編集履歴を収集した事例はきわめてまれであり，研究コミュニティにインパクトを与える成果であると考える．

また，特質する事項として clooca はインストール不要な SaaS 型のモデリングツールであるため，比較的大人数が使用する講義や開発においての導入コストが著しく低くなる．今回実施した講義についても，従来であればツールの導入に 1 日程度時間を要していたが，今回は数時間のインストール作業で十分であった．また，SaaS 型であるためすべてのモデルがサーバ上にあり，教員がチェックしやすい状態にあるため，レビューし修正させるべきモデルを講義中に発見することができた．

④ 既存のコードを対象としたメトリクスがモデルでも有効に働くかどうか評価

③において収集したデータを対象として，分析を行った．特に均質なデータが取得できている徳山工業高等専門学校における授業において収集できたデータを対象として分析した．また，当該データは MDD をしたチームとしなかったチームに分けて収集しているため，それらについての考察を行った．

分析は定量，定性の両面から実施した．定量的な分析としては，ステートマシン図の understandability と高い関係のあるメトリクスとして，Number of Activities(NA)，Number of States(NS)，Number of Transitions(NT) の 3 つについて計測を行った（下表）．

表 1　ステートマシン図の understandability に関するメトリクス
(MDD を利用しないグループ)

	NA	NS	NT	NSE
student1	9	9	11	2
student2	6	6	8	2
student3	8	8	11	2
student4	8	8	10	2
student5	7	7	9	2
student6	7	8	12	2
平均	7.5	7.7	10.2	2.0

表 2　ステートマシン図の understandability に関するメトリクス
(MDD を利用するグループ)

	NA	NS	NT	NSE
student1	7	19	20	7
student2	5	12	15	7
student3	4	6	8	3
student4	5	9	10	3
student5	5	7	12	7
student6	12	15	26	0
平均	6.3	11.3	15.2	4.5

定性的な分析としては，品質カテゴリに関するエラーを持つクラス数のカウント，及び，クラス名の分類を行った．

その結果,MDD をしたチームとしなかったチームにおけるモデルの品質傾向が明らかとなり，講義に関する改善点が明らかとなった．具体的には，UML モデリングの教育効果を上げるためには，単純に MDD ツールを用いた開発をさせるのではなく，コードの自動生成や動作確認の回数を制限した演習を実施する必要がある，レビューを適切なタイミングで実施すべきである，などである．

未達成の課題と今後の研究予定

① PBL の継続実施によるさらなるモデルとその編集履歴の集積

当初は ESS ロボットチャレンジの昨年度までの課題である模型飛行船制御を対象とした開発を進めてきた．しかしながら，12 月初旬ごろからアメリカにおけるヘリウム

ガスの生産設備のトラブルにより，ヘリウムガスが入手困難となった．飛行船を浮かせる上で必要不可欠であるヘリウムガスが入手困難となったことで，飛行船制御システムを題材としたPBLの続行が不可能になり，課題変更を余儀なくされた．そこでPBLチームは急遽,PBL課題を変更し2台の地上走行するロボットが協調走行する課題に変更した．その結果，当初想定した規模のモデルとその編集履歴が収集できなかった．

現在も当PBLは続行中であり，平成25年7月まで継続予定である．PBL終了時には当初想定した規模のモデルとその編集履歴が得られるものと思われる．

新たに見いだされた課題と今後の研究予定

① cloocaのソーシャルネットワークサービス(SNS)化

中間報告において，データの収集のみではなく，cloocaをSNS化しモデルや部品の交換，それらに対してコメントをつけ合うなどのプロジェクト環境を創出してほしい，とのコメントをいただいた．今回は研究期間，内容の関係上実施することができなかったが，cloocaのSNSサービス化は当初から構想しており，今後，実現していきたいと考えている．

② 他団体が提供しているプロジェクト管理，診断ツールとの連携

中間報告において， UMLの更新の履歴等、上流工程の開発過程データを収集したいという要求がたくさん出ているが，世の中で使われているツールにはそのような機能が付いていない，IPAが提供しているプロジェクト管理，診断ツールと連携できるようにし，世の中に広げるようにしてほしい，とのコメントを頂いた．他ツールとの連携，及び，cloocaの普及推進は重要な課題だと考えており，今後，取り組んでいきたい．

③ メトリクスのリアルタイムな提供

今回は比較的静的にメトリクスを計算し，表示を行った．この方式では講義終了後に演習の過程を分析するためには十分であるが，授業中に必要に応じて受講生をフォローするためには不十分である．受講生が設計成果物を変更したタイミングでメトリクスを再計算し，問題があるレベルに達したら講師に通知するなどの仕組み化が必要である．

④ 大規模なデータ収集のための実プロジェクトを対象とした実験

有効な知見を得るためには，講義やPBLではなく，実際の開発現場において今回開発したツールを利用してもらい，データを収集し，分析することが必要である．現状のcloocaはドメイン特化モデリング言語の開発支援ツールであり，その上でのモデリング言語の定義はユーザに任せられている．本研究を遂行する上で必要最低限のモデリング言語を定義し，コード生成器を開発して利用したが，実際に開発で利用するためには，UMLのサポートをさらに進めていく必要があるものと考える．

１．研究の背景および目的

1.1 背景

近年，大規模，複雑化し，また，諸外国との激しい競争にさらされている．そのため，ソフトウェアの工数削減や設計品質の向上を目的としてUMLなどのモデルを援用して開発することが，ソフトウェア開発における大きな流れである．モデリング技術を利用することで，正確な設計が可能となり，設計が可視化でき，開発者間でのコミュニケーションが円滑になる.さらにモデルを中心に据えた自動化技術であるモデル駆動開発(Model-Driven Development；MDD)を用いると，開発早期でのシミュレーションや検証を行えるようになるなどの多くのメリットがある．従来の自然言語とプログラミング言語を中心とした開発から，モデルを用いた開発への移行は不可避である．そのためモデリングやMDDに関する研究やケーススタディが盛んに行われている．

しかしながら，研究を実証的に遂行する上で必要不可欠なデータが圧倒的に不足している．研究などに利用できるモデルを中心として開発された第三者が利用可能な設計成果物がほとんど無く，さらに，モデルのバグ情報や編集履歴まで含めて利用できる成果物は皆無である．モデリング言語ではなく，プログラミング言語で記述されたオープンソースソフトウェアを対象として，ソースコード本体とバグ情報，編集履歴などを元に，開発に有用な情報を抽出するリポジトリマイニング[2]の研究が盛んなこととは対照的である．

1.2 研究課題

第三者が利用可能なモデルを含んだ設計成果物がないことは，以下の2点に起因する．

1) モデルの編集履歴を収集できるツールの不在

既存のモデリングツールはモデルの履歴を保存する機能が無い．また，既存のリポジトリシステムにモデルを単純に登録するだけでは，モデルの履歴に対して横断的に検索をかけることが困難である．モデルとその編集履歴を収集できるツールの開発が必要不可欠である．

2) 第三者に利用可能なデータを収集する場が無い

ソースコードを対象とする研究の場合は，オープンソースプロジェクトの成果物を利用可能であるが，モデルを含む設計成果物は公開されていないことが多い．モデルを対象とした実証的研究を促進するためには，第三者に利用可能な設計成果物の収集と公開が必要である．

また，これらの問題に伴って

3) モデルを含む設計成果物を大局的にとらえるための枠組みが不足している

研究対象とできる第三者が利用可能なモデルを含んだ設計成果物が不足しているため，これらを大局的に分析して開発を支援する手法が十分に発達していない．設計成

[2] 設計成果物とその履歴を格納するリポジトリに格納されている膨大なデータを元に，開発に有用な知見を導出する手法

果物のメトリクスを計測して，大局的視点から開発を支援する手法が必要である．

1.3. 研究の意義

本研究を遂行することによって，モデルとその編集履歴などの，モデルを対象とした実証的研究に必要なデータを集積するしくみができる．そのため，モデルを対象とした種々の研究の礎となりうる．さらに，SaaS 型開発環境を生かした開発に関するデータの利活用基盤となりうる．以下にそれぞれ詳しく説明する．

- モデルを対象としたリポジトリマイニングの推進

  モデルとその編集履歴を集積できるツールの開発を実施することによって，対象となるデータが少ないがためにあまり実施されてきてこなかった，モデルとその編集履歴を対象としたデータの分析，リポジトリマイニングに関する研究を促進する．具体的には，ソースコード対象で実施されているような，バグ密度を推定することによるテストやレビューを効率化する手法が，モデルを中心とした開発においても適用できるようになる．特に従来のソースコード対象の手法では不可能であった，比較的，上流のモデルにおいても，同様の手法が適用できるようになる．

- SaaS 型開発環境による開発に関するデータの利活用基盤

  本研究により開発したツールの利用が広まれば，従来であれば収集することのできなかったデータを組織横断的に収集することができるようになり，利活用できる．従来の開発環境はクライアント上で動作し，その上で設計成果物を作成した上で，必要に応じて設計成果物をリポジトリ上に登録する．また，このリポジトリは通常は組織を超えて共有されない．ところが，SaaS 型開発環境では第三者機関がツールを提供することになり，設計成果物を組織を超えて収集することができる．このデータの利用には匿名化，抽象化するなどの細心の注意が必要ではあるが，従来は困難であった組織を超えたデータの利活用が可能になる．

２．実施内容

2.1 研究アプローチ

本研究では，前章で指摘した 1）モデルの編集履歴を収集できるツールの不在，2）第三者に利用可能なデータを収集する場が無い，3）モデルを含む設計成果物を大局的にとらえるための枠組みが不足している，といった問題を解決するべく，モデルを用いた開発においても実証的に研究を進められるようにすべく，1）モデルを集積するためのモデリングツールを開発し，2）講義や Project-Based Learning（PBL）等でモデルを集積して，3）集積できたモデルとその編集履歴を公開する．そして，4）既存のメトリクスを利用して予備評価する．以下に本研究において実現する環境を図 2-1 に示す．

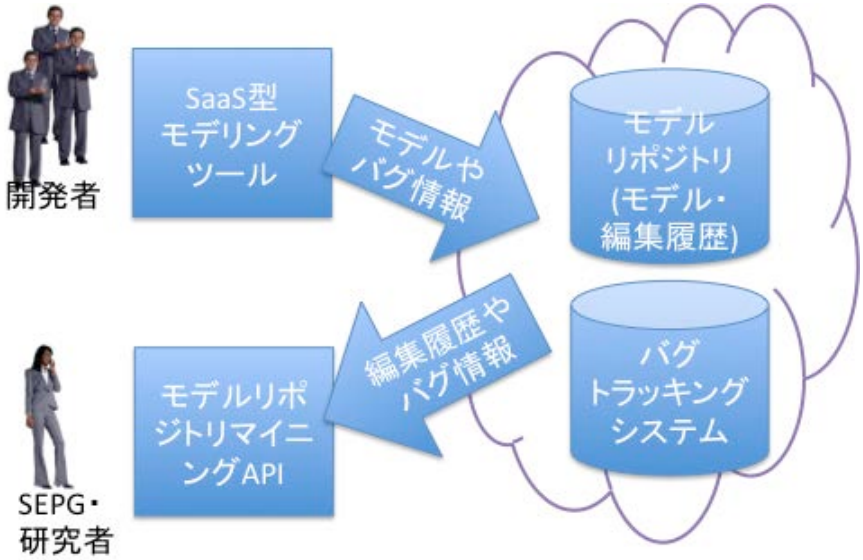

図 2-1　本研究が構築する環境

本研究課題の目標は，モデルを対象とした実証的研究に利用できる，モデルとその編集履歴などを継続的に集積し，集積したデータを利用するための環境を整備することである．この大目標を実現するためのサブゴールとして以下の 2 点を設定する．

1） モデルとその編集履歴などを集積するための，モデリングツールの開発

　モデルを利用した開発において，モデルとその編集履歴，バグ情報などの設計成果物を，自動的に収集し，データベースに格納するツールを開発する．そのツールの実用性を検証するために，本学で実施している講義，PBL などでツールを使用する．

2） 集積された設計成果物を利用したメトリクスの評価

　集積されたデータが有用であることを実証すべく，ソースコードを対象として提案されているメトリクスを集積データに適用し，バグ密度の予測手法が適用可能か評価する．

2.1.1 研究の全体像

本研究では以下の手順で研究を進める．まず，1）モデルとその編集履歴を集積するためのツール改変を実施し，モデルとその編集履歴を集積するための環境を整備する．次に，その改変したツールを用いて，2）講義やPBLでの利用を通したツールの有効性評価を実施し，講義やプロジェクトにおいてモデルとその編集履歴を収集できることを確認する．そして，3）ETロボットコンテストなどのコンテスト参加者へのツール提供を通した有効性評価，によって，講義，及び，プロジェクトにおけるモデルとその編集履歴を収集する．最後に3)で収集したデータを対象として4）モデルを対象としたリポジトリマイニングの実証評価を行い，知見を得る．以下にそれぞれについて詳しく述べる．

1） モデルとその編集履歴を集積するためのツール改変

Webベースのモデリング，MDDツールであるcloocaを改変することで，モデルとその編集履歴を集積するツールとする．cloocaはSaaS（Software-as-a-Service)型のツールで，Webブラウザ上で動作する．cloocaは本研究参加メンバーが，IPA未踏ソフトウェア創造事業の支援を一部受けて開発したツールである．SaaS型であるため，すべての描かれたモデルはサーバ側に蓄積される．

本研究においてはcloocaに機能を追加し，a）モデルの編集履歴を保存できるようにし，b）保存された膨大なモデルデータに対して検索を実行して，必要な情報を抽出するAPIを定義，実装する．さらに，c）バグ管理システムと連携し，バグ情報とのトレーサビリティがとれるようにする．

2） 講義やPBLでの利用を通したツールの有効性評価

改変したcloocaの有効性を評価するために，講義やPBL(Project-based Learning)で受講生に利用させる．九州大学システム情報科学府では，本研究メンバーが中心となり，a）モデル駆動開発を講義と演習を通して学ぶことができるモデル駆動開発特論，b）UMLなどのモデリング言語を援用したPBLを実施している．特にb)のPBLでは1年間かけて，比較的大規模なシステムを開発している．また，本PBLでは複数チームがあり，多様なシステムを開発している．講義とPBLを通して，モデルと編集履歴，バグ情報などを蓄積することで，ツールの有効性を評価する．

3） ETロボットコンテストなどのコンテスト参加者へのツール提供を通した有効性評価

提案者らはETロボコン九州地区大会において，実行委員長やモデル審査員を歴任しており，また，九州地区大会特別企画としてモデル駆動開発審査を実施している．さらに，ESSロボットチャレンジのコアメンバーである．これら活動を活用して，さらなるモデルと編集履歴の集積のために，ETロボットコンテストやESSロボットチャレンジなどのコンテスト参加者に，改変したツールの使用を依頼する．参加者がツールを利用して開発する過程において，モデルデータと編集履歴，バグ情報などの収集を行う．

4） モデルを対象としたリポジトリマイニングの実証評価

収集したモデルとその編集履歴を利用して，リポジトリマイニングが可能であることを

実証評価する．具体的には，コードを対象とした既存のメトリクスを基に，プロダクト及びプロセスメトリクスを測定し，それを元にバグ密度などを推定できるかを評価する．

2.1.2 関連するこれまでの研究について

関連研究として，SaaS 型の開発環境，及び，クライアントで動作するモデル駆動開発ツール等について取り上げる.次に我々の研究室において開発を進めてきた clooca について説明する.

(1) 従来研究

現在，利用可能な Web ベースの IDE として,CtrlSpace,orion,cloud9ID 等がある.CtrlSpace はプラグインを追加できる仕組みを持っている点が,他の Web ベース IDE と異なっている.orion と cloud9IDE は Javascript+HTML 製の Web ベース IDE であ る.orion は Eclipse Foundation が開発しており,cloud9IDE は ajax.org が開発している.これら既存のサービスや技術に共通する特徴として,リアルタイム協調開発のようなチームコラボレーション機能を持っていることがあげられる.また,これらはどれもソースコードを編集対象としている.紹介した Web ベースの IDE は,どれもソースコードを編集対象としていて,ソフトウェアモデルをうまく扱えている環境はない.今後の開発ツールはソースコードだけでなく,抽象度の高いソフトウェアモデルもうまく扱える必要がある.

モデル駆動開発ツールやドメイン特化言語ツール,その技術として MetaEdit, DSLTools, Graphitical Modeling Project, Enterprize Archtect がある.これらのツールはどれもクライアントマシンにインストールして使用するソフトウェアである.紹介した DSL ツールは,どれもクライアントマシンにインストールして使用する必要がある.このタイプのソフトウェアでは,ツールの PC への導入する際に,インストール作業や環境設定が必要で，非常に大変である.またデータを他のユーザと共有したり,異なる PC で,同じ環境を再現することが不得意である.これらの問題を解決するには,開発環境を SaaS として提供することが有効である.

(2) 研究室における従来の取り組み

そこで，我々の研究室では SaaS 型モデル駆動開発ツールである clooca の開発を進めてきた．まず，clooca の背景技術であるドメイン特化モデリング言語(Domain-Specific Modeling Language; DSML)[2-1]について述べ，次に我々が開発を進めてきた clooca について述べる.

Unified Modeling Language(UML)等のような汎用の言語では無く，独自のモデリング言語を定義し，そのモデリング言語を利用して開発を進める技術がある．この独自のモデリング言語のことを，ドメイン特化モデリング言語と呼ぶ．DSML を用いた開発では，まず DSML の開発者が，開発対象のアプリケーションドメインや技術ドメインに特化したモデリング言語と，そのドメインの共通部分をソースコード生成規則として，DSML を実装する．そしてプログラマが，その DSML を用いて,プラットフォームに依存しないモデルを作成,編集し，ソースコードはシステムによって自動生成される．DSML は，肥大化するソースコードに対して品質,効率を確保しなければいけない開発,責任者の間で注目が集まっている.

またDSMLには，それを使用するのに特殊な知識を必要としないという特徴がある．

次に今回の研究開発に使用したSaaS型モデル駆動開発ツールであるcloocaについて説明する．cloocaは当初SaaS型のExecutableUML[2-2]を入力とするモデル駆動開発ツールとして開発を進めてきた.現在はさらに機能を発展させてDSMLの開発支援ツールとなっている．

Webベースの IDEは，ブラウザ上で動作するために，ツールの導入が非常に簡単である．さらにWebの仕組みの上で開発されているために，ユーザ同士のコラボレーション機能を実装しやすく，実際にそのような機能を持っているものがほとんどである．しかし既存のどのサービスや技術も，ソースコードを対象にしていて，ソフトウェアモデルをうまく扱えている環境はなかった．一方で，既存のモデル駆動開発ツールやDSLツールは，どれもインストールが必要で，ブラウザ上でのモデルの確認や編集など，Web化はできていなかった．

そこで，Webベースの IDEと DSLツールの両方の利点を合わせ持つ，Webベースの DSLツールであるcloocaを開発し，上記の問題点を解決するに至った．

このcloocaをさらに発展させ，モデルとその編集履歴を収集する機能，及び，それらのデータを活用する機能を開発することで，本研究の目的を達成する．

2.1.3 研究目標

本研究課題の目標は，モデルを対象とした実証的研究に利用できる，モデルとその編集履歴などを継続的に集積し,集積したデータを利用するための環境を整備することである．この目標を達成するために，以下の4つの研究目標を定めた．

- 研究目標1「モデルとその編集履歴を集積できるようにツールを改変し,動作確認する」
  まず最初の段階としてモデルとその編集履歴を集積できるように開発を進めてきたツールであるcloocaを改変する．さらに，小規模なプロジェクトによりモデルとその編集履歴が集積できていることを確認する．
- 研究目標2「モデルの編集履歴を検索・利用するためのAPIを定義し，動作確認する」
  次にモデルとその編集履歴を活用するための環境整備をすべく，それらを検索・利用するためのライブラリを整備する．
- 研究目標3「講義，PBL等でツールを利用して，モデルとその編集履歴を集積する」
  また，集積したモデルとその編集履歴の活用方法を検討するために，研究目標1で実施したプロジェクトよりも大規模，多人数のプロジェクトにおいてデータを集積すべく，講義，PBL等でツールを利用しデータを収集する．
- 研究目標4「既存のコードを対象としたメトリクスがモデルでも有効に働くかどうか評価する」

研究目標3で集積したモデルとその編集履歴データを対象に,研究目標2で開発したAPIを活用して，集積したデータを対象とした分析が可能かどうかを確認する．

当初計画では,「実データを対象として，研究目標2で定義されたAPIでの有効性を確認する」という研究目標を設定したが，研究目標2と強く関連がある研究目標であるため，研究目標2に組み入れて，実施する．

2.2 研究の活動実績・経緯

(1) マイルストーン設定と活動実績概要

表 2-1 に活動実績を示す．項目毎に実績をオレンジの矢印で示している．
大きなマイルストーンとして，今回，以下の 5 項目を挙げた．

1) モデルの編集履歴を集積するためのツール改変
2) モデルの編集履歴を検索・利用するための API 定義と開発
3) 講義，PBL 等を対象とした評価実験
4) 実データを対象とした API の有効性評価
5) モデルを対象としたメトリクスの実装評価

1) モデルの編集履歴を集積するためのツール改変が達成できないと,その他のすべてのマイルストーンが達成できず，また，特に時間が必要となりそうな，3) 講義，PBL 等を対象とした評価実験の実施が危うくなるため，1)に関する研究，開発について，特に，編集履歴の収集機能のコアの部分最優先で行った．結果的には，編集履歴の収集機能のうち，モデルのマージ機能などの開発に時間を要したため全体の完成としては 10 月になったが，単純に履歴を収集するために必要な機能については 8 月までに開発を終えることができ，以降の研究に大きな影響を与えることは無かった．

2) モデルの編集履歴を検索・利用するための API 定義と開発，及び，対となる 4) 実データを対象とした API の有効性評価については，一部遅延があったもののおおむね順調に進捗した．

3) 講義，PBL 等を対象とした評価実験については，当初は本学大学院システム情報科学府における講義，PBL を想定していたが，徳山工業高等専門学校にご協力いただくことにより，講義についての当初の想定以上にデータを収集することができた．しかしながら，PBL においては，PBL 課題であった模型飛行船制御システム開発が頓挫したため，期待したほどの規模のデータを収集することはできなかった．プロジェクトが頓挫した理由は，飛行船を浮遊させるために必要なヘリウムが，製造工場のトラブルにより入手困難になったためである．

5) モデルを対象としたメトリクスの実装評価については，残念ながら前述したように PBL における設計成果物を利用することはできなかったが，講義における演習課題の設計成果物を利用し，順調に進めることができた．この成果の一部は情報処理学会 コンピュータと教育研究会 117 回研究発表会において発表を行った[2-3]．

表 2-1　活動実績

項番	項目	6月	7月	8月	9月	10月	11月	12月	1月
1	モデルの編集履歴を集積するためのツール改変								
2	編集履歴の収集機能の開発								
3	バグトラッキングシステムとの連携機能の開発								
4	小規模プロジェクトでの実証評価								
5	モデルの編集履歴を検索・利用するためのAPI定義と開発								
6	編集履歴の検索機能の開発								
7	編集履歴の可視化機能の開発								
8	モデルのメトリクス算出機能の開発								
9	小規模プロジェクトでの実証評価								
10	中間報告の準備								
11	成果物の取りまとめ								
12	デモ作成								
13	講義，PBL等を対象とした評価実験								
14	PBLでの利用実験								
15	モデル駆動開発特論での利用実験								
16	実データを対象としたAPIの有効性評価								
17	評価								
18	有効性評価のフィードバック								
19	モデルを対象としたメトリクスの実装評価								
20	プロダクトメトリクスの適用評価								
21	プロセスメトリクスの適用評価								
22	最終成果のとりまとめ								
23	成果報告書の構成案								
24	成果概要プレゼン資料								
25	成果報告書の作成								
26	実績報告書の作成								

(2) 研究管理概要

定期的に研究責任者は研究チームメンバが参加する進捗報告会を開催し，研究の進捗状況を確認するとともに，議論を行った．また，進捗報告会の議事録を記録した．この進捗報告会は長期休暇期間などを除き毎週一回，開催した．

(3) 利用機材等

今回の研究開発でのテスト，及び，講義，PBL 等を対象とした評価実験の際に使用するサーバを借用した．具体的には Amazon Web Service の Tokyo Region 上にサーバ用のインスタンスを準備し，運用を行った．比較的安価に押さえることができ，また，サーバ構築工数の削減にも役だった．

(4) 情報交換等

以下の 2 大学の教員との情報交換を実施した．

奈良先端科学技術大学院大学において，モデルリポジトリとバグトラッキングシステムとの連携方法について議論した．また，東洋大学において集積された設計成果物を利用したメトリクス評価の方法について議論を行った．

2.3 研究実施体制

研究実施体制図を図 2-2 に示す．また，研究責任者のプロフィールを示す．また，以下にそれぞれの研究チーム参画者のプロフィールを示す．

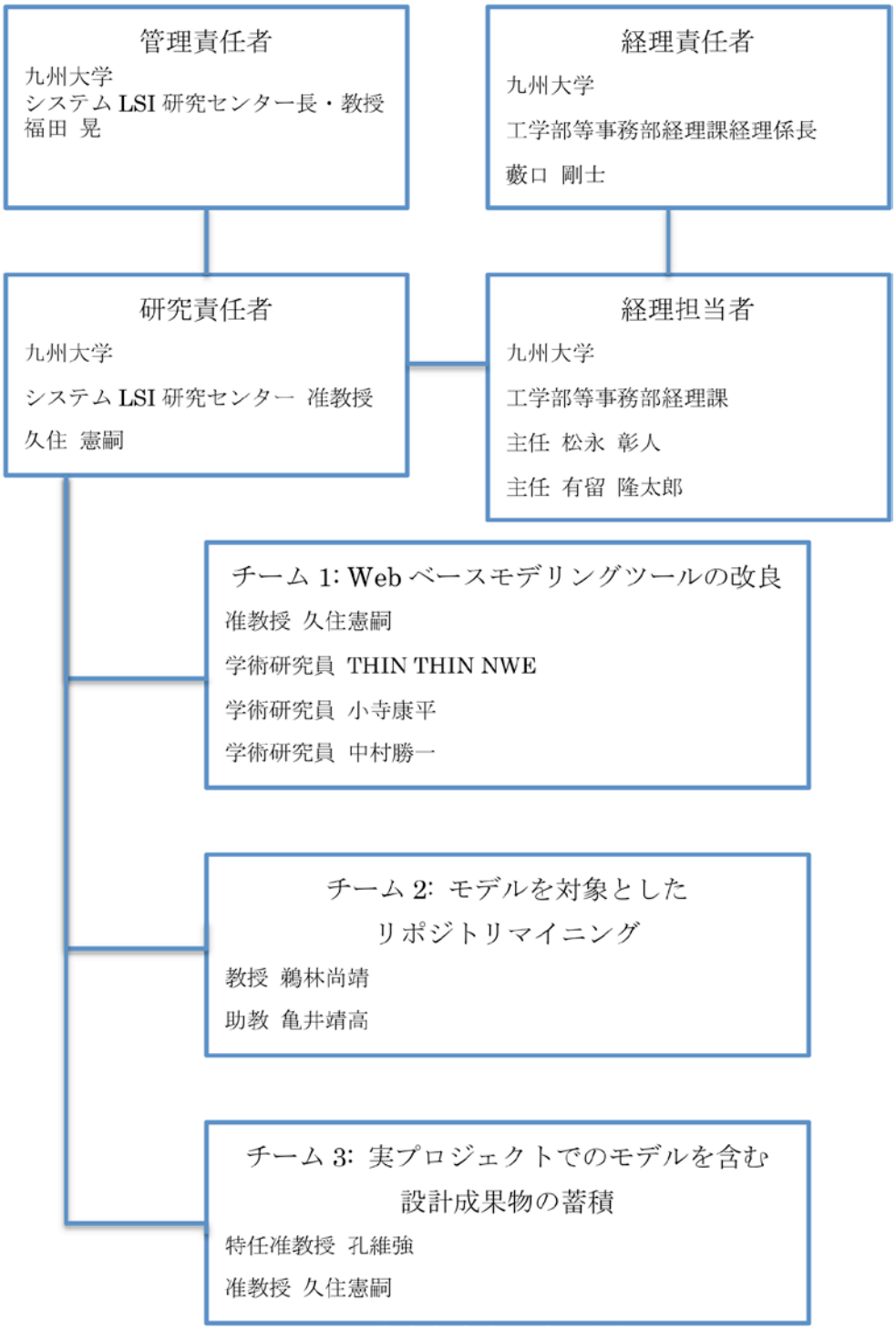

図 2-2　研究実施体制図

【研究責任者のプロフィール】

(ふりがな) 氏　名	ひさずみ けんじ 久住 憲嗣
生年月日	1977 年 1 月 30 日
所属機関	国立大学法人 九州大学
所属・役職	システム LSI 研究センター・准教授
住　所	〒819-0395 福岡市西区元岡７４４番地 システムＬＳＩ研究センター
TEL	092-802-3635
E-mail	nel@slrc.kyushu-u.ac.jp

【学歴（大学卒業以降)】
1999 年 3 月 信州大学工学部情報工学科 卒業
2001 年 3 月 奈良先端科学技術大学院大学 情報科学研究科 博士前期課程 修了
2004 年 3 月 九州大学大学院 システム情報科学府 情報工学専攻 博士後期課程 修了

【職歴】
2004 年 4 月～2005 年 9 月 科学技術振興機構 研究員
2005 年 10 月～2008 年 6 月 九州大学システム LSI 研究センター 特任講師
2008 年 7 月～現在 九州大学システム LSI 研究センター 准教授

【研究実績】
現在，組込みソフトウェアを対象としたモデル駆動開発，ドメイン特化型開発，ソフトウェアプロダクトライン開発方法論の研究に従事している．2009 年からは通信機器メーカとともにドメイン特化型開発の導入に関する共同研究を，また OA 機器メーカとともにモデル駆動開発に関する共同研究を実施している．この成果は組込みシステムシンポジウム 2011 において最優秀報告賞を，第 13 回組込みシステム技術に関するサマーワークショップ（SWEST13）においてベストポスター賞を受賞した．さらに，情報処理学会 九州支部 火の国情報シンポジウム 2009 において情報処理学会九州支部 2008 年度奨励賞を受賞し，指導学生が第 144 回 SLDM 研究会 優秀発表学生賞を受賞した．また，システム LSI 設計人材養成実践プログラムにおいて，システム LSI 及び組込みソフトウェアに関する教材開発，教育を実施している．また，組込みシステム開発とシステム LSI 設計を体験するための演習教材を開発した．この成果は組込みシステムシンポジウム 2009 において，実践報告 優秀賞を受賞した．

【主な論文・著書】

1) 福田哲志，末安史親，庭木勝也，森田健治，久住憲嗣，峯恒憲，鵜林尚靖，平山雅之，濱田直樹，二上貴夫「飛行船自動航行システム開発における SysML を用いたモデル駆動開発事例」，情報処理学会 組込みシステムシンポジウム 2011（ESS2011），2011 年 10 月．［査読付き］
2) 赤山聖子，久保秋真，久住憲嗣，二上貴夫，北須賀輝明「ソフトウェア初学者へのモデリング教育における MDD の活用」，情報処理学会 組込みシステムシンポジウム 2011（ESS2011），2011 年 10 月．［査読付き］
3) 大塚潤，河原畑光一，岩崎孝司，内場誠，中西恒夫，久住憲嗣，福田晃「大規模移動体ネットワーク機器ファームウェア開発へのソフトウェアプロダクトライン適用事例」，情報処理学会 組込みシステムシンポジウム 2011（ESS2011），2011 年 10 月．［査読付き，**実践報告 優秀賞**］
4) Jun Otsuka, Kouichi Kawarabata, Takashi Iwasaki, Makoto Uchiba, Tsuneo Nakanishi, Kenji Hisazumi, and Akira Fukuda: Small Inexpensive Core Asset Construction for Large Gainful Product Line Development, Proc. of 3rd International Workshop on Model-driven Approaches in Software Product Line Engineering and 3rd Workshop on Scalable Modeling Techniques for Software Product Lines, 2011 年 8 月.
5) 林田 隆則，久住 憲嗣，ヴィクトル グラール，築添 明，福田 晃，中西 恒夫，安浦 寛人 「多機能電話を題材としたシステム LSI 設計実習」，情報処理学会 組込みシステムシンポジウム 2009，論文集，pp. 13-18，2009 年 10 月.［査読付き，**実践報告 優秀賞**］
6) Wayne Wolf（著），中西 恒夫，北須賀 輝明，久住 憲嗣，室山 真徳，田頭 茂明（翻訳），「組込みシステム設計の基礎」，日経 BP 社，2009 年 3 月.

Web ベースモデリングツールの改良

中村 勝一 博士(工学) 九州工業大学
　九州大学 システム LSI 研究センター 学術研究員
　九州工業大学 情報工学研究科博士課程 情報システム専攻 修了
　ソフトウェア工学，及び，センサネットワークとその応用技術などを専門とする．

THIN THIN NWE 博士(工学) 長岡技術科学大学
　九州大学 システム LSI 研究センター学術研究員
　長岡技術科学大学大学院工学研究科博士課程 エネルギー・環境工学専攻 修了
　ソフトウェア工学，インターネットサービスなどを専門とする．

小寺 康平 博士(工学) 九州大学
　九州大学 システム LSI 研究センター学術研究員
　九州大学大学院システム情報科学府　情報知能工学専攻　博士課程後期修了
　無線ネットワーク，ソフトウェア工学などを専門とする．

以上に加えて，開発補助を行うアルバイトを雇用する．

モデルを対象としたリポジトリマイニング

鵜林 尚靖　博士(学術) 東京大学
　九州大学 大学院システム情報科学研究院 情報知能工学部門 教授
　プログラミング言語機構，モデリング言語機構，及び，形式手法などを専門とする．

亀井 靖高 博士(工学) 奈良先端科学技術大学院大学
　九州大学 大学院システム情報科学研究院 情報知能工学部門 教授
　ソフトウェア工学，特に，リポジトリマイニングを専門とする．

以上に加えて，データ分析の補助を行うアルバイトを雇用する．

実プロジェクトにおけるモデルを含む設計成果物の蓄積

孔 維強 博士(工学) 北陸先端科学技術大学院大学
　九州大学 大学院システム情報科学研究院 情報知能工学部門 特任准教授
　ソフトウェア工学，ソフトウェア工学教育などを専門とする．

以上に加えて，改変したツールを用いて開発を実施しデータ収集に協力するアルバイトを雇用する．

３．研究成果

3.1 研究目標 1「モデルとその編集履歴を集積できるようにツールを改変し，動作確認する」

3.1.1 当初の想定

(1) 想定する仮説等

既存のモデリングツールはモデルの履歴を保存する機能が無い．また，既存のリポジトリシステムにモデルを単純に登録するだけでは，モデルの履歴に対して横断的に検索をかけることが困難である．モデルとその編集履歴を収集できるツールの開発が必要不可欠である．

(2) 当初の到達目標

モデルとその編集履歴を蓄積すべくツールの改変を行う，そして小規模なプロジェクトにおいて，モデルとその編集履歴が蓄積できることを検証する．具体的には，Web ベースのモデリングを可能とする MDD ツールである clooca に対して,モデルとその編集履歴を集積可能とするツールとして改変する．そして改変したツールを動作検証向けに準備した小規模なプロジェクトにおいて，モデルとその編集履歴が蓄積できていることを検証する．

(3)当初の期待される効果

モデルとその編集履歴，及び，バグ情報などを，ツールの支援のもとに収集できるようになるため，clooca を使用している開発現場において定量的手法を用いて，開発を支援することができるようになる．

3.1.2 研究プロセスと成果

(1) 研究プロセス

「モデルとその編集履歴を蓄積するためのツール改変の研究開発においては，モデルとその編集履歴の集積項目に抜け漏れが発生し，研究目標 2 以降で必要なデータが集積できないという事態の発生が課題として挙げられる．これを防ぐことを目的とし，研究開発の準備の段階で調査に基づき，蓄積項目の整理を行い，事前に検討および設計を実施してツールの改変実装を行った.」さらに，小規模なプロジェクトにおいて実証評価を行い，リポジトリマイニングに必要なメトリクスを計算するために必要な蓄積項目が取得できているかを確認した．以下に，実施した作業項目を示す．

- 編集履歴の収集機能の開発
- バグトラッキングシステムとの連携機能の開発
- 小規模プロジェクトでの実証評価

① 開発について

Web ベースのモデリングを可能とする MDD ツールである clooca を改変することで，モデルとその編集履歴を集積するツールとする．clooca は SaaS 型のツールであり，Web ブラウザ上で動作する．clooca は SaaS 型であるため，すべての描かれたモデルはサーバ側に蓄積されるアーキテクチャとなっている．「編集履歴の収集機能の開発」においては，既存の clooca にモデル編集履歴の保存機能を導入し，モデルリポジトリシステムとして改変実装を行った．その際に，事前の入念な調査や検討に基づいて，改変実装を行った．

「バグトラッキングシステムとの連携機能の開発」においては，モデルのリポジトリへのコミットとバグトラッキングシステムのチケットとの対応付けを行う連携機能の実装を行った．この連携機能によりモデルのリポジトリ情報に対応したバグ情報の追跡を可能とした．

② 検証について

改変実装を実施したモデルリポジトリシステムに対して，モデルとその編集履歴を集積できることを明らかにするため，「小規模プロジェクトでの実証評価」を行った．表 3-1-1 に，実施した検証項目を示す．

表 3-1-1　検証項目

大項目	小項目	説明
リポジトリによるモデルの編集履歴の収集	モデルの編集履歴の保存	モデルをリポジトリに保存できる
	モデルの編集履歴閲覧	リポジトリに保存したモデルが確認できる
	編集履歴から任意のバージョンへの移行(リバート)	リポジトリに保存したモデルをワークスペースに反映できる
複数人によるリポジトリの利用	チェックアウト	リポジトリ上のモデルをワークスペースに反映できる
	コミット	ワークスペース上のモデルをリポジトリに保存できる
	アップデート	リポジトリに保存されているモデルによって，現在作業中のワークスペース上のモデルを更新する
複数人による円滑なプロジェクト運用	コンフリクトの検出	複数人で同時に同じモデルを操作し，コミット時に競合(コンフリクト)が起きたことが確認できる
	マージ	複数人で，同じモデルに独立した別々の操作を行い，コミットする．その後，アップデートし両方の操作の意図が損なわれることなく，モデルに反映できる

次に，実証実験を行った小規模プロジェクトについて説明する．少人数で小規模なプロジェクト(テストプロジェクト)を実施した．モデルリポジトリシステムを用いて図 3-1-1 に示す NXT[3-1]の動作設計を行う(図 3-1-2)．NXT の動作設計を行うことで NXT 設計モデルの編集履歴がリポジトリに保存され，編集履歴を複数人で利用しながら，NXT の動作設計が可能であることを確認した．さらに，設計したモデルの実装コードを NXT に実装し動作確認も行った．

図 3-1-1　検証評価に用いた NXT

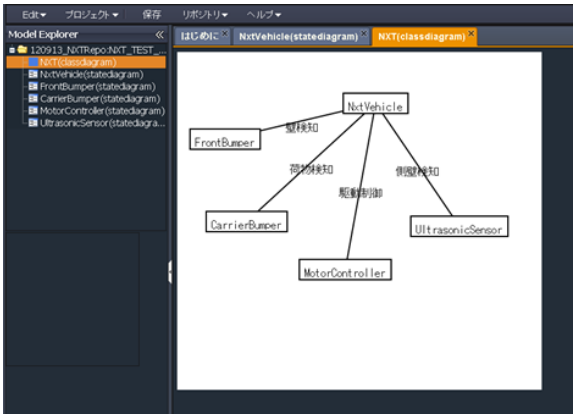

図 3-1-2　NXT の動作設計におけるモデルリポジトリシステム画面例

(2) 具体的な研究成果の内容

「編集履歴の収集機能の開発」においては，既存の clooca にモデル編集履歴の保存機能を導入し，モデルリポジトリシステムとして改変実装を行った．その際に，事前の入念な調査や検討に基づいて，改変実装を行った．また「バグトラッキングシステムとの連携機能の開発」においては，モデルのリポジトリへのコミットとバグトラッキングシステムのチケットとの対応付けを行う連携機能の実装を行った．この連携機能によりモデルのリポジトリ情報に対応したバグ情報の追跡を可能とした．以下に，具体的な研究成果として開発仕様および設計，改造実装したモデルリポジトリシステムを示す．

① 開発仕様および設計について

既存の clooca を改変して実装するモデルリポジトリシステムの概要図を図 3-1-3 に示し，既存の clooca の仕様について説明する．

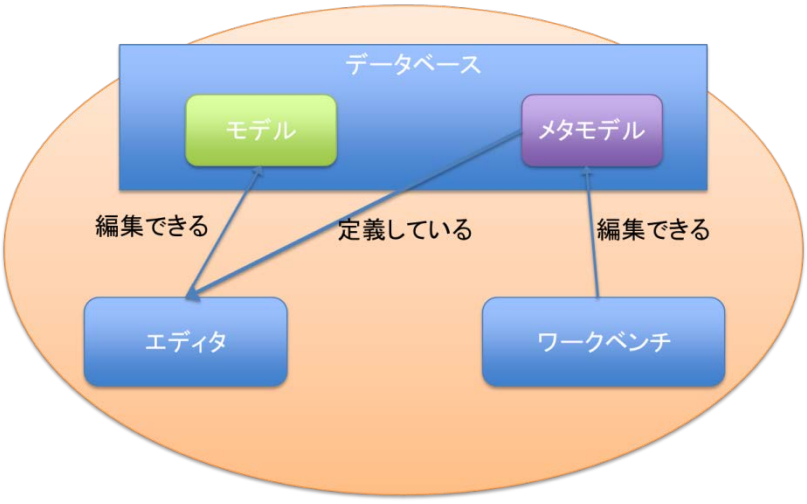

図 3-1-3　モデルリポジトリシステム概要図

既存の clooca の開発仕様は以下の通りである.

- Web ベースの DSL ツール
  - ワークベンチ
    - DSML の定義
  - エディタ
    - ダイアグラムエディタ
- プログラム規模
  - クライアント
    - javascript+ExtJS で実装され約 5,000 ステップ
  - サーバ
    - Python+flask で実装され約 3,800 ステップ

モデルリポジトリシステムの基本動作を図 3-1-4 に示す.

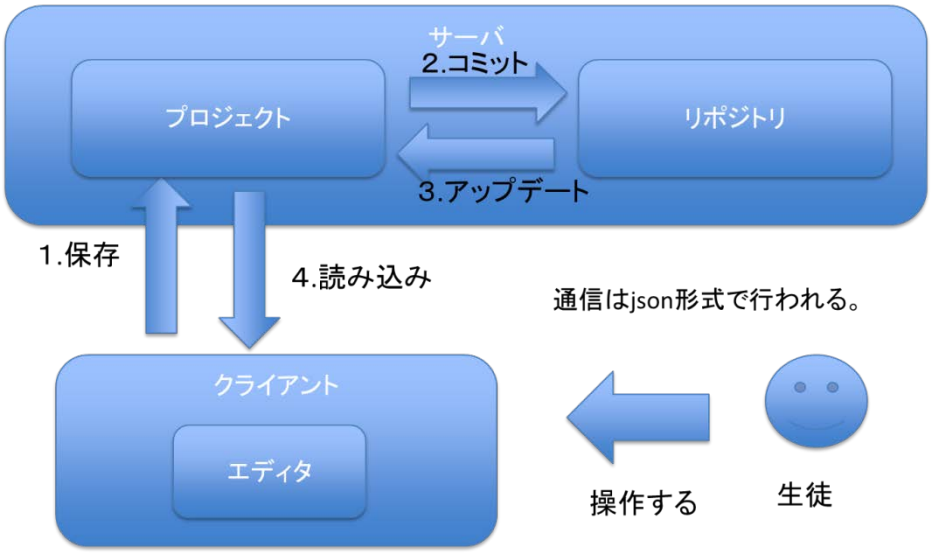

図 3-1-4　基本動作

次に，Json で定義されるモデル構造について図 3-1-5 に示す．1 つのモデルは，以下に挙げた単体または複数のモデル構成要素からなる.

- ダイアグラム
  - クラス図，ステートマシン図など，オブジェクトとリレーションで表現されたモデル図
- オブジェクト
  - ダイアグラムを構成する要素の一つ
- リレーション
  - ダイアグラムを構成する要素の一つ

オブジェクト同士の関係性を定義する

- プロパティ
  - ダイアグラム，オブジェクト，リレーションに付随する情報

さらに各々のモデル構成要素には以下の情報が付随する．

- id
  - 個別のダイアグラム，オブジェクト，リレーション，プロパティを識別するための識別子
- 情報
  - ダイアグラム，オブジェクト，リレーション，プロパティのステータスを表す情報 追加・削除・変更の各ステータスを識別する「ver type」を定義する
- 関連
  - ダイアグラム，オブジェクト，リレーション，プロパティの関連情報を示す
- バージョン情報
  - 属しているバージョンの情報

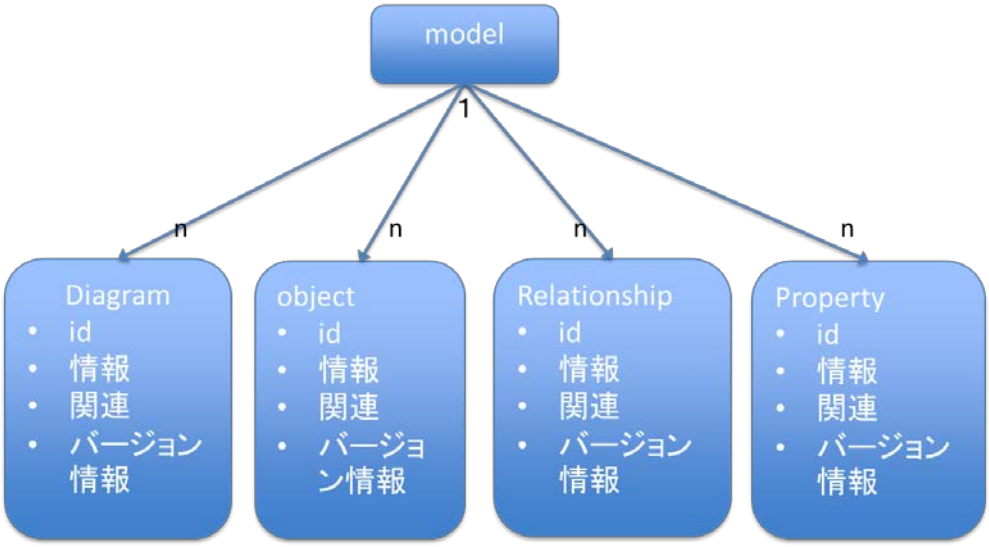

図 3-1-5　モデル構造

次にモデルリポジトリとモデルの関係図を図 3-1-6 に示す．モデルの編集履歴を収集することを目的としているため，一つのモデルは一つのリポジトリに保存され，またリポジトリは一つのモデルのみ保存するものとする．

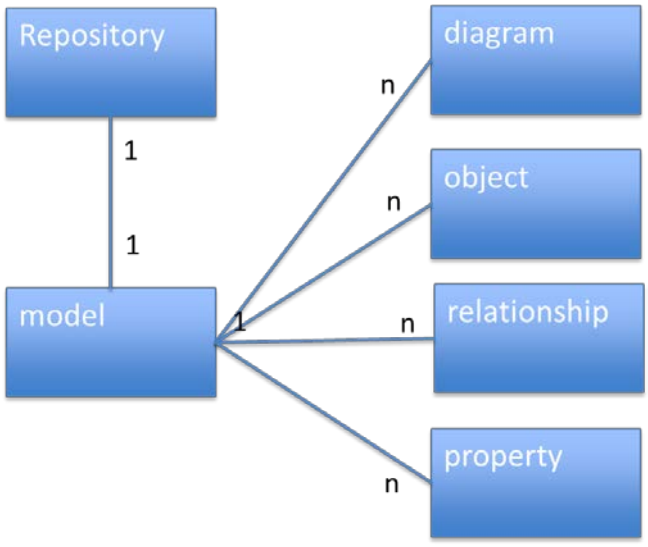

図 3-1-6　モデルリポジトリとモデルの関係図

UML などのグラフィカル形式のモデルの変更履歴を対象として,データマイニングをし，ソフトウェア開発に有用な情報を抽出できるようにすることが本研究の改変目的である．ソースコードを対象とした研究では，ファイルごとの変更履歴をもとに，ファイルごとの変更回数等に基づいて，バグの発生し易さを推定できることが知られている．本研究では，モデルを対象にして，より木目細かい粒度で，同様のことを設計し実装した.尚,モデリングツールとしては Web ベース clooca を採用するものとした.

- リポジトリ機能

  モデルとその変更履歴を集中して管理する．Subversion，git 等のソースコードを対象としたバージョン管理システム(VCS；Version Control System) のモデル版である．リポジトリに対する基本的な操作は，VCS に準ずるものとした．

  ただし，一般的な VCS がファイル単位で管理するのに対し，clooca ではモデル要素単位で管理する．たとえばクラス図であれば，クラス図全体を 1 要素として管理するのではなく，クラス図の要素であるクラス，操作，属性などの変更が追跡できるように設計した．

  - インポート機能
    - ✓ 概要
      - ✧ 現在編集中のワークスペース上のモデルを，リポジトリに新規に格納する
    - ✓ 入力
      - ✧ プロジェクト名
    - ✓ 正常
      - ✧ ユーザが clooca クライアントにインポートを指示すると，clooca クライアントは現在作業中のワークスペース上のすべてのデータをサーバに保存する．clooca サーバは指定された名前のプロジェクトをリポジトリ上に作成し，現在作業中のワークスペース上のすべてのデータをリポジトリに保存する．
    - ✓ 同名のプロジェクトがすでにリポジトリにある場合
      - ✧ エラーを出力し，名前の変更を促す．

  - チェックアウト機能
    - ✓ 概要
      - ✧ リポジトリに格納されているモデルを，現在作業中のワークスペース上に展開する．
    - ✓ 入力
      - ✧ モデル名
    - ✓ 正常
      - ✧ ユーザが clooca クライアントにチェックアウトを指示すると，clooca サーバは指定された名前のプロジェクトをリポジトリから取得し，現在作業中のワークスペースに取り込む．取り込む際に，ワークスペース上にリポジトリ上のどのモデルを取り込んだのかを記録する．取り込み終了後，clooca クライアントはモデルを取得し直し，再描画する．
    - ✓ コンフリクト時
      - ✧ 現在作業中のワークスペースに，すでにモデルがある場合には，エラーを出力して終了する．

- コミット機能
  - ✓ 概要
    - ✧ 現在作業中のモデルをリポジトリに格納する.
  - ✓ 入力
    - ✧ コミットメッセージ
      - ・ バグ管理システムと連携して, バグ情報とのトレーサビリティをとることを目的とするため, チケット番号とチケット内容に関するコメントをメッセージとする.
  - ✓ 正常系
    - ✧ ユーザが clooca クライアントにコミットを指示すると，clooca クライアントは現在作業中のワークスペース上のすべてのデータをサーバに保存する．clooca サーバは現在作業中のワークスペース上のモデルをリポジトリに格納する.
  - ✓ コンフリクト時
    - ✧ 矛盾したコミットの場合, すなわち, コミットしようとしたプロジェクトが，ほかの開発者によってすでに書き換えられてしまっている場合には, エラーを出力して終了する.

- アップデート機能
  - ✓ 概要
    - ✧ リポジトリに格納されているモデルによって, 現在作業中のワークスペース上のモデルを更新する.
  - ✓ 入力
    - ✧ リビジョン番号: オプショナルであり指定されなかった場合には最新版とする
  - ✓ 正常系
    - ✧ ユーザが clooca クライアントにアップデートを指示すると，clooca クライアントは現在作業中のワークスペース上のすべてのデータをサーバに保存する．clooca サーバはリポジトリに格納されている，指定されたリビジョンのモデルによって, 現在作業中のワークスペース上のモデルを更新する.

- リバート機能
  - ✓ 概要
    - ✧ リポジトリに格納されているモデルによって, 現在作業中のワークスペース上のモデルを強制的に更新する.
  - ✓ 入力
    - ✧ リビジョン: オプショナルであり指定されなかった場合には最新版とする
  - ✓ 正常系

- ユーザが clooca クライアントにリバートを指示すると，現在作業中のワークスペース上のすべてのデータを削除する．clooca サーバはリポジトリに格納されている，指定されたリビジョンのモデルによって，現在作業中のワークスペース上のモデルを更新する．

- マージ UI を考慮したアップデート機能
  - 概要
    - リポジトリ指定リビジョンをワークスペースに反映する．マージ UI を考慮したワークスペースを図 3-1-7 に示す．ワークスペースとリポジトリ指定リビジョンに差分が存在する場合はマージする手段を提供するものとする．マージ完了したものはワークスペース上で，指定リビジョンのモデルとして取り扱い，その後コミットできる設計とする．ただし，指定リビジョンで最新以前のリビジョンをアップデートした場合は，コミット時に再度コンフリクトが発生することになることを制約条件とする．

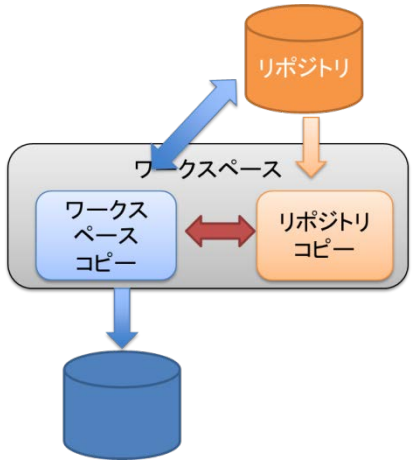

図 3-1-7　マージ UI を考慮したワークスペース

表 3-1-2 に本研究開発で改変した機能一覧と機能概要を示す．

表 3-1-2　機能一覧と機能概要

機能(大項目)	機能(小項目)	機能概要
リポジトリによるモデルの編集履歴の収集	モデルの編集履歴の保存	モデルをリポジトリに保存する
	モデルの編集履歴閲覧	リポジトリに保存したモデルを確認する
	編集履歴から任意のバージョンへの移行(リバート)	リポジトリに保存したモデルをワークスペースに反映する
複数人によるリポジトリの利用	チェックアウト	リポジトリ上のモデルをワークスペースに反映する
	コミット	ワークスペース上のモデルをリポジトリに保存する
	アップデート	リポジトリに保存されているモデルによって，現在作業中のワークスペース上のモデルを更新する
複数人による円滑なプロジェクト運用	コンフリクトの検出	複数人で同時に同じモデルを操作し，コミット時に競合が起きたことが確認できる
	マージ	複数人で，同じモデルに独立した別々の操作を行い，コミットする．その後，アップデートし両方の操作の意図を損なうことなく，モデルに反映する

また，「バグトラッキングシステムとの連携機能の開発」として，モデルリポジトリシステムのコミット機能にメッセージ付加機能を設けた．メッセージ付加機能を活用したバグトラッキングシステムとの連携について図 3-1-8 に示す．モデルリポジトリシステムにおいて，ワークスペースのモデルのコミットの際に，バグ追跡のためのバグチケット番号をメッセージとして書き込む．この際に書き込んだバグチケット番号をバグトラッキングシステムに登録されているバグチケット番号に対応させる．これにより，モデルリポジトリシステムのコミットログに含まれるバグ情報の追跡を可能とした．

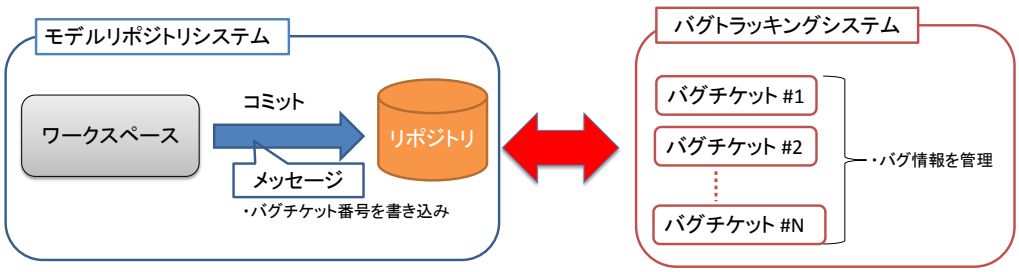

図 3-1-8　バグトラッキングシステムとの連携機能

② 開発したモデルリポジトリシステムの検証評価について

改変実装を実施したモデルリポジトリシステムに対して，モデルとその編集履歴を集

積できることを明らかにするため,「小規模プロジェクトでの実証評価」を行った．以下に具体的な成果として実証評価を示す.

モデルリポジトリシステムを用いて,少人数で小規模なプロジェクト(テストプロジェクト)を実施した.図 3-1-1 に示した NXT において表 3-1-3 に示すテストプロジェクト要求仕様に基づき NXT の動作設計を行うものとした．モデルリポジトリシステムで作成した NXT のクラス設計モデルを図 3-1-9 に,ステートマシン設計モデルを図 3-1-10 に示す.これら設計モデルの編集履歴がリポジトリに保存されることを確認した．さらに，編集履歴を複数人で利用可能であり，複数人が共同作業の一環として設計モデルを更新しながら NXT の動作設計が可能であることも確認した．以上の「小規模プロジェクトでの実証評価」において動作確認テストを行い，正常動作を確認したモデルリポジトリシステム要求機能の項目を表 3-1-4 に示す.

表 3-1-3　テストプロジェクト要求仕様

要求	機能概要	機能詳細
荷物を運搬する	停止する	モーターを停止させる
	ライントレースする(運搬時)	光センサの値を取得する
		モーターを動作させる
	発進合図を検知する	バンパーのイベントを検知する
	荷物の有無を検知する	バンパーのイベントを検知する
	側壁を認識する	超音波の値を取得する
荷物を転送する	ライントレースする(転送時)	光センサの値を取得する
		モーターを動作させる
	転送先到着を検知する	バンパーのイベントを検知する
回送する	車庫に入る	バンパーのイベントを検知する
	反転する	モーターを動作させる
	ライントレースする(回送時)	光センサの値を取得する
		モーターを動作させる

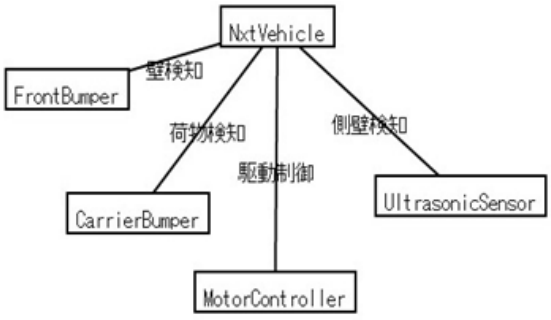

図 3-1-9　NXT のクラス設計モデル

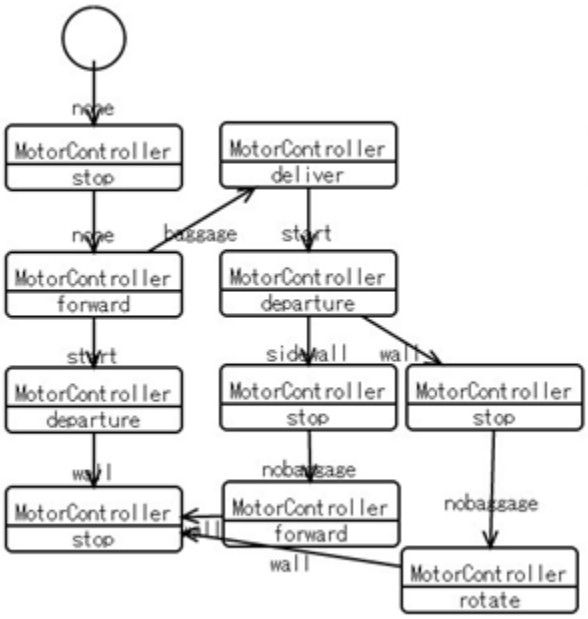

図 3-1-10　NXT ステートマシン図

表 3-1-4　要求機能テスト項目

大テスト項目	小テスト項目	テストの確認内容
リポジトリによるモデルの編集履歴の収集	モデルの編集履歴の保存	モデルをリポジトリに保存できることを確認
	モデルの編集履歴閲覧	リポジトリに保存したモデルが確認できることを確認
	編集履歴から任意のバージョンへの移行(リバート)	リポジトリに保存したモデルをワークスペースに反映できることを確認
複数人によるリポジトリの利用	チェックアウト	リポジトリ上のモデルをワークスペースに反映できることを確認
	コミット	ワークスペース上のモデルをリポジトリに保存できることを確認
	アップデート	リポジトリに保存されているモデルによって，現在作業中のワークスペース上のモデルを更新できることを確認
複数人による円滑なプロジェクト運用	コンフリクトの検出	複数人で同時に同じモデルを操作し，コミット時に競合が起きたことが確認できることを確認
	マージ	複数人で，同じモデルに独立した別々の操作を行い，コミットする．その後，アップデートし両方の操作の意図が損なわれることなく，モデルに反映できることを確認

最後に，モデルリポジトリシステムにて設計したモデルの実装コードを NXT に実装し，テストプロジェクト要求仕様を満足したシステムを開発できた事を，表 3-1-5 に示すテストプロジェクトの動作試験項目に従い確認した．試験結果は全て正常であった．

表 3-1-5　テストプロジェクトのテスト項目

システム要件		機能		動作カテゴリ	確認内容	結果	備考
No.	要件カテゴリ	No.	機能項目				
1	運搬業務	1-1	起動直後、NXTを待機(停止)させ荷物を受け取るのを待つ	正常	NXT起動直後、何もしなければNXTは停止したままとなる	OK	想定通りの動作
		1-2	NXT起動直後、荷物を受け取り、フロントバンパーが押されたら運搬開始	正常	NXT起動直後、荷台ボタンが押された状態で、フロントバンパーを押すと、NXTがライントレースを始める	OK	想定通りの動作
		1-3	超音波で側壁を認識したときは、停止して荷物が降ろされるのを待つ	正常	NXT運搬走行中、側壁を横切るとNXTが停止する	OK	想定通りの動作
		1-4	側壁がなければ止まらず運搬継続	正常	NXT運搬走行中、側壁を通過しなければNXTはライントレースを継続	OK	想定通りの動作
		1-5	側壁ありで運搬中に、荷台から荷物が下ろされたら、回送開始	正常	NXT運搬走行中に、側壁を感知した後、荷台ボタンを離すと、 NXT本体が回送走行を始める	OK	想定通りの動作
		1-6	壁にぶつかったら転送先到着、荷物が降ろされるのを待つ	正常	NXTが運搬走行中、フロントバンパーが押されるとNXTが停止する	OK	想定通りの動作
		1-7	荷物が下ろされたら反転して回送開始	正常	NXT運搬走行中に、壁にぶつかったあと、荷台ボタンを離すと、 NXT本体が反転し、回送走行を始める	OK	想定通りの動作
2	回送業務	2-1	NXT起動直後、荷物なしに押されたら回送開始	正常	NXT起動直後、荷台ボタンが押されていない状態で、フロントバンパーを押すと、NXTがライントレースを始める	OK	想定通りの動作
		2-2	荷物なしなら側壁有無によらず、止まらず回送継続	正常	NXT回送走行中、側壁を横切ってもNXTはライントレースをつづける	OK	想定通りの動作
		2-3	回送中に分岐点を発見したら、直進する	正常	NXTがグレーマーカーに接触しても、そのまま進路を変えず、ライントレースを続ける	OK	想定通りの動作
		2-4	回送走行中、壁にぶつかったら車庫に入ったとみなす	正常	NXT回送走行中、壁にぶつかる(フロントバンパーを押す)とNXTは停止する 以後、停止状態のまま	OK	想定通りの動作
		2-5	運搬済みで回送後、壁にぶつかったら車庫に入ったとみなす	正常	NXT回送走行中、壁にぶつかる(フロントバンパーを押す)とNXTは停止する 以後、停止状態のまま	OK	想定通りの動作
		2-6	転送済みで回送後、壁にぶつかったら車庫に入ったとみなす	正常	NXT回送走行中、壁にぶつかる(フロントバンパーを押す)とNXTは停止する 以後、停止状態のまま	OK	想定通りの動作

3.1.3 実用化へ向けた課題と問題点

(1) 課題と問題点

当初の到達目標，期待される効果で述べた，モデルとその編集履歴を蓄積すべくツールの改変を行い，小規模なプロジェクトにおいて，モデルとその編集履歴が蓄積できることを検証した．本研究成果を実用化する際の課題と問題点について検討を行い，以下に示す三つの観点で言及する．

- システムUIブラッシュアップ
- 分散協調開発への対応強化
- 開発管理ツールとの有機的連携

① システムUIブラッシュアップ

不適切なモデルに対してソースコードダウンロード（コード生成）を実施した際，システムのエラーハンドリングが不十分なためServer Errorを表示してしまうという問題がある．この問題はユーザにとって作成したモデルに不備があるのか，システムの不具合かを切り分けすることが難しく，ユーザビリティを下げる要因となる．

実用化に向けてテンプレート機能を強化することにより，不適切なモデルのハンドリングを可能とすることによってユーザビリティを図ることができる．さらに，エラーメッセージと合わせてモデルの問題点を指摘し，モデル開発の教育効果を高めることも期待できる．

② 分散協調開発への対応強化

モデルリポジトリシステムには一般的なバージョン管理システムが保有するブランチ機能を持たない．実用化に向けてブランチ機能の実現により，大人数・大規模な協調開発や，製品バージョンと開発バージョン等を別ブランチとして取り扱うことが可能な中長期開発への応用を可能とする．

分散協調開発の観点では，もう一つ，exportしたプロジェクトをインポートした際に，システム上の不具合が発生する可能性がある．これは設計したモデルにおけるノードの内部表現に用いるIDの採番方式が，複数人によるリポジトリの共有利用に対して充分な対応ができてないことに起因する．実用化に向けて，製品レベルまで本研究開発成果を向上させる際に，IDの採番方式を見直すことにより，exportしたプロジェクトをインポートする操作を行った時でもシステムを安定動作可能とすることができる．この取り組みを実施することで，ツールの流通のみならずツール上で開発されたモデルの流通をも可能にすることができ，製品化した際のセールポイントとなる事が期待できる．

③ 開発管理ツールとの有機的連携

モデルリポジトリシステムはRedmine等の開発管理ツールとの連携機能を持っているが，連携機能を有効に利用するにはユーザの手動入力を含む運用ポリシーの厳格な適用が必須となっている．これは，連携の自動化および，連携を補助する機能の追加により，ユーザが運用ポリシーを意識することなく開発管理ツールとの連携を図れるようにする．実用化に向けて，モデルの編集履歴機能を最大限活用するためには，開発管理ツールとの連携方法を，想定するビジネス分野，製品構成等を検討することで市場の獲得や拡大が大いに期待できる．

(2) 将来の応用方法

モデルとその編集履歴を蓄積すべくツールの改変を行い，小規模なプロジェクトにおいて，モデルとその編集履歴が蓄積できることを検証した．本研究成果の将来の応用方法について説明する．

開発現場において定量的手法を導入することができ，多種多様な開発の支援が可能となると想定される．モデル開発においてテンプレート機能を強化する事により，比較的経験の浅いモデル開発者が作成したモデルの問題点をツールがアドバイスし，モデル開発の工程を行う際に，開発者の教育，スキル向上を行う教育ツールの応用事例も考えられる．

ブランチ機能を備えることで中長期の分散協調開発，プロダクトライン開発への応用も期待できると考えられる．さらに，exportしたプロジェクトをインポートする機能が実現できると，ツールの流通のみならずツール上で開発されたモデルの流通をも可能にすることができ，新しいモデルモデルとしてモデル流通プラットフォーム，モデル部品を活用した新規ビジネスへの応用も考えられる．

3.2 研究目標 2「モデルの編集履歴を検索・利用するための API を定義し，動作確認する」

3.2.1 当初の想定

(1) 想定する仮説等

第三者が利用可能なモデルを含んだ設計成果物が不足しているため，これらを大局的に分析して開発を支援する手法が十分に発達していない．設計成果物のメトリクスを計測して，大局的視点から開発を支援する手法が必要である．

(2) 当初の到達目標

中間目標 1 の改変で集積できるデータに対して検索し，必要なメトリクスを計算できる API を定義する．さらにその API が利用できるかどうかを確認する．具体的には，研究目標 1 で開発したモデルリポジトリシステムにより集積したモデルとその編集履歴のデータに対して検索し，リポジトリマイニングに必要なメトリクスを定量的に計算できる API を定義する．さらにその API が利用できるかどうかを確認する．

(3) 当初の期待される効果

すでに提案されているコードを対象としたメトリクスが，モデルにおいても有効に働くかどうかを検証できる手法を確立できる．さらに，モデルにおいて既存のコードを対象としたメトリクスが有効に働かない応用や場面においても，有効に働くモデル特有のメトリクスを検証できる手法を確立できる．

3.2.2 研究プロセスと成果

(1) 研究プロセス

予備的な研究として，リポジトリマイニングのためのドメイン特化言語(DSL)の設計，開発を行ってきた．モデルの編集履歴を検索・利用するための API の定義においては，DSL を設計する上で整理した情報を活用することで，利用しにくい API とならないよう留意し API を検討・定義した．また，現在，継続的に実施しているリポジトリマイニングのサーベイ結果をもとに，リポジトリマイニングに必要な API の整理を行い，API の設計および，実装を行った．以下に，実施した作業項目を示す．

- 編集履歴の検索機能の開発
- 編集履歴の可視化機能の開発
- モデルのメトリクス算出機能の開発
- 小規模プロジェクトでの実証評価

① 開発について

Web ベースのモデリングを可能とする MDD ツールである clooca を改変することで，モデルとその編集履歴を集積するツールとする．clooca は SaaS 型のツールであり，Web ブラウ

ザ上で動作する．clooca は SaaS 型であるため，すべての描かれたモデルはサーバ側に蓄積されるアーキテクチャとなっている．「編集履歴の検索機能の開発」，「編集履歴の可視化機能の開発」，「モデルのメトリクス算出機能の開発」においては，保存された膨大なモデルデータに対して検索を実行して，その結果を可視化するとともに，モデルのメトリクスも算出し可視化する API を定義，実装した．

② 検証について

メトリクスを計算するために「小規模プロジェクトでの実証評価」で集積したデータに対し，API によるメトリクス計算を行った．API のメトリクス計算結果がデータのメトリクスの想定値と一致するかの検証を行った．

(2) 具体的な研究成果の内容

「編集履歴の検索機能の開発」，「編集履歴の可視化機能の開発」，「モデルのメトリクス算出機能の開発」においては，保存された膨大なモデルデータに対して検索を実行して，その結果を可視化するとともに，モデルのメトリクスも算出し可視化する API を定義，実装した．以下に，具体的な研究成果としてモデルとその編集履歴を検索しメトリクスを算出するメトリクス API 仕様および設計，結果確認のための確認用 Web を示す．

① メトリクス API の仕様および設計

モデルリポジトリシステムに保存されているデータを対象とするメトリクスを定義し，定義したメトリクスを計算するためのメトリクス API を開発した．リポジトリマイニングにおいて使用される既存のメトリクス(表 3-2-1)を参考に，モデルリポジトリシステムの収集データを対象としたメトリクスを検討した．

表 3-2-1　既存のメトリクス

	対象の版	これまでの履歴
	領域 (1)	**領域 (2)**
コード	行数 [4, 10, 14, 24, 26–29, 32, 37–40, 43, 47, 55, 57, 68, 69, 72] オペレータ数 [9, 10, 26–29, 37, 40, 43, 47, 55, 68] McCabe メトリクス [26, 32, 40, 43, 55, 58, 68, 75] Halstead メトリクス [10, 40, 43, 47, 68] CK メトリクス [9, 12, 17, 24, 26, 63, 66, 73] メソッド呼び出し関係 [55, 64] リユースコード [51] 静的解析ツールの出力 [53, 62] ディレクトリ [32] 横断的関心事 [11, 15] コーディング規約違反 [3] コードクローン [60] トークン [21, 22, 32, 48, 49] 依存グラフ [74]	変更行数 [9, 26, 32, 36, 46, 52, 54, 62, 75] メソッド呼び出し関係の履歴 [64] 静的解析ツール出力履歴 [62]
	領域 (3)	**領域 (4)**
プロセス	プログラミング言語 [57, 69] コミットログ [32] コミット日時 [32]	過去のフォールト [13, 16, 36, 57, 75] コードの存在期間 [9, 16, 20, 26, 33, 57, 62, 69] 変更回数 [9, 16, 20, 26, 33, 46, 52, 56, 57, 59, 69] 不具合修正回数 [9, 20, 26, 33, 37, 52, 69] リファクタリング回数 [9, 26, 52] ロジカルカップリング [33, 50, 52] 変更の複雑度 [9, 19, 50] テストケース数 [36]

図 3-2-1 にモデルリポジトリシステムのデータから抽出したメトリクスを計算するアーキテクチャを示す．

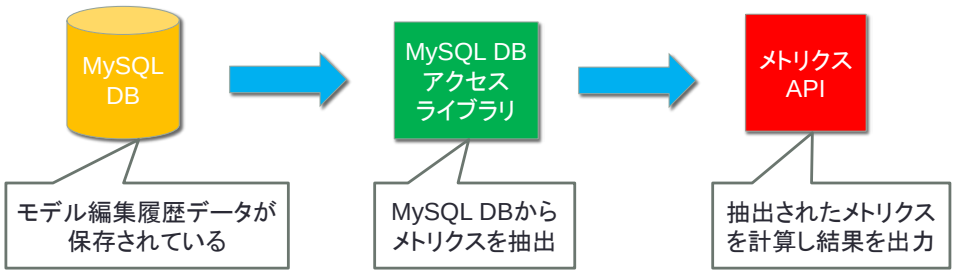

図 3-2-1　アーキテクチャ

モデルリポジトリシステムの MySQL DB にはモデル編集履歴のデータが保存されている．MySQL DB アクセスライブラリはモデルリポジトリシステムの MySQL DB からメトリクスを抽出し，メトリクス API に出力する．メトリクス API は入力されたメトリクスを計算し，その計算結果を出力する API である．抽出するメトリクスについては表 3-2-2 に示す．そして表 3-2-3 にメトリクス API を示す．

メトリクスは表 3-2-2 に示すように，プロセスメトリクス，プロダクトメトリクス，バグメトリクスに大別される．プロセスメトリクスには，ワークスペースをリポジトリにコミットした回数(コミット回数)と，コミットの際にリポジトリに追加された要素数(追加要素数)，リポジトリから削除された要素数(削除要素数)，変更された要素数(変更要素数)が存在する．プロダクトメトリクスには，リポジトリに保存されたダイアグラムごとの要素数，オブジェクトごとの関連するオブジェクト数，オブジェクトごとの関連するプロパティ数が存在する．バグメトリクスには，リポジトリに存在するバグ数が存在する．ここで，バグ数はワークスペースのモデルのコミットの際に，メッセージとして書き込んだバグチケット番号の数とする．さらに各メトリクスは，集計粒度毎にも分類される．例えば，プロセスメトリクスの一つであるコミット回数の場合，各リポジトリのコミット回数，各開発者のコミット回数，ある期間におけるコミット回数という 3 つの集計粒度に分類される．

メトリクス API においては表 3-2-3 に示すように，各メトリクスの計算に対応した合計 29 の API を定義した．ここで，バグメトリクス API については，中間目標 1 において開発を行った「バグトラッキングシステムとの連携機能」を利用したバグメトリクスの計算を可能とした．

表 3-2-2　メトリクス

メトリクス系	メトリクス	集計粒度
プロセスメトリクス	コミット回数	リポジトリ
		開発者
		集計期間
	追加要素数，削除要素数， 変更要素数	リポジトリ
		ダイアグラム
		開発者
		集計期間
プロダクトメトリクス	ダイアグラムごとの要素数	ダイアグラム
		開発者
		集計期間
	オブジェクトごとの関連するオブジェクト数	リポジトリ
		ダイアグラム
		開発者
	オブジェクトごとの関連するプロパティ数	リポジトリ
		ダイアグラム
		開発者
バグメトリクス	バグ数	リポジトリ
		ダイアグラム
		開発者
		集計期間

表 3-2-3　メトリクス API

メトリクス API	計算対象のメトリクス	メトリクス API 名	説明
プロセスメトリクス API	コミット回数	get コミット NumRepository	・特定のリポジトリに対して，特定の開発者が行ったコミットの回数を計算する ・集計期間を指定可能
		get コミット NumRepositories	・特定の開発者が行ったコミットの回数を，各リポジトリごとに計算する ・集計期間を指定可能
		get コミット NumAuthors	・特定のリポジトリに行われたコミットの回数を，各開発者ごとに計算する ・集計期間を指定可能
	追加要素数, 削除要素数, 変更要素数	getItemNumRepository	・特定のリポジトリに対して，特定の開発者が追加した要素数のみ，または削除した要素数のみ，変更を行った要素数のみを計算する ・集計期間を指定可能
		getItemNumRepositories	・特定の開発者が追加した要素数のみ,または削除した要素数のみ,変更を行った要素数のみを各リポジトリごとに計算する ・集計期間を指定可能
		getItemNumAuthor	・特定のリポジトリに追加した要素数のみ，または削除した要素数のみ,変更を行った要素数のみを,各開発者ごとに計算する ・集計期間を指定可能
		getDiagramItemNum	・特定のダイアグラムに対して,特定の開発者が追加した要素数のみ，または削除した要素数のみ，変更を行った要素数のみを計算する ・集計期間を指定可能
		getDiagramItemNumDiagrams	・特定の開発者が追加した要素数のみ,または削除した要素数のみ,変更を行った要素数のみを各ダイアグラムごとに計算する ・集計期間を指定可能
		getDiagramItemNumAuthors	・特定のダイアグラムに追加した要素数のみ，または削除した要素数のみ，変更を行った要素数のみを，各開発者ごとに計算する ・集計期間を指定可能

	追加・削除・変更要素数の全て	getAllItemNumRepository	・特定のリポジトリに対して，特定の開発者が追加・削除・変更を行った要素数全てを計算する ・集計期間を指定可能
		getAllItemNumRepositories	・特定の開発者が追加・削除・変更を行った要素数全てを各リポジトリごとに計算する ・集計期間を指定可能
		getAllItemNumAuthor	・特定のリポジトリに追加・削除・変更を行った要素数全てを，各開発者ごとに計算する ・集計期間を指定可能
		getDiagramAllItemNum	・特定のダイアグラムに対して，特定の開発者が追加・削除・変更を行った要素数を計算する ・集計期間を指定可能
		getDiagramAllItemNumDiagrams	・特定の開発者が追加・削除・変更を行った要素数を全て各ダイアグラムごとに計算する ・集計期間を指定可能
		getDiagramAllItemNumAuthors	・特定のダイアグラムに追加・削除・変更を行った要素数全てを，各開発者ごとに計算する ・集計期間を指定可能
プロダクトメトリクス API	ダイアグラムごとの要素数	getItemNumDiagram	・特定のリポジトリに存在する追加・削除・変更要素数を各ダイアグラムごとに計算する
		getItemNumVersion	・特定のダイアグラムに存在する追加・削除・変更要素数を各バージョンごとに計算する
		getItemNumAllRepositories	・各リポジトリの最新バージョンに存在する追加・削除・変更要素数を計算する
	オブジェクトごとの関連するオブジェクト数・プロパティ数	getRelatedItemNumVersions	・特定のリポジトリにおける，オブジェクトごとの関連するオブジェクト数・プロパティ数を各バージョンごとに計算する
		getRelatedItemNumObjects	・特定のリポジトリにおける，オブジェクトごとの関連するオブジェクト数・プロパティ数を各オブジェクトごとに計算する
		getRelatedItemNumDiagramObjects	・特定のダイアグラムにおける，オブジェクトごとの関連するオブジェクト数・プロパティ数を各オブジェクトごとに計算する

バグメトリクス API	バグ数	getBugNumRepository	・特定のリポジトリに対して，特定の開発者が発行したバグ数を計算する ・集計期間を指定可能
		getBugNumRepositories	・特定の開発者が発行したバグ数を各リポジトリごとに計算する ・集計期間を指定可能
		getBugNumAuthors	・特定のリポジトリに存在するバグ数を，各開発者ごとに計算する ・集計期間を指定可能
		getBugNumVersions	・特定のリポジトリに存在するバグ数を，各バージョンごとに計算する ・集計期間を指定可能
		getBugNumDiagram	・特定のダイアグラムに対して，特定の開発者が発行したバグ数を計算する ・集計期間を指定可能
		getBugNumDiagrams	・特定の開発者が発行したバグ数を各ダイアグラムごとに計算する ・集計期間を指定可能
		getBugNumDiagramAuthors	・特定のダイアグラムに存在するバグ数を，各開発者ごとに計算する ・集計期間を指定可能
		getBugNumDiagramAuthors	・特定のダイアグラムに存在するバグ数を，各バージョンごとに計算する ・集計期間を指定可能

次に，メトリクス API の設計について，例としてバグメトリクス API の一つである getBugNumRepository の処理フロー図を図 3-2-2 に示す．getBugNumRepository は特定のリポジトリに対して，特定の開発者が発行したバグチケット数を計算する API である．

メトリクス API は，実行開始直後，メトリクス計算対象のリポジトリ ID，開発者(ユーザ)ID を取得する．次に，MySQL DB アクセスライブラリを実行する．MySQL DB アクセスライブラリは，モデルリポジトリシステムの DB から，コミット情報の保存されているテーブルを参照し，コミットリストを生成する．このとき，コミットリストには，メトリクス計算対象のリポジトリ ID，開発者(ユーザ)ID に関するコミット情報が保存されている．

メトリクス API はコミットリストに保存されているメッセージ文字列から，チケット番号をパースしていく．チケット番号はバグトラッキングシステムに登録されているチケットの番号である．メトリクス API はパースした各チケット番号に対し，該当番号のチケットがバグチケットに該当するかバグトラッキングシステムに問い合わせを行う．パースしたチケットがバグチケットであれば，バグと判断しバグ数に加算する．

メトリクス API は，コミットリストに存在する全てのチケットのパースおよび，バグト

ラッキングシステムへの問い合わせが終了した後，計算したバグ数を出力して実行を終了する．

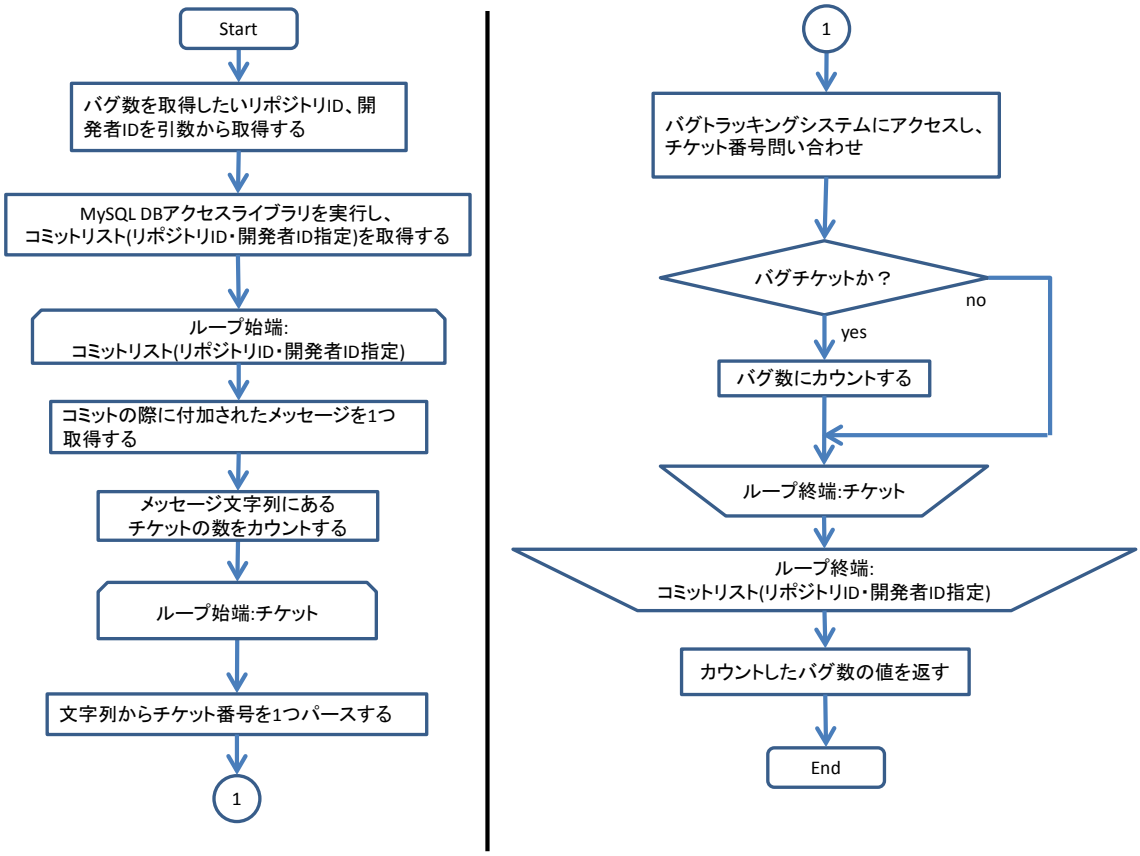

図 3-2-2　バグメトリクス API getBugNumRepository の処理フロー図

モデルリポジトリシステムで集積したデータに対しメトリクス API による計算が正しく行われるか確認を行った．メトリクス API の計算結果が各項目において想定されるメトリクスの想定値と一致することを確認し，API によるメトリクスが正確に計算可能であることを確認した(表 3-2-4)．バグメトリクス API においては，モデルリポジトリシステムに集積されたモデルのバグ情報を，コミットリストからパースし，バグメトリクスを計算可能であることを確認した．また中間目標 1 において開発を行った「バグトラッキングシステムとの連携機能」により，バグメトリクスの計算が可能であることを確認した．

表 3-2-4　メトリクス API の動作確認テスト項目

メトリクス API	テストにより動作確認 する機能	各メトリクス 集計粒度	テストの確認内容
プロセスメトリクス API	コミット回数の取得	リポジトリ	特定リポジトリのコミット回数が取得できることを確認
		開発者	特定開発者のコミット回数が取得できることを確認
		集計期間	特定期間におけるコミット回数を取得できることを確認
	追加要素数の取得，削除要素数の取得，変更要素数の取得	リポジトリ	特定リポジトリに追加，または削除，変更を行った要素数が取得できることを確認
		ダイアグラム	特定ダイアグラムに追加，または削除，変更を行った要素数が取得できることを確認
		開発者	特定開発者が追加，または削除，変更を行った要素数が取得できることを確認
		集計期間	特定期間において追加，または削除，変更を行った要素数が取得できることを確認
	追加・削除・変更された全ての要素数の取得	リポジトリ	特定リポジトリに追加・削除・変更を行った全ての要素数が取得できることを確認
		ダイアグラム	特定ダイアグラムに追加・削除・変更を行った全ての全要素数が取得できることを確認
		開発者	特定開発者が追加・削除・変更を行った全ての要素数が取得できることを確認
		集計期間	特定期間において追加，または削除，変更を行った全ての要素数が取得できることを確認
プロダクトメトリクス API	ダイアグラムごとの要素数の取得	ダイアグラム	特定ダイアグラムに追加・削除・変更を行った全要素数が取得できることを確認
		開発者	特定開発者が追加・削除・変更を行った全要素数が取得できることを確認
		集計期間	特定期間において追加，または削除，変更を行った要素数が取得できることを確認
	オブジェクトごとの関連するオブジェクト数の取得	リポジトリ	特定リポジトリに存在する，オブジェクトごとの関連するオブジェクト数が取得できることを確認
		ダイアグラム	特定ダイアグラムに存在する，オブジェクトごとの関連するオブジェクト数が取得できることを確認
		開発者	特定開発者が追加した，オブジェクトごとの関連するオブジェクト数が取得できることを確認
	オブジェクトごとのプロパティ数の取得	リポジトリ	特定リポジトリに存在する，オブジェクトごとのプロパティ数が取得できることを確認

		ダイアグラム	特定ダイアグラムに存在する,オブジェクトごとのプロパティ数が取得できることを確認
		開発者	特定開発者が追加した,オブジェクトごとのプロパティ数が取得できることを確認
バグメトリクス API	バグ数の取得	リポジトリ	特定リポジトリに存在するバグ数が取得できることを確認
		ダイアグラム	特定ダイアグラムに存在するバグ数が取得できることを確認
		開発者	特定開発者が出したバグ数が取得できることを確認
		集計期間	特定期間に存在するバグ数が取得できることを確認

② 確認用 Web の仕様および設計

「小規模プロジェクトでの実証評価」の取り組みとして，小規模プロジェクトにて適切に動作するか検証するため，メトリクスを表示する Web アプリケーション(確認用 Web)を設計，実装した．以下に確認用 Web の仕様および設計について説明する．

モデルリポジトリシステムを用いてモデル開発の講義を行い，そこで得られたリポジトリ及びバグ情報に対し，確認用 Web を用いて分析することを想定し，ユースケース分析を行った．

- アクター
  - 「学生」: clooca を利用してモデル作成を行う．
  - 「講師」: 確認用 Web を用いて学生が作成したモデルを解析し，適宜指導を行う．
  - 「研究者」: 確認用 Web を用いて学生が作成したモデルを解析し，推論モデルを作成する．
- ユースケース分析

  代表的なユースケースを以下に示す
  - ユースケース１:講師が講義期間中にモデルを解析し,指導を行うケース(図 3-2-3)

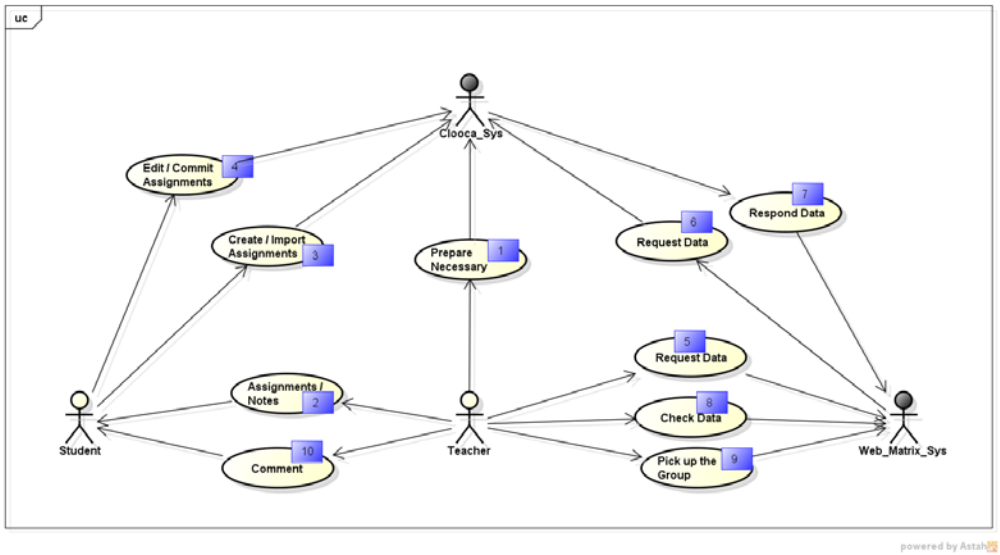

図 3-2-3　講師が講義期間中にモデルを解析し，指導を行うケース

- ユースケース２：研究者が講義期間終了後にモデルを解析し，推論モデルを作成するケース(図 3-2-4)

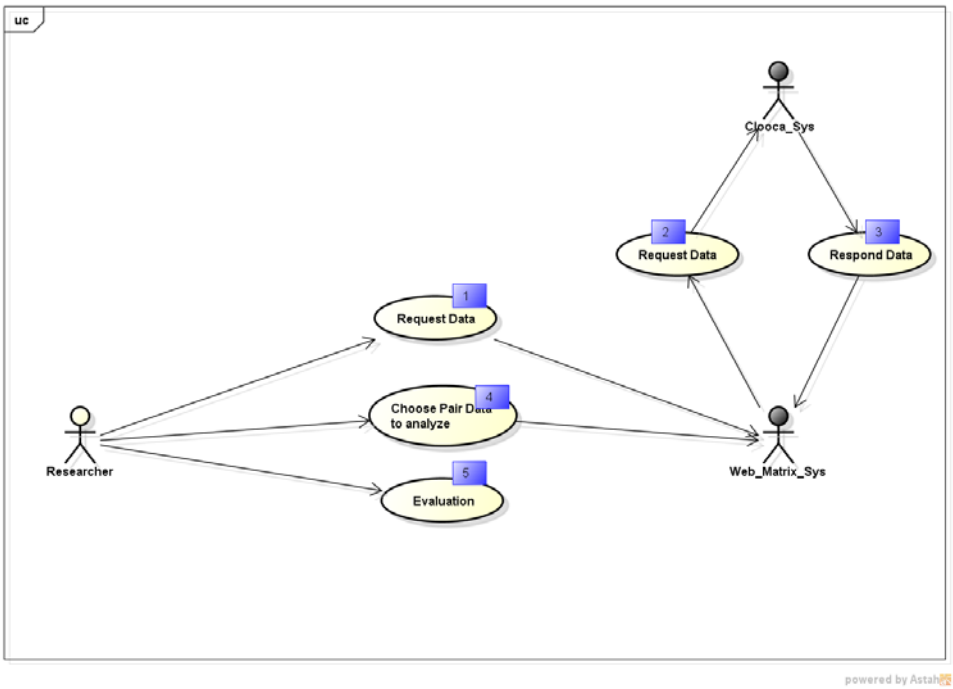

図 3-2-4　研究者が講義期間終了後にモデルを解析し，推論モデルを作成するケース

ユースケース分析から確認用 Web の仕様を，講師や研究者を対象に，すべての学生のメトリクスを様々な条件で検索・表示可能な web システムとする．

検索条件としてリポジトリ・ダイアグラム・ユーザ（学生)・バージョン・ターゲットオブジェクトからいくつかの条件(Conditions)を指定し，リポジトリ・ダイアグラム・ユー

ザ・バージョン・オブジェクトごとに（RowItem）並べて結果表示（Results）を行う．

研究者は様々な組み合わせの中からメトリクスの相関関係を探し出し推論モデルを作成するため，検索条件を自由に設定可能とする．

講師は研究者が発見した推論モデルを元に，プリセットされたメトリクスの組み合わせを選択可能(PresetItem)とする．

前述した仕様に基づき図 3-2-5 に示す確認用 Web の画面を設計し，メトリクス API を用いて実装を行った．実装を行った確認用 Web の実行画面を図 3-2-6 に示す．確認用 Web においてメトリクスの検索条件を設定し，モデルリポジトリシステム上の編集履歴データに対するメトリクス計算結果を表示可能であることを確認した．

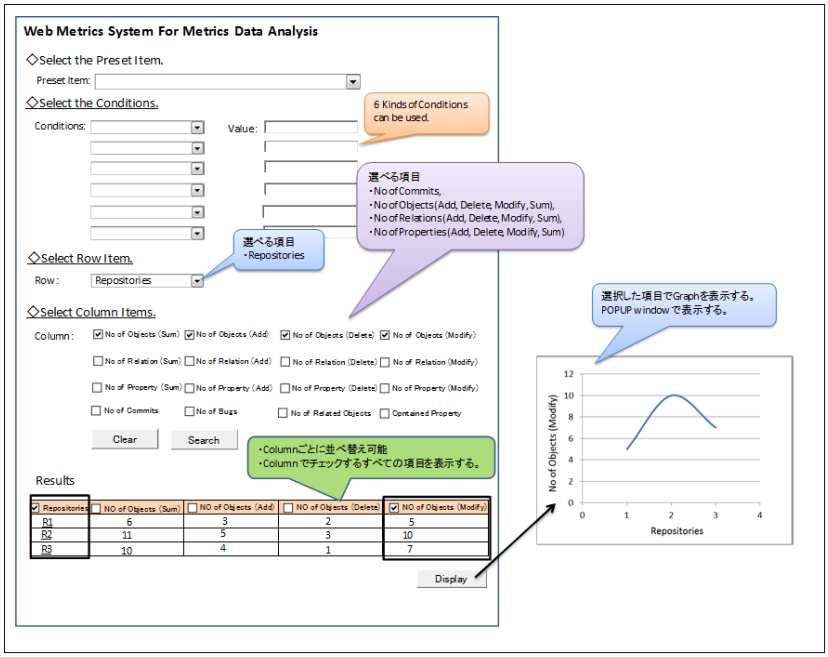

図 3-2-5　確認用 Web を活用した画面イメージ図

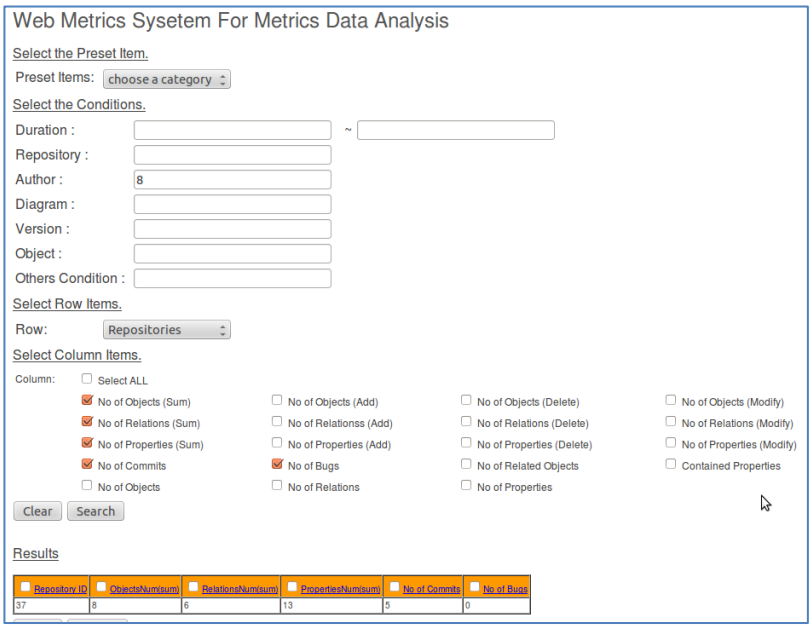

図 3-2-6　確認用 Web の実行画面

3.2.3 実用化へ向けた課題と問題点

(1) 課題と問題点

当初の到達目標で述べたとおり，モデルリポジトリシステムにおいて集積できるデータに対して検索し，必要なメトリクスを計算できるAPIを定義した．さらにその API が利用できるかどうかの確認を行った．本研究成果を実用化する際の課題と問題点について検討を行い，以下に示す三つの観点で言及する．

- モデル要素に着目したDBアクセスライブラリの再構築
- 外部のデータ解析ツールとの連携機能
- 一部メトリクスの取得に関する問題

① モデル要素に着目したDBアクセスライブラリの再構築

DBアクセスライブラリの出力がモデル要素の個数をベースとしているため，リポジトリ中のモデル要素をPythonObjectとしてPython上で自由に取り扱うことができない制約が発生する．これはOR-Mappingを用いてDBアクセスライブラリを再構築することで解決可能である．本取り組みを実用化に向けて取り組むことで，Pythonの豊富なライブラリを用いてより複雑なメトリクス解析が可能となる．さらに，モデルリポジトリシステムで開発したツールとの連携動作が幅広く適用可能となることが期待できる．

② 外部のデータ解析ツールとの連携機能

重回帰分析等，データ解析を行うR等の外部ツールとの連携機能を，現状のモデルリポジトリシステムは保有させていない．これは，外部プログラムとの間にAPIを設けてデータ授受により外部ツールとの連携を可能にすることによって比較的容易に実装可能であるが，実用化に向けて，適用するビジネス分野を踏まえて連携する解析ツール

を選定する必要がある．さらに，web表示や可視化機能の強化にも寄与すると考えられる．

③ 一部メトリクスの取得に関する製品化における課題

リレーションの変更等DB構造上取得不可能なデータや，プロパティの削除回数，ダイアグラムに関連する各種メトリクス等，取得に時間を要するメトリクスが存在している．製品化を行う際には採用するDBも高速化が必要であると想定されるので，製品化の際に，DB構造の見直しや補助的なテーブルを用いてメトリクス取得の高速化を図ることが製品検討事項となる．このことによって，webシステムや，講義中リアルタイムに受講生の習熟度を知りたい教育現場等への展開が容易となり幅広い分野への展開が大いに期待できる．

(2) 将来の応用方法

本研究成果として，すでに提案されているコードを対象としたメトリクスが，モデルにおいても有効に働くかどうかを検証できる手法を確立でき，モデルにおいて既存のコードを対象としたメトリクスが有効に働かない応用や場面においても，有効に働くモデル特有のメトリクスを検証できる手法も確立できたことで，教育，研究，開発に関する応用が考えられる．

3.3 研究目標 3「講義,PBL 等でツールを利用して,モデルとその編集履歴を集積する」

3.3.1 当初の想定

(1) 想定する仮説等

講義や Project-Based Learning (PBL)において，受講者に開発したツールを利用させ，モデルとその編集履歴を集積する．

(2) 当初の到達目標

九州大学大学院システム情報科学府で実施している,「モデル駆動開発特論」及び「PBL2」において開発したツールを利用し，モデルとその編集履歴を集積できるかどうかを確認する．

(3) 当初の期待される効果

講義及び PBL において clooca を利用することによってモデルとその編集履歴を収集し，モデリング教育においては教育効果の分析，開発においては開発に有用な情報を得るための材料とする．

3.3.2 研究プロセスと成果

(1) 研究プロセス

モデルとその編集履歴を収集するための手順について述べる.

本研究ではモデルとその編集履歴を収集するために，モデリング等を学ぶ講義，及び，Project-Based Learning (PBL)において改変した clooca を利用することとする．大規模な実プロジェクトにおいて改変した clooca を利用する方がより望ましいが,比較的短時間にモデルを収集できること，課題やプロセスを制御して望みの結果が得やすいことなどを考慮して，授業，及び，PBL においてモデルとその編集履歴を収集することとした.

1) 講義におけるモデルとその編集履歴の収集

講義は 2 種類実施することとした．ひとつは UML などのモデルを初めて学ぶ初学者向けの講義，もうひとつは UML を用いたモデル駆動開発を学ぶモデル駆動開発特論における講義である．両者ともにモデリングを必要とするような演習課題を与え，その演習課題を受講生が実施する過程のモデルを収集することとした.

1-1) 初学者向けの講義

UML を初めて学ぶ，もしくは，UML の文法は学んだことがあるものの，運用方法についてはあまり理解していない受講生を対象とする.

また，UML では多数の図が定義されているが，すべての図を学ばせることは目的とせず，システムを UML にてモデリングすることのエッセンスのみを学ばせることを目的とする．その結果，必要最低限のモデルとして，システムの静的構造を表現するためのクラス図，動的な振る舞いを表現するためのステートマシン図を選択する.

講義内容を表 3-3-1 に示す.1 限は 50 分である.1,2 限では主にオブジェクト指向，UML の復習のあとに利用するツールである clooca の解説を行う．3，4 限目に clooca と演習機材に習熟するための基礎演習を実施する．次にあとで述べる総合演習を実施する．基礎演習 1，2，及び，総合演習において受講生が作成したモデルを集積する．総合演習においては，モデルを記述しコンパイルすると，すぐにロボットを動作させることができる環境を教員側から提供する.

表 3-3-1　初学者向けの講義 講義内容

	講義内容
1 限目	オブジェクト指向，UML の復習，MDD とは？
2 限目	clooca の利用方法に関するチュートリアル
3 限目	基礎演習 1
4 限目	基礎演習 2
5 限目	基礎演習のレビュー，総合演習課題の説明
6 限目	総合演習課題の実施
7 限目	
8 限目	

1-2) モデル駆動開発特論

モデル駆動開発特論では修士 1 年生を対象とし,UML を用いたシステム開発の基礎知識があることを前提としてモデル駆動開発(MDD)について学ぶ講義である．具体的には,MDD する上で必要なモデリング手法，モデル上での検証法，モデルからのソースコード自動生成法について講義，演習を行う．さらに MDD 体制に移行するための方法論について講義，演習を行う．

また,モデル駆動開発特論においても UML で定義されているすべての図は使用せず,クラス図とステートマシン図を使用することとする．

講義内容について本研究に関連するコマのみを抜粋したものを表 3-3-2 に示す．1 限は 90 分である．1 限ではモデル駆動開発に関する基礎知識を学ぶ．2 限目では利用するツールである clooca の解説を行う.3〜6 限目にあとで述べる総合演習を実施する.また,総合演習は授業時間内だけでは終わらないため,授業時間外にも実施している．

初学者向けの講義と比べて総合演習の時間を多く必要とするのは，初学者向けの総合演習はモデルを記述するとすぐにロボットを動作させることができる環境を教員側から提供しているのに対し，モデル駆動開発特論においてはロボットを動作させるために必要なモデルから生成されるコードと，ロボットを制御する API とのインタフェースを開発することを目標に含めているからである．

表 3-3-2　モデル駆動開発特論 講義内容抜粋

	授業内容
1 限目	モデリング手法・演習
2 限目	clooca の利用方法に関するチュートリアル
3 限目	総合演習課題の実施
4 限目	
5 限目	
6 限目	

1-3) 総合演習

両講義において共通の総合演習について説明する．

総合演習では架空の運輸会社の自動搬送ロボットの開発業務を請け負うこととする．開発ターゲットは LEGO Mindstorms NXT を用いて開発した自律型車両ロボットである．自律型車両ロボットには，超音波センサ，バンパ，光センサ，タッチセンサが搭載されており，それぞれ，ロボット側面の側壁を検知することによる配送先判定，障害物のあたり判定，ライン判定，荷物判定を行うために利用する．

自動化する業務は，運搬業務，転送サービス，回送業務の 3 種類であり，これらの業務は荷物や配達先の有無により選択される．課題は光センサでコースの黒いラインをとレスし，配達先もしくは転送先，車庫で停止し，それぞれの地点で適切な動作を行い，所定の位置に荷物を届けることである．演習のコースを図 3-3-1 に示す．

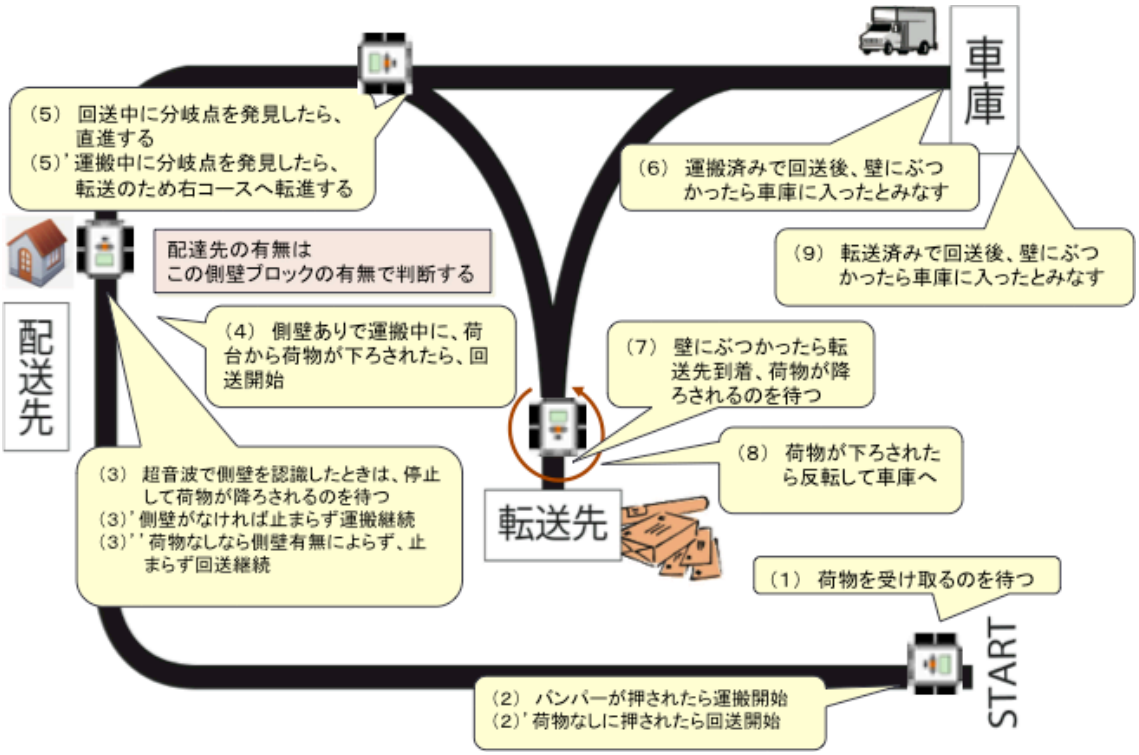

図 3-3-1　総合演習課題コース

2) PBL におけるモデルとその編集履歴の収集

より大規模なモデルとその編集履歴の収集を目指し PBL を対象とした集積を行う．PBL においては ESS ロボットチャレンジという情報処理学会 組込みシステム研究会 組込みシステムシンポジウムにおける特別企画である．本チャレンジでは小型飛行船を自動航行させるシステムを開発し，その飛行動作の達成度合いを競う．

システムの構成は以下の通りである．

- 飛行船に指令を出すグランドコントロール
- 飛行船搭載システム
  - 右，左，上下の 3 つの推進プロペラ
  - 高度測位のためのソナー
  - 方位推定のための角速度センサ
- 地上超音波マトリクスによる平面位置測位システム

グランドコントロールは飛行船と通信し，方位，高度の情報を受信し，地上超音波マトリクスからの情報をもとに平面位置を推定する．また，方位，高度，平面位置の情報を元にプロペラの推進力を決定し，飛行船に送信する．グランドコントロールが引こう指示を行い，離陸，ホバリング，指定された地点への移動，着陸などの決められたシナリオでの飛行動作を競う．

(2) 具体的な研究成果の内容

1) 講義におけるモデルとその編集履歴の収集

講義を通したモデルとその編集履歴の収集については，2 拠点で実施した．初学者向けの講義は徳山工業高等専門学校における特別講義にて実施した．また，モデル駆動開発特論は九州大学大学院システム情報科学府にて実施した．期間は 2012 年 9 月～2013 年 1 月であり，被験者は 16 名(高専 5 年生 2 名，専攻科 1 年生 10 名，修士 1 年 4 名)である．また，特に徳山工業高等専門学校における特別講義においては，モデルのみを記述したグループと，モデルからの自動コード生成を実施したグループの 2 グループに分けて実施した．

表 3-3-3 に講義におけるモデルとその編集履歴の収集結果を示す．また，被験者が作成したモデルの一部を図 3-3-2～3-3-5 に掲載する．

表 3-3-3　講義におけるモデルとその編集履歴の収集結果

クラス図数	1
総クラス数	6
ステートマシン図数	2
総ステート数	10
コミット数	22

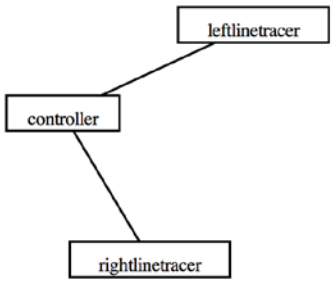

図 3-3-2　ある被験者のクラス図

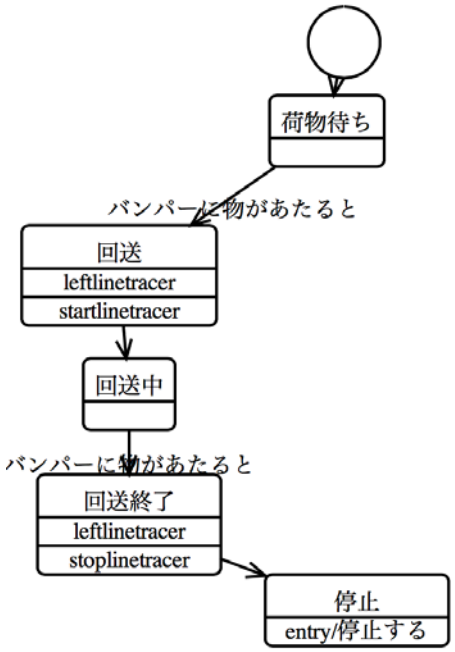

図 3-3-3　ある被験者のステートマシン図 1

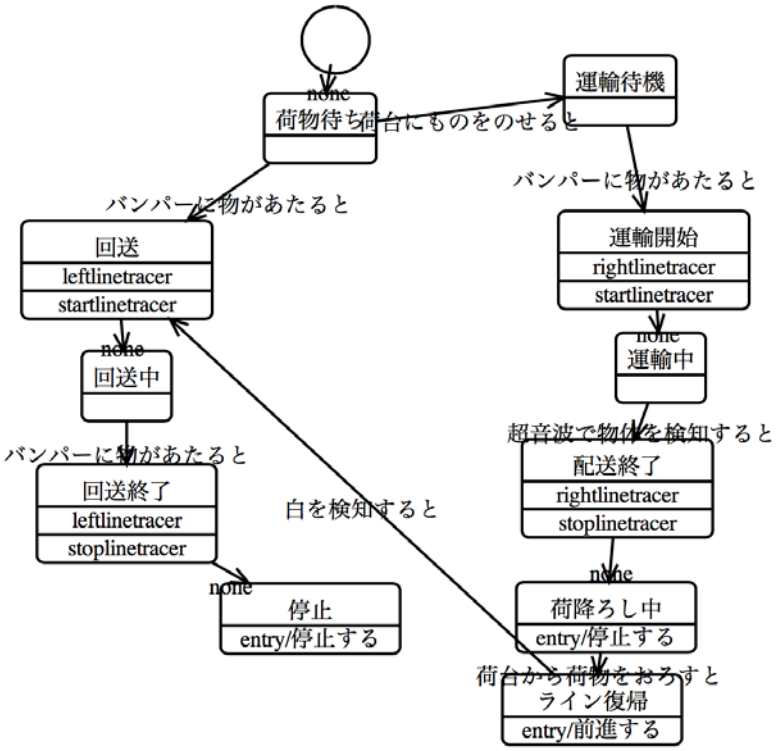

図 3-3-4　ある被験者のステートマシン図 2

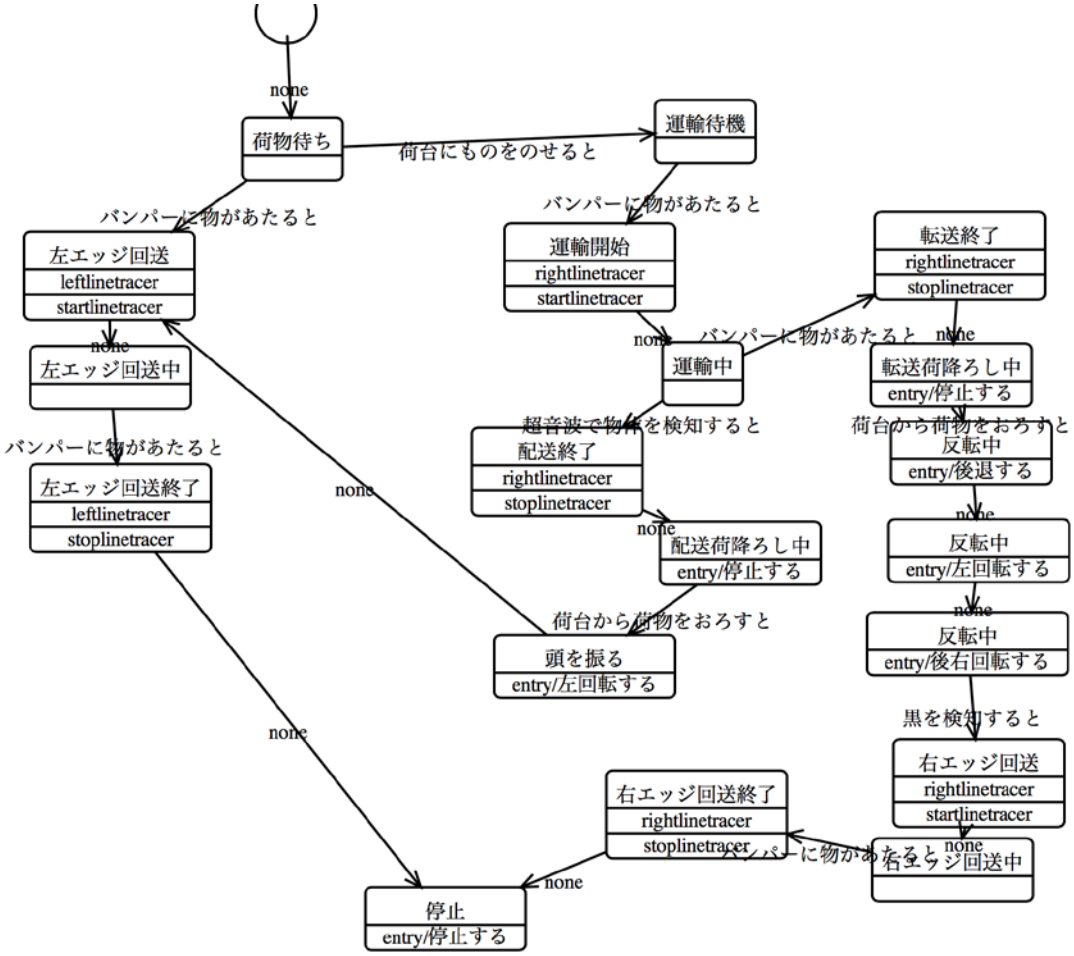

図 3-3-5　ある被験者のステートマシン図 3

2) PBL におけるモデルとその編集履歴の収集

九州大学大学院システム情報科学府情報知能工学専攻社会情報システム工学コースにおいて実施している，PBL2 においてモデルとその編集履歴を収集した．PBL2 は修士 1 年後期の 10 月～1 月に実施した．被験者は 4 名(すべて修士 2 年生)である．

当初，PBL は順調に遂行していた．しかしながら，12 月初旬ごろからアメリカにおけるヘリウムガスの生産設備のトラブルにより，ヘリウムガスが入手困難となった．飛行船を浮かせる上で必要不可欠であるヘリウムガスが入手困難となったことで，飛行船制御システムを題材とした PBL の続行が不可能になった．また，大会側も 2013 年夏頃に実施される次回大会については，飛行船競技の実施は難しい旨を通達してきた．

そこで PBL チームは急遽，PBL 課題を変更することとした．制御対象のロボットを図 3-3-6，3-3-7 に示す．本ロボットは LEGO Mindstorms NXT で作成した走行体 2 セットから構成されており，協調動作を行う．協調動作を行う両車体の上に板を渡し，空き缶を乗せ，その空き缶が落ちないように制御することとする．

コースを図 3-3-8 に示す．右下のスタート地点から①平地を直進走行し，②U 字にカーブし，③坂道を登る．④坂道の途中で停止し，再度，坂道を登り，坂道を登り切る．登り切ったところで⑤その場で 180 度転回を行い，⑤帰還する．

表 3-3-4 に PBL におけるモデルとその編集履歴の収集結果を示す．この収集結果は本報告書執筆時点のものであり，開発期間のおおよそ 50%を過ぎたところのデータである．PBL 期間の中盤における演習課題の急な変更により，総クラス数が 6，総ステート数が 10 と，比較的小規模なモデルにとどまっている．参考までに図 3-3-9～3-3-11 に開発を実施したモデルを掲載する．

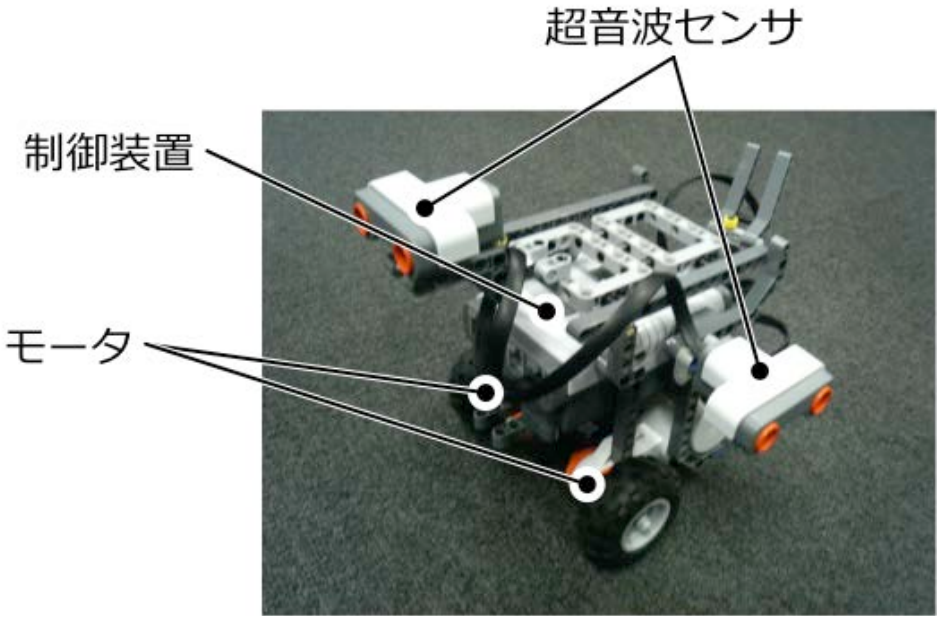

図 3-3-6　協調動作ロボット用走行体

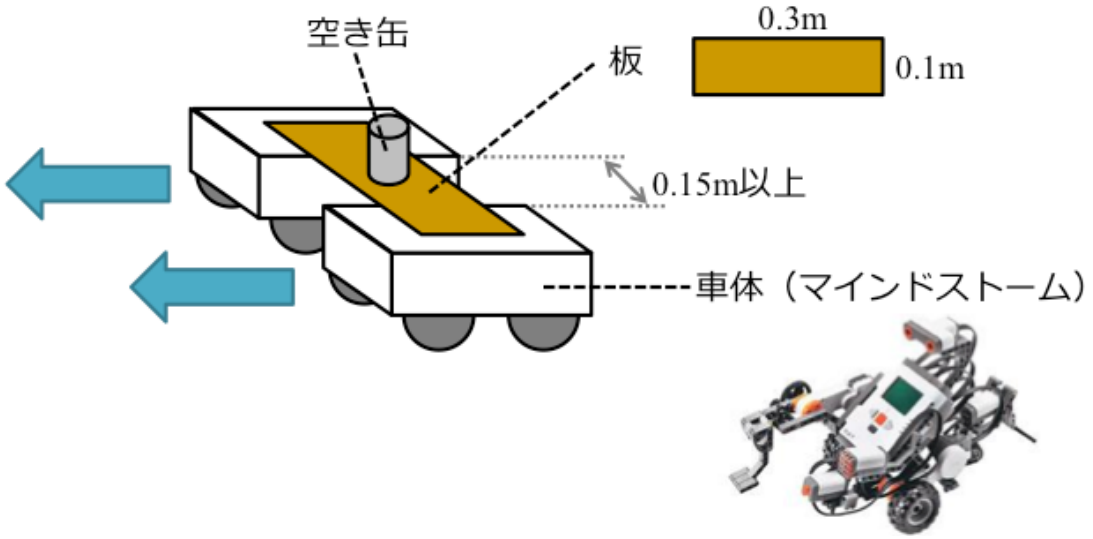

図 3-3-7　協調動作ロボット構成図

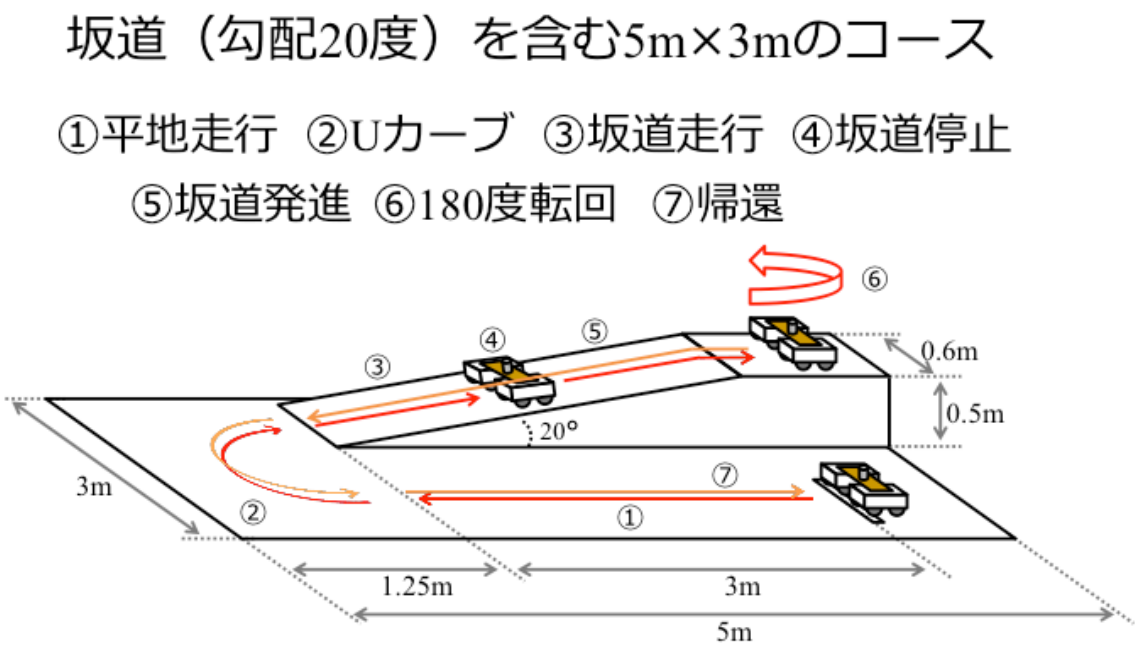

図 3-3-8　協調動作ロボットコース

表 3-3-4　PBL におけるモデルとその編集履歴の収集結果

クラス図数	1
総クラス数	6
ステートマシン図数	2
総ステート数	10
コミット数	22

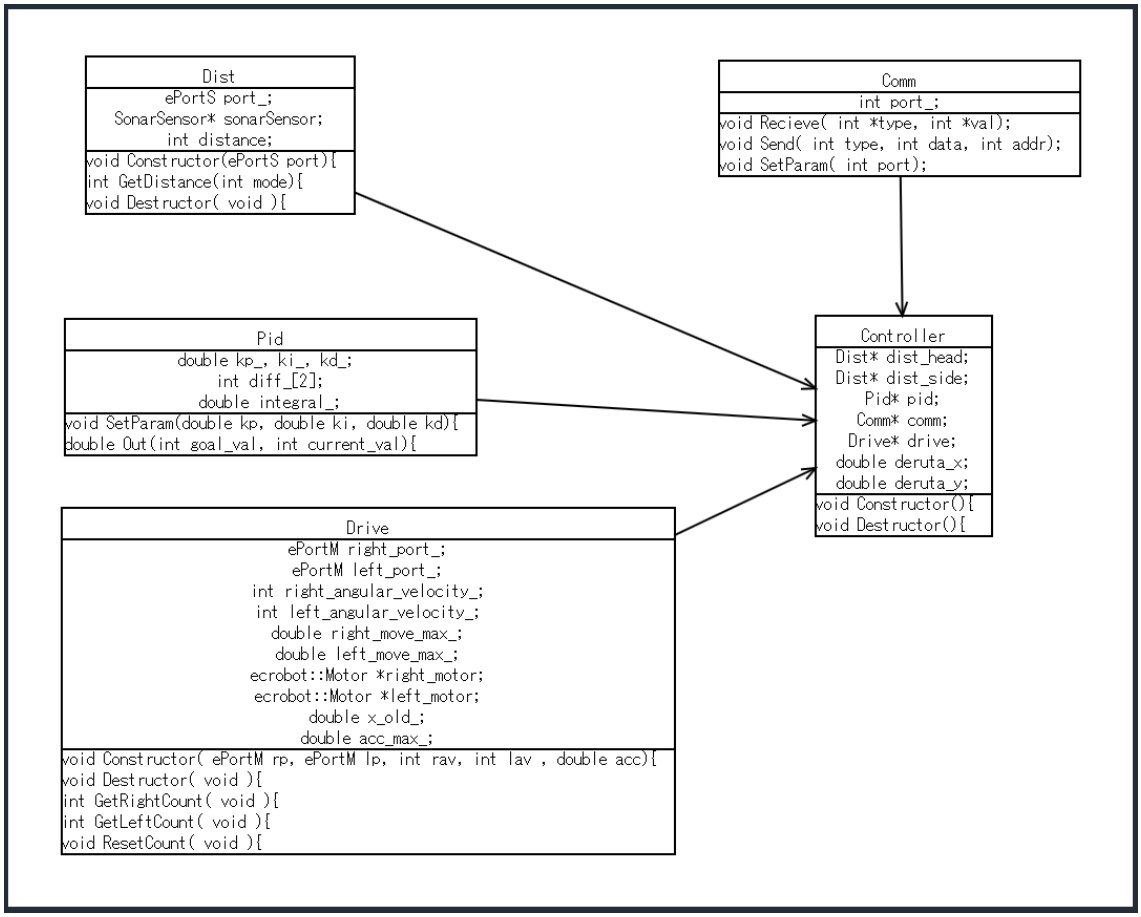

図 3-3-9　PBL において開発したクラス図

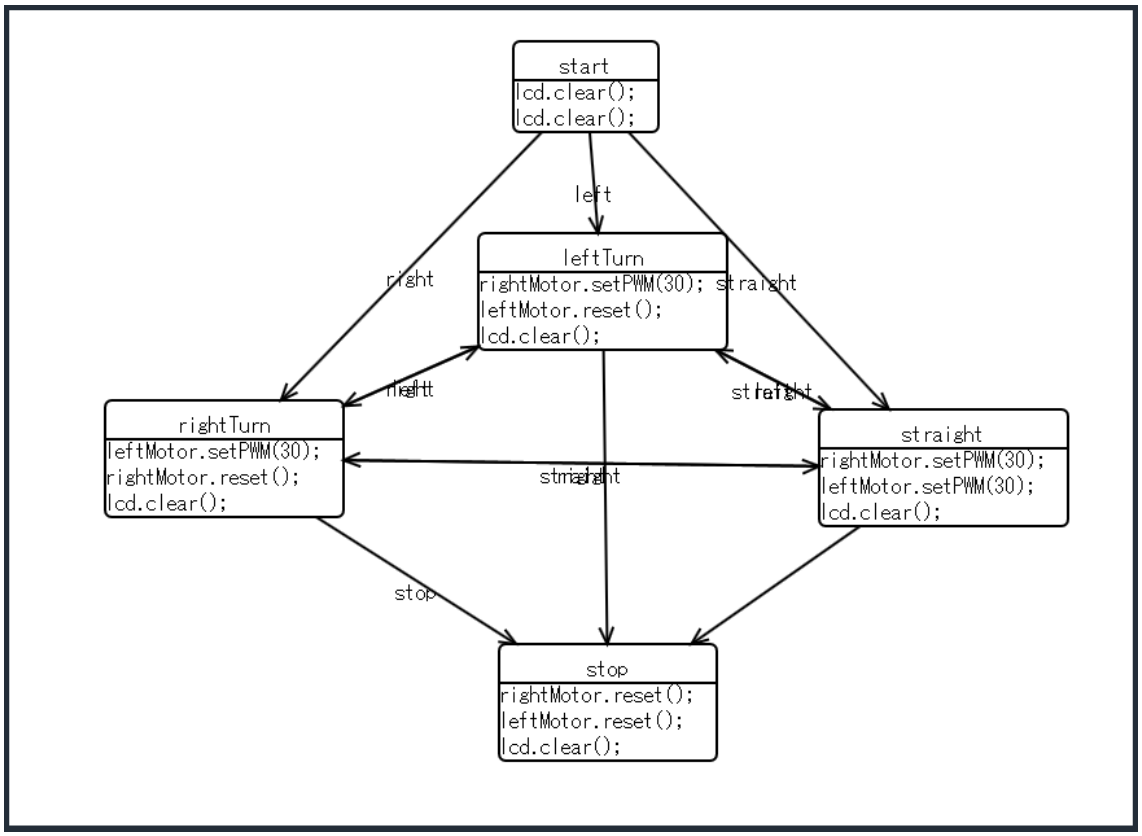

図 3-3-10　PBL において開発したステートマシン図（Controller）

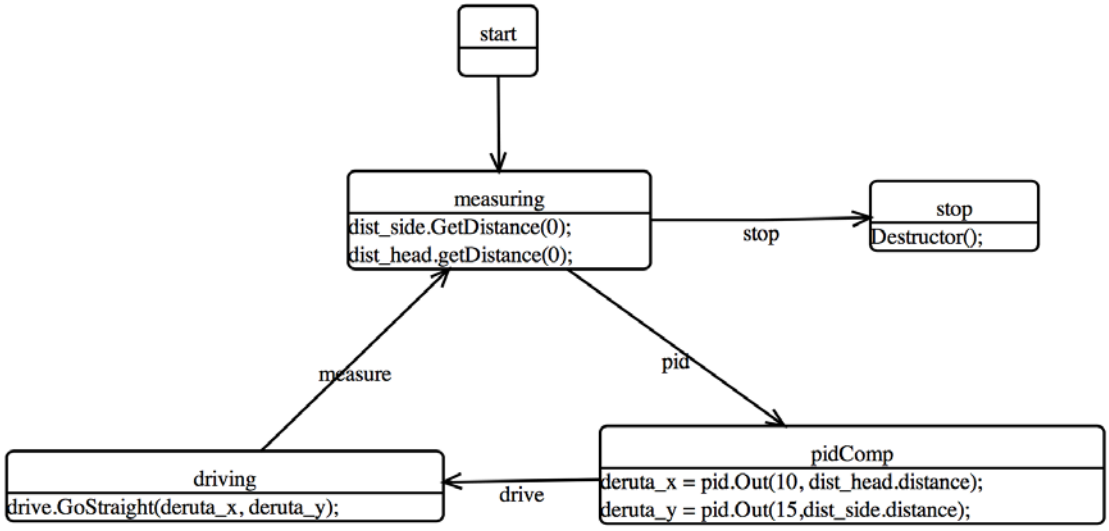

図 3-3-11　PBL において開発したステートマシン図（Dist）

3.3.3 実用化へ向けた課題と問題点

(1) 課題と問題点

当初の目標で述べたとおり授業及びPBL課題を通して,10人程度の受講生を対象として,モデルとその編集履歴の集積実験を行い，それらが集積できることを確認した．しかしながら期待された効果で述べた学習上の知見を得るために障害となり得る問題点として，以下の問題点が見いだされた．

- モデルとその編集履歴の収集頻度の不足
- 講義中に受講生のモデルを一覧することが困難

① モデルとその編集履歴の収集頻度の不足

収集したモデルとその編集履歴を解析する際に，一部の受講生はリポジトリにコミットする頻度が低く，その受講生が作成した設計成果物の成長過程を詳しく見ることができない事例が発生した．今回，モデルをリポジトリにコミットするタイミングは，1機能追加，1バグフィックス，1課題終了ごと，など，講義の中でインストラクションを与えたが，コミットの実施自体は受講生の主体性に任せる結果となった．

対策として，コミットしたタイミングで編集履歴を保存するのではなく，単に受講生がモデルを保存したタイミングですべての編集履歴を残すことを検討中である．その場合でも1編集が終了したというマークとしてのコミット操作は残す．

② 講義中に受講生のモデルを一覧することが困難

講義，及び，PBL 実施中に，受講生が設計した成果物を確認しつつ適切な指示を与えたいが，今回開発したモデルとその編集履歴を表示するユーザインタフェースは，多数の受講生が作成したモデルを一覧することが困難であった．そのため多数の受講生のモデルを見て，レビューしつつ指示を与えるごとができなかった．

対策としては，多数の受講生のモデルについて，注目すべきモデルのみ教員の意図に合わせて抽出し，一覧する機能の開発が必要である．

(2) 将来の応用方法

講義，及び，PBLにおいて，改変したcloocaを使用することでモデルとその編集履歴が収集できることが確認できた．このことによって，講義の現場，もしくは，開発の現場において同様の仕組みを利用することで，講義，開発支援のために必要なデータが得られるようになるものと考える．

また，今回の研究で得られたデータについては，整理した上で公開し，さまざまな立場の研究者や開発者に利用していただきたいと考えている．

3.4 研究目標 4「既存のコードを対象としたメトリクスがモデルでも有効に働くかどうか評価」

3.4.1 当初の想定

(1) 想定する仮説等

既存のコードを対象としたメトリクスがモデルでも有用に働くかどうかを評価する.

(2) 当初の到達目標

集積したデータを対象データとして，既存のコードを対象としたメトリクスが有効に働くかどうかを評価実験する.

(3) 当初の期待される効果

集積したデータを対象データとして分析し，講義においては教育効果の分析などの教育上手法上の知見を得ること，PBL においては開発に有用なバグ密度予測などの知見を得る.

3.4.2 研究プロセスと成果

(1) 研究プロセス

研究目標 3「講義，PBL 等でツールを利用して，モデルとその編集履歴を集積する」において収集したデータを対象として，分析を行う．特に均質なデータが取得できている徳山工業高等専門学校における授業において収集できたデータを対象として分析する．また，当該データは MDD 開発をしたチームとしなかったチームに分けて収集しているため，それらについての考察も行う.

モデルの品質は，Syntactic，Semantic，Pragmaticの3種類の品質カテゴリに分類できる[3-2]．各項目に関して関連するメトリクスを収集し考察を行う．また，必要に応じてモデルとその編集履歴を見て考察する．各品質カテゴリに関する主な評価内容は以下の通りである.

- Syntactic quality: 表記法の正確性
- Semantic quality: システム表現の正確性
- Pragmatic quality: 第三者が解釈する場合の正確性

まず，各品質カテゴリに関するエラーを持つクラス数を計算する．具体的には以下の通りである.

- Syntactic quality
  - 関連が引かれていない
  - 関連名が無い

- Semantic quality
  - 不必要なクラス（振る舞いが記述されていない）がある
- Pragmatic quality
  - クラス名がそのクラスの責務がわかる名前になっていない
  - 1つのクラス無いに複数の責務が盛り込まれている
  - 複数クラスに同一の機能が盛り込まれている
  - 関連名が不適切

さらに,Pragmatic 品質について分析を行うためにunderstandabilityに関する評価を行う[3-3]. 文献［3-4］によると，ステートマシン図の understandability と高い関係のあるメトリクスとして，Number of Activities(NA)， Number of States(NS)， Number of Transitions(NT) の3つを挙げている．計測するメトリクスの詳細は次の通りである．

- Number of Activities (NA)
  各クラスのステートマシン図それぞれの entry アクションの数のうち最大のもの．文献［3-4］では，アクションを entry， do， exit の 3 つに分類して扱っているが，本研究で用いた DSL では entry アクションのみ記述できるため，NA を entry アクションの数とする．
- Number of States (NS)
  各クラスのステートマシン図それぞれの状態数のうち最大のもの．
- Number of Transitions (NT)
  各クラスのステートマシン図それぞれの遷移数のうち最大のもの．遷移数には，初期状態，通常の状態，イベント送信状態間すべての遷移を含める．
- Number of Send Event States (NSE)
  各クラスのステートマシン図それぞれのイベント送信状態数のうち最大のもの．他のクラスにイベントを送信する状態が多いほど，モデルの understandability が低下すると考えられるため，Number of Send Event States(NSE)を計測する．

(2) 具体的な研究成果の内容

まず，品質カテゴリに関するエラーを持つクラス数を表 3-4-1 に示す．ただし，各グループ 6 名の受講生がクラス図を 1 つずつ作成しているため，最大数は 6 である．

MDD を利用しない場合は,開発する業務という抽象度の高い内容が思考の中心となり,走り方などの業務の実現方法には，意識がいきにくいため， 複数のクラスに同一の機能が盛り込まれていることが多いと考えられる．

一方，MDD を利用する場合には，機能ごとの単体テストを行いながら，機能の追加を行えるため,抽象度の低い業務の実現方法が思考の中心となり,全体としてどのような業務を行うクラス図なのかわかりにくいモデル図となっている．さらに,動作として確認することを優先しがちになるため,関連が引かれていないクラスや,クラスが一つしかなく,適切な

責務分割が行われていないモデルなど，品質に問題があるクラス図が多く見られた．

表 3-4-1　品質カテゴリに関するエラーを持つクラス数

品質カテゴリ	エラー項目	MDD なし	MDD あり
Syntactic	関連が引かれていない	0	2
	関連名がない	2	4
Semantic	不必要なクラス（振舞いが記述されていない）がある	0	1
Pragmatic	クラス名がそのクラスの責務が分かる名前になっていない	0	2
	1 つのクラス内に複数の責務が盛り込まれている	0	1
	複数クラスに同一の機能が盛り込まれている	6	0
	関連名が不適切	0	2

次に，作成されたステートマシン図の understandability に関するメトリクスを，MDD なしグループについては表3-4-2に，MDD ありグループについては表3-4-3に示す．

表3-4-2によると，メトリクスの平均においては NA 以外のメトリクスはすべて MDD ありの方が高くなっている．個別の被験者の結果を見ると，MDD なしは各被験者のモデル間のばらつきが少ないのに対し，MDD ありではすべてのメトリクスにおいて高い数字になっているモデルがある．MDDありの方が understandability が低いモデルが多く，複雑度が高いと考えられる．この結果からもMDDありのほうが，動作として確認することを優先しがちになるために，品質に注意を払われない傾向があることがうかがえる．

表 3-4-2　ステートマシン図の understandability に関するメトリクス
(MDD を利用しないグループ)

	NA	NS	NT	NSE
student1	9	9	11	2
student2	6	6	8	2
student3	8	8	11	2
student4	8	8	10	2
student5	7	7	9	2
student6	7	8	12	2
平均	7.5	7.7	10.2	2.0

表 3-4-3　ステートマシン図の understandability に関するメトリクス
(MDD を利用するグループ)

	NA	NS	NT	NSE
student1	7	19	20	7
student2	5	12	15	7
student3	4	6	8	3
student4	5	9	10	3
student5	5	7	12	7
student6	12	15	26	0
平均	6.3	11.3	15.2	4.5

さらにモデルを詳細に分析する．作成したクラス図におけるクラス名を「全体を制御するクラス」，「業務内容に関するクラス」，「走り方に関するクラス」の3つに分類した結果及び総クラス数を表3-4-3，表3-4-4に示す．MDDなしとMDDありグループの傾向を比較するとMDDありグループでは，業務内容に関するクラス名のみであるのに対し，MDD なしグループでは走り方に関するクラスの数の方が多く，業務に関するクラスを記述していない被験者も 4 名いた.基礎演習 2 では，総合演習課題の内容の一部であるライントレースのモデルの作成を行っており，基礎演習 2 のモデルをそのまま利用している，または，一部修正を加えて利用している数をモデルリポジトリから確認すると，MDD ありグループの 6 名中 4 名であった.

表 3-4-3　クラス名の分類(MDD なし)

	student1	student2	student3	student4	student5	student6	平均
全体を制御するクラスの数	1	1	1	1	1	1	1.0
業務の内容を定義するクラスの数	4	4	3	3	4	3	3.5
走り方を定義するクラスの数	0	0	0	0	0	0	0.0
総クラス数	5	5	4	4	5	4	4.5

表 3-4-4　クラス名の分類(MDD あり)

	student1	student2	student3	student4	student5	student6	平均
全体を制御するクラスの数	1	1	1	1	1	1	1.0
業務の内容を定義するクラスの数	0	2	0	0	2	0	0.7
走り方を定義するクラスの数	2	3	3	4	3	0	2.5
総クラス数	3	6	4	5	6	1	4.2

次に，図3-4-1，3-4-2にMDDなし，図3-4-3，3-4-4 にMDDありグループの代表的なステートマシン図を示す． MDDありグループのモデルは，図3-4-3のように状態名が無いモデルや状態名がa， bなど状態の意味が読み取れないものなどが見られた．

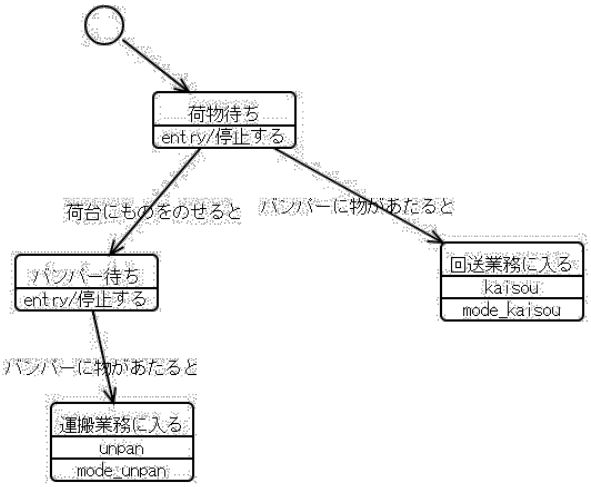

図 3-4-1　MDD なしグループのステートマシン図例 1

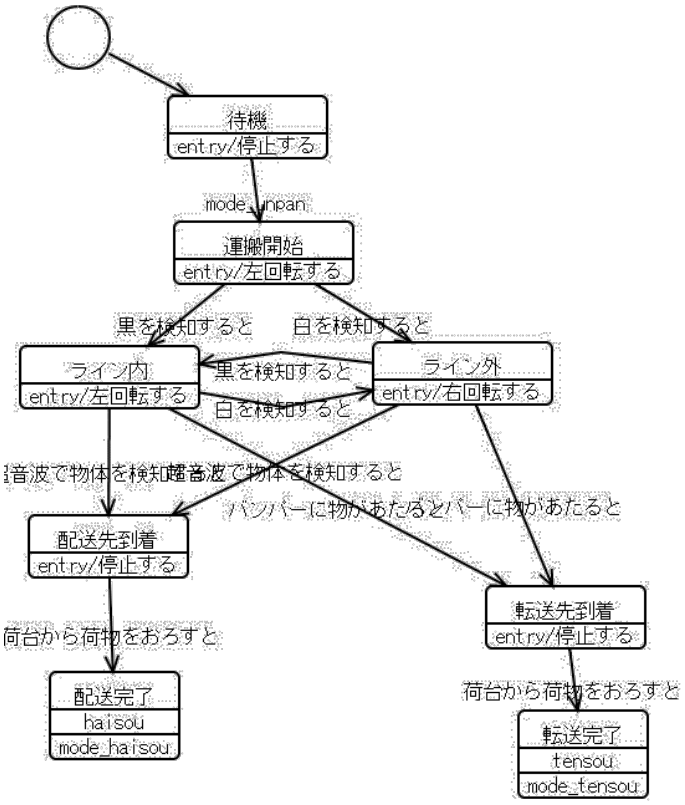

図 3-4-2　MDD なしグループのステートマシン図例 2

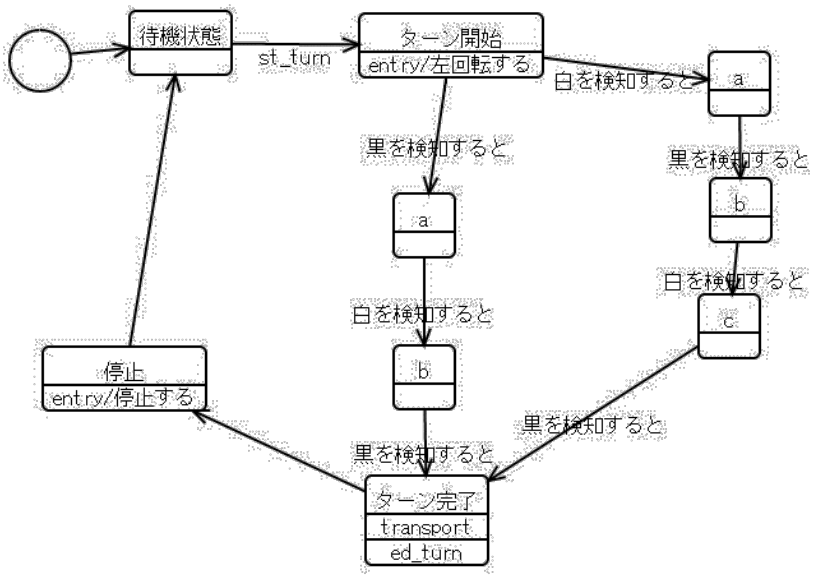

図 3-4-3　MDD ありグループのステートマシン図例 1

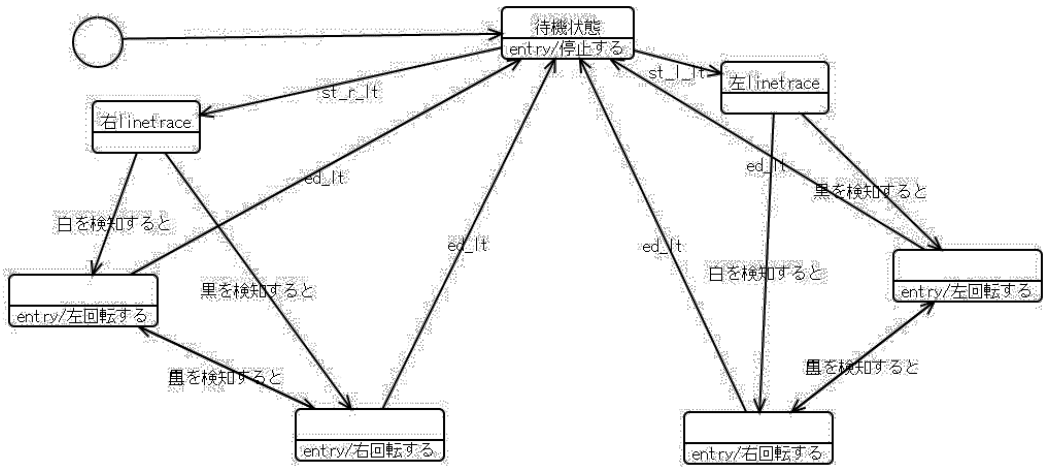

図 3-4-4　MDD ありグループのステートマシン図例 2

MDD なし，MDD ありの各グループの被験者 6 名をそれぞれMDDなし，ありの2つのグループに分けてモデリング演習を行った結果より，MDD の利用の有無それぞれの UML モデリング技術習得における利点を以下に示す．

- MDDなし
  - システムの業務に着目したモデリングを行える
  - モデリング記法に従い正確に表記できる
  - クラスや状態に適切な名前を付けられる
  - 読みやすいモデルになるように心がけられる

- MDDあり
  - システムの機能に着目したモデリングが行える
  - 短期間に機能を完成させることができる

これらのことより,UMLモデリングの教育効果を上げるためには,単純に MDD ツールを用いた開発をさせるのではなく，コードの自動生成や動作確認の回数を制限した演習を実施する必要があるといえる.例えば,基礎的な演習で用いるライントレースなどの小さな機能の完成のタイミングでコードの自動生成及び動作確認をさせて，機能を実現できているか確認させる.この時,クラス名や状態名のつけ方,モデルの表記法などを合わせてチェックさせるようにする必要がある.その後に行う，業務システム(本稿では，自動搬送システムを用いた)開発の場合には,十分なクラス図の検討を行った後にコードの自動生成や動作確認を行うように指導するのが望ましいと考える.

3.4.3 実用化へ向けた課題と問題点

(1) 課題と問題点

今回の適用実験においては，当初の研究目標，期待される効果で述べたとおり，講義に関係する知見を得ることができたが，PBL に関しては十分な規模のデータを収集することができなかった．結果として以下の課題が残った.

- 大規模，実開発データの収集
- リアルタイムなメトリクス提示

① 大規模，実開発データの収集

今回，収集できたデータは，数十コミット程度，開発人数は最大 4 名程度と比較的小規模にとどまった．モデルとその編集履歴から開発に関する知見を得たい場合には不十分な開発規模であった.対策として,引き続き PBL を対象として大規模なデータ収集を試みる.また，実開発で利用しデータ収集できるようにすべく，ツールの完成度の向上，及び，ツールの普及に努めたい.

② リアルタイムなメトリクスの提示

今回は比較的静的にメトリクスを計算し，表示を行った．この方式では講義終了後に演習の過程を分析するためには十分であるが，授業中に必要に応じて受講生をフォローするためには不十分である．受講生が設計成果物を変更したタイミングでメトリクスを再計算し，問題があるレベルに達したら講師に通知するなどの仕組み化が必要である.

(2) 将来の応用方法

本研究ではモデルとその編集履歴からメトリクスを計算し，また，モデルを抽出して，モデリング教育上の知見を得ることができた．教育においては，本手法をさらに発展させて講義，演習の支援，及び，講義の改善につなげることができると思われる．また，実開

発に関するデータが集積できれば，モデルとその成果物から実際の開発に関する知見を定量的に得ることができるようになるものと思われる．

４．考察

4.1 研究により判明した効果や問題点等

各目標について研究を進めてきた結果，判明した効果や問題点等を述べる．

① モデルとその編集履歴を集積できるようにツールを改変し，動作確認する

既存のモデリングツールはモデルの履歴を保存する機能が無かった．また，既存のリポジトリシステムにモデルを単純に登録するだけでは，モデルの履歴に対して横断的に検索をかけることが困難である．そのため，モデルとその編集履歴を収集できるツールの開発が必要不可欠であった．

本研究の遂行によりモデルとその編集履歴，及び，バグ情報などを，ツールの支援のもとに収集できるようになった．具体的には，SaaS 型ドメイン特化モデリング言語ツールである clooca を対象として，モデルとその編集履歴を集積し，分散協調開発を行えるようにするためのモデルリポジトリを構築した．さらに，バグ情報とリポジトリ上の変更を対応づけるために，バグトラッキングシステムとの連携機能を開発した．

これらの活動により，モデルとその編集履歴，及び，バグ情報などを，ツールの支援のもとに収集できるようになったが，一部，編集履歴の蓄積に不十分な点が発見された．③の活動において収集したモデルとその編集履歴を解析する際に，一部の受講生はリポジトリにコミットする頻度が低く，その受講生が作成した設計成果物の成長過程を詳しく見ることができない事例が発生した．今回，モデルをリポジトリにコミットするタイミングは，1 機能追加，1 バグフィックス，1 課題終了ごと，など，講義の中でインストラクションを与えたが，コミットの実施自体は受講生の主体性に任せる結果となった．対策として，コミットしたタイミングで編集履歴を保存するのではなく，単に受講生がモデルを保存したタイミングですべての編集履歴を残すことを検討中である．

② モデルの編集履歴を検索・利用するための API を定義し，動作確認する

①で集積した設計成果物に対するメトリクスを研究者や教育者の要望に応じて計算，提示することを支援するための API を構築した．計算したいメトリクスは，研究内容，授業支援内容などにより異なるため，できる限り多くのメトリクスを計算できるように開発するのでは無く，成長させやすいアーキテクチャとした．具体的には，データベースに格納されているモデルとその編集履歴を取り出しメトリクスの計算を支援するレイヤを構築し，さらのその上位にメトリクスを計算するレイヤ，指定したメトリクスを計算し提示する Web インタフェースを開発した．

③ 講義，PBL 等でツールを利用して，モデルとその編集履歴を集積する

講義や Project-Based Learning (PBL)において，受講者に開発したツールを利用させ，モデルとその編集履歴の集積を行った．このことによってモデルとその編集履歴を収集し，教育や研究上の知見を得るための材料とする．

講義は 2 種類実施することとした．ひとつは UML などのモデルを初めて学ぶ初学者

向けの講義，もうひとつは UML を用いたモデル駆動開発を学ぶモデル駆動開発特論における講義である．両者ともにモデリングを必要とするような演習課題を与え，その演習課題を受講生が実施する過程のモデルを収集することとした．初学者向けの講義は徳山工業高等専門学校における特別講義にて実施した．また，モデル駆動開発特論は九州大学大学院システム情報科学府にて実施した．その結果，改変した clooca を使用することで，当初想定した規模と数のモデルとその編集履歴が収集できることが確認できた．被験者 16 名分のモデルとその編集履歴を収集することができた．

より大規模なモデルとその編集履歴の収集を目指しPBLを対象とした集積を行った．PBL においては ESS ロボットチャレンジという情報処理学会 組込みシステム研究会 組込みシステムシンポジウムにおける特別企画である． PBL においてはプロジェクト途中で課題の変更を余儀なくされたため，当初想定した規模の設計成果物を得ることができなかった．

これらの活動を通じて，講義，及び，PBL において，改変した clooca を使用することでモデルとその編集履歴が収集できることが確認できた．このことによって，講義の現場，もしくは，開発の現場において同様の仕組みを利用することで，講義，開発支援のために必要なデータが得られるようになるものと考える．

公開可能なモデルとその編集履歴を収集した事例はきわめてまれであり，研究コミュニティにインパクトを与える成果であると考える．

また，特質する事項として clooca はインストール不要な SaaS 型のモデリングツールであるため，比較的大人数が使用する講義や開発においての導入コストが著しく低くなる．今回実施した講義についても，従来であればツールの導入に 1 日程度時間を要していたが，今回は数時間のインストール作業で十分であった．また，SaaS 型であるためすべてのモデルがサーバ上にあり，教員がチェックしやすい状態にあるため，レビューし修正させるべきモデルを講義中に発見することができた．

また，教育上の効果として，これらの活動に受講生として参加した学生，被験者として参加したアルバイト学生は，実際にモデル駆動開発を体験することができ，モデル駆動開発環境が提供されている状況において開発を進めるスキルを身につけることができたことを，事後課題で確認できた．また，その中でも特に一部の学生は，従来型の開発からモデル駆動開発体制に移行するための基礎的なスキルを身につけることが確認できた．

④ 既存のコードを対象としたメトリクスがモデルでも有効に働くかどうか評価

③において収集したデータを対象として，分析を行った．特に均質なデータが取得できている徳山工業高等専門学校における授業において収集できたデータを対象として分析した．また，当該データは MDD をしたチームとしなかったチームに分けて収集しているため，それらについての考察を行った．その結果，MDD をしたチームとしなかったチームにおけるモデルの品質傾向が明らかとなり，講義に関する改善点が明らかとなった．具体的には，UML モデリングの教育効果を上げるためには，単純に MDD ツールを用いた開発をさせるのではなく，コードの自動生成や動作確認の回数を制限した演習を実施する必要がある，レビューを適切なタイミングで実施すべきである，など

である.

PBL については今後の課題に述べるとおり，データ収集が十分にできなかったため，開発上の知見を得ることができなかった.

4.2 今後の課題

4.2.1 未達成の課題と今後の研究予定

① PBL の継続実施によるさらなるモデルとその編集履歴の集積

当初は ESS ロボットチャレンジの昨年度までの課題である模型飛行船制御を対象とした開発を進めてきた．しかしながら，12 月初旬ごろからアメリカにおけるヘリウムガスの生産設備のトラブルにより，ヘリウムガスが入手困難となった．飛行船を浮かせる上で必要不可欠であるヘリウムガスが入手困難となったことで，飛行船制御システムを題材とした PBL の続行が不可能になり，課題変更を余儀なくされた．そこで PBL チームは急遽,PBL 課題を変更し 2 台の地上走行するロボットが協調走行する課題に変更した．その結果，当初想定した規模のモデルとその編集履歴が収集できなかった.

現在も当 PBL は続行中であり，平成 25 年 7 月まで継続予定である．PBL 終了時には当初想定した規模のモデルとその編集履歴が得られるものと思われる.

4.2.2 新たに見いだされた課題と今後の研究予定

① clooca のソーシャルネットワークサービス(SNS)化

中間報告において，データの収集のみではなく，clooca を SNS 化しモデルや部品の交換，それらに対してコメントをつけ合うなどのプロジェクト環境を創出してほしい，とのコメントをいただいた．今回は研究期間，内容の関係上実施することができなかったが，clooca の SNS サービス化は当初から構想しており，今後，実現していきたいと考えている.

② 他団体が提供しているプロジェクト管理，診断ツールとの連携

中間報告において， UML の更新の履歴等、上流工程の開発過程データを収集したいという要求がたくさん出ているが，世の中で使われているツールにはそのような機能が付いていない，IPA が提供しているプロジェクト管理，診断ツールと連携できるようにし，世の中に広げるようにしてほしい，とのコメントを頂いた．他ツールとの連携，及び，clooca の普及推進は重要な課題だと考えており，今後，取り組んでいきたい.

③ メトリクスのリアルタイムな提供

今回は比較的静的にメトリクスを計算し，表示を行った．この方式では講義終了後に演習の過程を分析するためには十分であるが，授業中に必要に応じて受講生をフォローするためには不十分である．受講生が設計成果物を変更したタイミングでメトリ

クスを再計算し，問題があるレベルに達したら講師に通知するなどの仕組み化が必要である．

④ 大規模なデータ収集のための実プロジェクトを対象とした実験

有効な知見を得るためには，講義や PBL ではなく，実際の開発現場において今回開発したツールを利用してもらい，データを収集し，分析することが必要である．現状の clooca はドメイン特化モデリング言語の開発支援ツールであり，その上でのモデリング言語の定義はユーザに任せられている．本研究を遂行する上で必要最低限のモデリング言語を定義し，コード生成器を開発して利用したが，実際に開発で利用するためには，UML のサポートをさらに進めていく必要があるものと考える．

参考文献

[2-1] Steven Kelly and Juha-Pekka Tolvanen, Domain-Specifc Modeling: Enabling Full Code Generation, John Wiley Son, Inc (2008).
[2-2] Stephen J. Balcer, Marc J. Mellor, Executable UML: A Foundation for Model-Driven Architecture, Addison-Wesley Object Technology Series (2002).
[2-3] 赤山聖子, 久住憲嗣, 福田 晃: UML モデリング教育におけるモデル駆動開発ツールの利用方法の検討, 情報処理学会 第 117 回コンピュータと教育研究発表会 (2013).
[3-1] 産業界と連携した高品質組込みソフトウェア技術者養成プロジェクト: 自動搬送ロボット組立図, 産業界と連携した高品質組込みソフトウェア技術者養成プロジェクト報告書 (2011).
[3-2] Lindland, O. I., Sindre, G. and Sølvberg, A.: Under- standing Quality in Conceptual Modeling, IEEE Softw.,
Vol. 11, No. 2, pp. 42–49 (1994).
[3-3] Genero, M., Fernandez-Saez, A. M., Nelson, H. J., Poels, G. and Piattini, M.: Research Review: A Systematic Literature Review on the Quality of UML Models, J. Database Manag., Vol. 22, No. 3, pp. 46–70 (2011).
[3-4] Miranda, D., Genero, M. and Piattini, M.: Empirical Validation of Metrics for UML Statechart Diagrams, Enterprise Information Systems V (Camp, O., Filipe, J., Hammoudi, S. and Piattini, M., eds.), Springer Netherlands, pp. 101–108 (2005).