GitHub上のOSSを対象とするSBOMデータセット構築の試み

ー SBOM生成ツールの利用に関する問題点の調査 ー

岸本 理央[1]　神田 哲也[2]　眞鍋 雄貴[3]　井上 克郎[4]
仇 実[5]　肥後 芳樹[1]

[1] 大阪大学 大学院情報科学研究科
[2] ノートルダム清心女子大学 情報デザイン学部情報デザイン学科
[3] 福知山公立大学 情報学部情報学科
[4] 南山大学 理工学部ソフトウェア工学科
[5] 株式会社東芝 デジタルイノベーションテクノロジーセンター

背景｜ソフトウェアの管理の必要性

ライブラリが広く利用されている

- ソフトウェアの開発期間の短縮や開発費用の削減
- 推移的な依存も含めると多くのライブラリに依存

ソフトウェアの依存の管理は不十分である

- 脆弱性への対応の遅れや対応漏れ
- ライセンス違反の発生

適切な管理のためにソフトウェア部品表（SBOM）の利用が推奨される

背景｜ソフトウェア部品表（SBOM）

ソフトウェアに関するドキュメント

　開発者が作成し，ソフトウェア本体とともに提供する

主要なフォーマットはSPDXとCycloneDX

ソフトウェア部品表（Software Bill of Materials, SBOM）

ソフトウェアコンポーネント（ライブラリ等）の情報

- 名前
- バージョン
- ライセンス
- 提供者 etc.

コンポーネント間の関係

SBOMの作成者

SBOMの作成日時

背景｜SBOMに記述される情報

例：SPDX形式のSBOMにおけるライブラリの情報の記述

```
{
    "name": "log4net",
    "SPDXID": "SPDXRef-Package-log4net",
    "licenseConcluded": "Apache-2.0",
    "versionInfo": "2.0.8",
    "externalRefs": [{
        "referenceCategory": "PACKAGE-MANAGER",
        "referenceType": "purl",
        "referenceLocator": "pkg:nuget/log4net@2.0.8"
    }],
    "checksums": [{
        "algorithm": "SHA1",
        “checksumValue": "40fdba136f864c8a2f3e3f…"
    }], …
}
```

ライブラリ名

ライセンス

バージョン

一意な識別子

チェックサム

背景｜SBOM生成ツールと課題

SBOMの作成には依存関係の情報が必要

- 手作業で情報を収集するのは容易ではない

SBOM生成ツール

- ソースコードなどから情報を自動で収集
- SBOMの作成作業を省力化

SBOM生成ツールは未成熟

- 生成されるSBOMが正確でないことがある
- SBOMの普及を阻害する要因の1つ

背景｜SBOMのデータセット

ソフトウェア工学の他分野ではデータセットが存在

- ツールやアルゴリズムの評価におけるベンチマーク
- 例：Semantic Clone Bench（コードクローン群）

SBOM分野にはデータセットが存在しない

- SBOMのサンプルファイルは存在するが，
  ツールの評価を目的とした整理はされていない
- ツールの改善には，データセットの存在が望ましい

SBOMデータセットの要件

依存関係の網羅性

ソフトウェアのすべての依存関係に関する情報を含む

完全性

依存するコンポーネント（ライブラリ等）について
ツールの評価に必要な情報がすべて記述されている

正確性

依存するコンポーネント（ライブラリ等）について
記述されている情報が正しい

十分な規模

様々なエコシステム（プログラミング言語）の
ソフトウェアを対象に作成されたSBOMを含む

SBOMデータセットの構築方針

要件を満たすSBOMを可能な限り自動生成する

- 既存のSBOM生成ツールを活用
- 自動化が難しいビルド作業をせずに必要な情報を収集

SBOMの種類	定義
Design	設計段階に作成され，利用が想定される要素の情報を含む
Source	**開発環境，ソースファイル，ビルド時の依存関係情報から直接作成される**
Build	リリース可能な成果物をビルドする過程で生成される
Analyzed	**成果物を解析することで生成される**
Deployed	設定ファイルの内容やデプロイ環境での動作の分析結果などに基づいて生成される
Runtime	実行中に使用されているコンポーネントの情報を収集することで生成される

SBOM生成ツールをデータセット構築に利用するための課題調査

SBOM生成ツールを用いてSBOMを生成

データセット構築に利用する場合の課題を調査

1. 生成対象となるソフトウェアが十分に存在するか
2. 既存の生成ツールが出力するSBOMを活用できるか

SBOMの生成対象

- JavaまたはC#を用いているGitHub上のOSS

調査するSBOM生成ツール

- Sourceタイプ・Analyzedタイプの
  SPDX形式のSBOMを生成するツール

調査方法｜Javaを対象とするSBOM生成

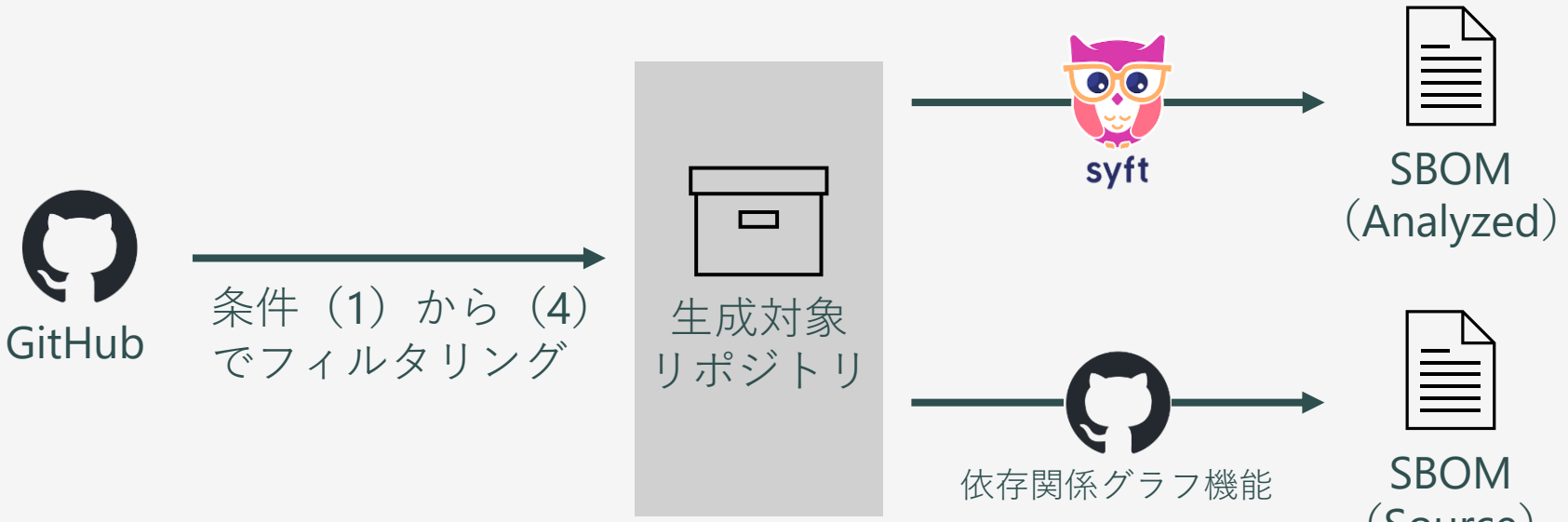
GitHubから生成対象のソフトウェアを選択

（1）使用されているプログラミング言語がJava

（2）2023年以降にコミットが存在

（3）アーカイブされていない

（4）スター数が1000以上



調査結果（Java）｜SBOMの生成対象

GitHubのRelease機能で公開されているバイナリをAnalyzedタイプのSBOM生成時の解析対象とした

- 220リポジトリ（6.88%）がバイナリを提供

生成対象の選択条件のスター数下限を下げることで十分な数のSBOM生成対象を確保可能

条件	リポジトリ数	割合
（1）から（4）	3,198	100.00%
ビルド済みバイナリをGitHubのRelease機能で提供	**220**	**6.88%**
├ Javaのアーカイブファイル	108	3.38%
└ Javaのアーカイブファイルを含むzipファイル	121	3.78%

調査結果（Java）｜生成ツール間の比較

依存関係の網羅性

Syft	直接的	推移的
実行時	○	○
開発時	×	×

GitHub	直接的	推移的
実行時	○	×
開発時	○	×

完全性・正確性

Syft：記述された情報は正確だが改善の余地がある

GitHub：静的な解析では得られない情報が不足

	完全性	正確性
Syft	○	△
GitHub	×	○

調査結果（Java）｜Syftによる生成の例

```
{
  "name": "caffeine",
  "SPDXID": "SPDXRef-Package-java-caffeine...",
  "versionInfo": "2.9.3",
  "checksums": [{
    "algorithm": "SHA1",
    "checksumValue": "b162491..."
  }],
  "licenseDeclared": "LicenseRef-https---www.apache.org-licenses-LICENSE-2.0",
  "externalRefs": [{
      "referenceCategory": "PACKAGE-MANAGER",
      "referenceType": "purl",
      "referenceLocator": "pkg:maven/com.github.ben-manes.caffeine/caffeine@2.9.3"
  } ...], ...
}
```

チェックサム
SHA1で計算された値のみ
対応する複数のアルゴリズムで記述すべき

ライセンス
ライセンス識別子を用いていない
（例：Apache-2.0）

調査結果（Java）｜GitHubによる生成の例

```
{
  "SPDXID": "SPDXRef-maven-com.github.ben-ma...",
  "name": "maven:com.github.ben-manes.caffe...",
  "versionInfo": "",
  ...
}
```

バージョン
範囲指定されていたため
バージョンを特定できず空文字列

ライセンス
- バージョンが特定できない場合は記述されない
- バージョンが特定できる場合はライセンス識別子を用いて記述される

チェックサム，ライブラリの一意な識別子
- 常に記述されない

調査結果（Java）｜考察

生成ツール毎に生成したSBOMが含む情報が異なる

- 単一のツールでは要件を満たすSBOMを作成できない

ツールの組み合わせによる情報の補完が必要

- 複数のツールを組み合わせて不足する情報を補う
- SBOM生成ツールのみで情報が不足する場合は，その他のツールも利用する

調査方法｜C#を対象とするSBOM生成

GitHubから生成対象のソフトウェアを選択

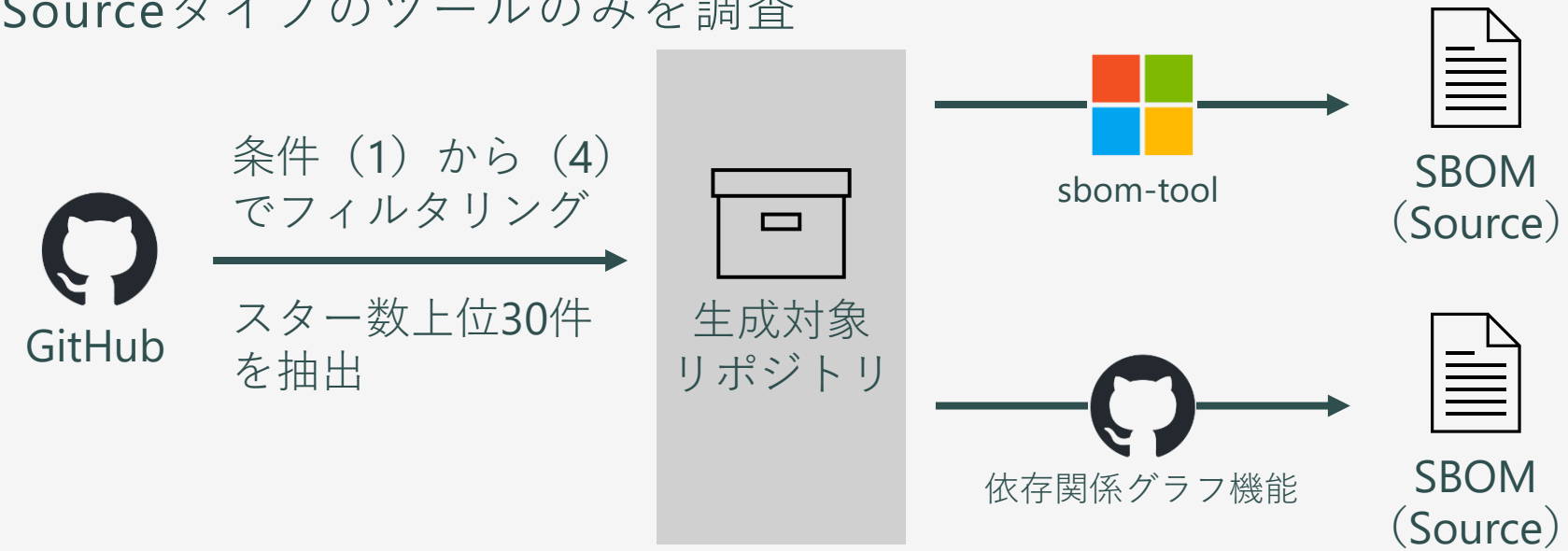
（1）使用されているプログラミング言語がC#

（2）2023年以降にコミットが存在

（3）アーカイブされていない

（4）サンプルコードやドキュメント以外

Analyzedタイプの生成ツールが存在しなかったため，
Sourceタイプのツールのみを調査



調査結果（C#）｜SBOMの生成対象

sbom-toolでSBOMを生成するためには
事前に「依存関係の復元」を行う必要がある

- バージョンが範囲指定されたライブラリについて
  使用するバージョンが決定される
- 30個中19個（63%）のソフトウェアで成功

十分な数のSBOM生成対象を確保するうえで
大きな問題にはならないと予想される

調査結果（C#）｜生成ツール間の比較

依存関係の網羅性

sbom-tool	直接的	推移的
実行時	○	○
開発時	△	△

GitHub	直接的	推移的
実行時	○	×
開発時	○	×

完全性・正確性

sbom-tool：ライセンスとチェックサムの情報が不足

GitHub：静的な解析では得られない情報が不足

	完全性	正確性
sbom-tool	×	○
GitHub	×	○

調査結果（C#）｜sbom-toolによる生成

```
{
  "name": "ReactiveUI.WPF",
  "SPDXID": "SPDXRef-Package-A0E054779263F...",
  "licenseDeclared": "NOASSERTION",
  "versionInfo": "19.6.1",
  "externalRefs": [{
      "referenceCategory": "PACKAGE-MANAGER",
      "referenceType": "purl",
      "referenceLocator": "pkg:nuget/ReactiveUI.WPF@19.6.1"
  }], ...
}
```

ライセンス
情報なし（NOASSERTION）

チェックサム
記述なし

調査結果（C#）｜GitHubによる生成

```
{
  "SPDXID": "SPDXRef-nuget-ReactiveUI.WPF-20.1.1",
  "name": "nuget:ReactiveUI.WPF",
  "versionInfo": "20.1.1",
  "externalRefs": [{
      "referenceCategory": "PACKAGE-MANAGER",
      "referenceLocator": "pkg:nuget/ReactiveUI.WPF@20.1.1",
      "referenceType": "purl"
  }], ...
}
```

ライセンス，チェックサム
記述なし

調査結果（C#）｜考察

Javaと同様に生成ツール毎に含む情報が異なる

ツールの組み合わせによる情報の補完が必要

- 複数のツールを組み合わせて不足する情報を補う
- SBOM生成ツールのみで情報が不足する場合は，その他のツールも利用する

調査のまとめ

SBOMの生成対象

Java，C#ともに十分数のSBOM生成対象を確保できる

既存の生成ツールの活用

JavaとC#のいずれの場合も単一の生成ツールでは
データセットの要件を満たすSBOMは作成できない

複数のツールを組み合わせることで，
不足する情報を補うことが可能

今後の課題・展望

他のソフトウェアエコシステムを対象とした調査

- エコシステム毎にSBOMの作成における課題は異なる可能性がある
- PythonやJavaScriptなどを対象として同様の調査を行う

複数のツールから得た情報を組み合わせて
要件を満たすSBOMを作成する具体的な方法の検討

- どのツールを利用するのか
- 各ツールから得たどの情報を信頼するのか