ウォーターフォール型ソフトウェア開発 PBL における ビルドエラーの調査方法の提案

古川 貴一[†1] 櫨山 淳雄[†1] 橋浦 弘明[†2] 鷲崎 弘宜[†3]

概要：ソフトウェア開発においてビルドは重要な工程の 1 つである. 既存研究として企業や学生のソフトウェア開発におけるビルド活動の調査が行われている. 一方, 高等教育機関で開講されているウォーターフォール(以下 WF)型のソフトウェア開発 PBL(Project Based Learning, 以下 SDPBL)におけるビルド活動の調査, WF 型特有の成果物の一貫性の調査は行われていない. 本研究では WF 型の SDPBL におけるビルドエラーの調査, 成果物の一貫性の調査方法を提案する.

キーワード：ビルドエラー, ソフトウェア開発 PBL, ウォーターフォールモデル, Gradle, 成果物の一貫性

1. はじめに

ソフトウェア開発においてビルドは重要な工程である. 企業を対象に, ビルドが成果物にどのような影響を及ぼすのかの研究, ビルド活動の支援手法の研究が多く存在する[1], [2]. 企業だけでなく, 学生による SDPBL におけるビルド活動の実態調査も行われている[3]. 学生のビルドエラーを調査, 分類することにより, 教員が指導する時の手助けになることがわかっている. ソフトウェア開発初学者にとって開発を行う時に, どのような種類のエラーが出ているのかを理解せずに開発を進めることは良い学習とは言えない. また, WF 型の SDPBL においては, 上流工程で作成される成果物(データベース(以下 DB)設計書, シーケンス図等)と下流工程の成果物(ソースコード)の一貫性を確認することも重要である. 実際に我々の対象としている SDPBL で学生が作成した成果物の一貫性が取れていない事例も確認している. 本研究では, WF 型の SDPBL において, チームによるビルドエラーがどのようなものかを調査, 分類する. 特に, WF 型の SDPBL において, 上流工程と下流工程の成果物の一貫性がどれだけ確保されているのかを調査することを目的とする.

2. 関連研究

Seo ら[1]は, Google 社における Java, C++プログラマーのビルドエラーを調査した. Google 独自のビルド環境におけるビルドエラーを収集し, 分類している. 結果として, 依存関係に起因するビルドエラーが多く発見された. しかしこれは企業の研究であり学生, 主に初学者を対象とした SDPBL において同様のことが言えるとは限らない.

また, 槇原ら[3]はアジャイル開発を取り入れた SDPBL においてビルドエラーの実態調査を行い, 学生が陥りやすいビルドエラーの特徴を分析した. 調査によりビルドエラーがリモート開発環境に依存するものなど, 未然に防げるものを発見した. しかしこれは, アジャイル開発の SDPBL を対象とした調査であり, WF 型の SDPBL においても同様の結果とは限らない. WF 型の SDPBL 特有のエラーが存在する可能性もある.

3. リサーチクエスチョン

本研究の目的は, WF 型の SDPBL における学生のビルドの特徴を明らかにすることである. WF 型の SDPBL において, 上流工程と下流工程の成果物の一貫性はとても重要である. そこで以下に本研究の目的を達成するために, リサーチクエスチョンとそれぞれの概要を示す.

3.1 RQ1:発生したビルドエラーはどんな種類があるか

ビルドエラーについて, チーム開発では個人で行うビルド活動(以下ローカルビルド)とリモート環境下で行うビルド活動(以下リモートビルド)が行われる. これらのビルド活動におけるビルドエラーを分類する. 分類の種類として, Seo ら[1]また槇原ら[3]による分類を参考に, Dependency, Syntax, Type Mismatch, Semantic, Environment(それぞれ C1～C5)とした. また本研究では, 上流工程と下流工程の成果物の一貫性についてもビルド時のエラーとしたいため, 新たに Consistency(C6)を定義する. 本論文で述べるビルドエラーは C6 を含んだ C1～C6 を指す. Consistency の定義を以下に述べる.

Consistency:上流工程の成果物(DB 設計書, シーケンス図等)と, 下流工程の成果物(ソースコード)の一貫性が確認できないものとして分類する. C6 のビルドエラーの取得はテストの記述により実現する.

C1~C6 をもとに得たビルドエラーを分類する.

3.2 RQ2:発生したビルドエラーの種類ごとの数と割合はどうか

RQ1 によって分類したビルドエラーの数と割合を求める. 数と割合はチームごと, また個人によるものを収集する. これにより WF 型の SDPBL において陥りやすいビル

†1 東京学芸大学
†2 日本工業大学
†3 早稲田大学

ドエラーが明らかになる.

3.3 RQ3:上流工程の成果物と下流工程の成果物の一貫性は取れているのか

チームによる WF 型の SDPBL で, 上流工程と下流工程の成果物の一貫性が取れているのかを確認する. 3.1 で述べたビルドエラーの分類の C6 に該当するものを詳しく調査する. 具体的な調査内容としては, 次の 3 種類(I~III)である.
(I)エラー全体に占める C6 の割合
(II)エラーを起こした際にその上流工程の成果物の作成者と下流工程の成果物の作成者との関係
(III)ビルドエラー時に C6 が原因で他のビルドエラー(C1～C5)を引き起こしているものの割合
(I)で, エラー全体に占める C6 の割合を調査することにより, WF 型の SDPBL において, C6 がどの程度発生しうるものかを開発者および, 指導者が理解することができる.
(II)について, 表 1 にその組み合わせを記す.

表 1:上流工程と下流工程の成果物の作成者との関係

エラー率(X+Y=1)	上流工程と下流工程の製作者
X	同一人物による関係
Y	異なった人による関係

この結果により, チームで開発する上で成果物に関する情報の共有ができているのかが把握できる.
(III)で, C6 のビルドエラーを確認した際に, C1～C5 が共存していたとき, C6 のエラーが原因で C1~C5 が発生したものは全体の何割程度なのかを調査する. ここで述べた全体とは C6 のみが出たビルドエラーを除いたものを指す.
これらを求めることにより, 開発者または指導者は一貫性のミスの把握を行うことができる.

4. 研究のアプローチ

4.1 ビルドログの取得方法

本節では, 前節で述べた RQ を明らかにするためのビルドエラーの取得方法を述べる. 本研究では, ビルドログの取得はローカルビルドのログと, サーバー上で行うリモートビルドのログの 2 種類を取得する. ローカルビルドログの取得方法の手順を図 1 に示す.

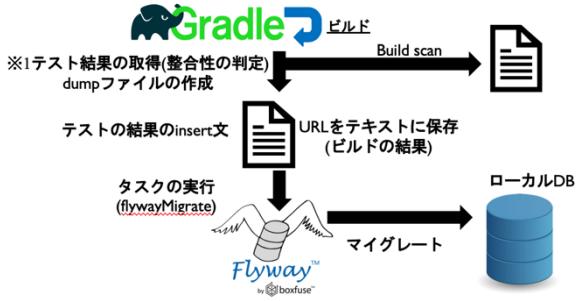

図 1:ローカルビルドのビルドログを DB に格納する手順

ビルドはビルドツール Gradle[4]を用いていくつかのタスクを実行する. ビルド実行時に Java のマイグレーションツールの flyway[5]を用いてビルドログの DB へのマイグレーションを自動で行う. これは Gradle のスクリプトファイルに, 自動で実行するタスクを記述し実現する. この手順をローカルビルドに適用する. リモートビルドでは, CI ツール Jenkins[6]のログから取得する.

4.2 一貫性の調査方法

本研究で扱う C6 で, 一貫性を確認するものは(1)DB 設計書(上流工程)とソースコード(下流工程), (2)シーケンス図(上流工程)とソースコード(下流工程)の 2 つを対象とする. (1)では DB 設計書に記載されているテーブル名とそのテーブル内のカラムが, ソースコード内の Data Access Object(DAO)で, DB に接続する際に接続するテーブル名, カラムと一致しているのかのテストを記述する. そのテスト結果を DB に格納する. 成功か失敗か, 誰の作成物(自分かそうでないか)かの判断ができるようにする. これはリモートのビルド時に確認する. (2)では, シーケンス図どおりにソースコードが記述されているのかのテストを行う. これも(1)と同様に, DB に検証結果を格納する. 図 2 は, 図 1 の※1 で実行するビルド時のシーケンス図においての一貫性が取れているのかのワークフローを示す.

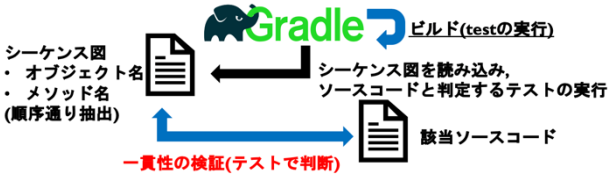

図 2:シーケンス図の一貫性を確認するワークフロー

シーケンス図から取得するオブジェクト名(クラス, jsp などの画面)とメソッド名が正しい順序で実行されているのかを, シーケンス図のワークフローに沿って条件で分類し判断する. 判断する条件として, view→クラスの呼び出し, クラス→クラスの呼び出し, DAO クラスから DB へのアクセス, クラス→view の呼び出しの 4 つを想定している. 今回対象としている SDPBL は jsp&サーブレット技術を用いたものを想定しており, シーケンス図において考えられるものを抽出した.

5. おわりに

本論文では, WF 型の SDPBL において, 上流工程と下流工程の成果物の一貫性を確認する手法の提案をした. 今後は実際の SDPBL において, 本手法を実践する.

謝辞

本研究は科研費基盤研究(B)16H02804, 基盤研究(C)18K11579 の助成を受けて行っている. 記して謝意を表す.

参考文献

[1] Seo H. et al.. Programmers' Build Errors: A Case Study (at Google), Proc. 36th International Conference on Software Engineering, 2014 pp.724–734.
[2] Adams B. et al.. Design recovery and maintenance of build systems, Proc. 23th International Conference on Software Main- tenance, pp.114–123.
[3] 槇原絵里奈他. ソフトウェア開発 PBL におけるビルドエラーの調査, 2017, 情報処理学会論文誌, pp 871 – 884.
[4] "Gradle" .http://gradle.org/ (参照 2019-7-4) .
[5] "flyway" .https://flywaydb.org/ (参照 2019-7-4) .
[6] "Jenkins" .https://jenkins.io/ (参照 2019-7-4) .