SPDX を用いたソフトウェアパッケージの概要表示法

2019SE069 安田簡
指導教員：井上克郎

1　はじめに

SPDX(Software Bill Of Materials，エスボム) はその重要性が高まっているソフトウェア部品表 (SB Software Bill Of Materials) のひとつで，多くのソウェアパッケージで利用が始まっている [1, 2]．SPDX組織やコミュニティにとって重要なデータを共有すめの共通の形式を提供する．それによってコンプライス，セキュリティ、および信頼性を向上させることがされている．

本研究では SPDX ファイルを要約し見やすくするで，SPDX に詳しくない一般の技術者にとっても簡その概要内容を把握出来るようにすることを目的とする．本研究は，将来的に SPDX の簡約化を効率よくうプログラムのプロトタイプとして機能する．

2　SBOM と SPDX の概要

SBOM はソフトウェア部品表と言い，ソフトウェアパッケージの構成要素（ライブラリや他のパッケージ）やそれらの依存関係を一覧できるリストのことを指す．SBOM を参照することで脆弱性への対応やリスク管理が簡単になることが期待される．

SPDX は，Linux Foundations が定めて普及を促進している SBOM を実現するためのひとつの形式である [3]．SPDX は，対象となるソフトウェアパッケージの開発者名やライセンス，含まれるファイルやそれらの依存関係などの情報が，JASON 形式などを用いて標準化され，記述されている [2]．

SPDX は，1.Document Creation Informatio，2.Package Information 3.File Information，4.Checksum Information，5.License Information，6.Relationships，7.Review Information 8.Annotation Information の８つの要素から構成される．

3　SPDX 概要表示法の提案

SPDX ファイルは多くの情報を記載できるため長くなることが多い．小さなパッケージを対象としたものでも，数百行や数千行にも及ぶことがある．そのため SPDX を見て，対象のパッケージの全体像を理解することは容易ではない．そこで本研究では文献 [1] に述べられている食品に対する食品表示法にヒントを得て，SPDX の概要表示法を行う方法を提案する．これは食品ラベルに似た簡潔な表示をすることで SPDX に精通していない利用者にとっても，必要な情報を簡潔に把握できることを目的としている．

図１は，hello という簡単なパッケージに対する SPDX を json 形式で表したものの一部である．図２は本研究で行うそれの簡約化した表示である．表示するものは，SPDX バージョン，SPDXID，作成日，作成者，SPDX 作成ツール，ファイル一覧の６種類の情報である．これらは１.Document Creation Information と３.File Information の一部（ファイル名）を抜粋して得ている．

```
"spdxVersion": "SPDX-2.2",
"dataLicense": "CC0-1.0",
"SPDXID": "SPDXRef-DOCUMENT",
"name": "hello 0.0.1",
"documentNamespace": "https://swinslow.net/spdx-examples/example11/hello/hello/0.
"creationInfo": {
  "created": "2022-11-04T11:11:52Z",
  "creators": [
    "Organization: DocFest",
    "Tool: Microsoft.SBOMTool-0.2.7"
          . . .
 "hasFiles": [
   "SPDXRef-File--Gemfile.lock-DD14EBD399202141553C772590FE3977CE39F36C",
   "SPDXRef-File--Gemfile-D95216A381538DE969C759A32A427D8EB437E532",
   "SPDXRef-File--sig-hello.rbs-F11ED3C48976FA5EFF33CEA9EEF89DC36DB3D68C",
   "SPDXRef-File--CHANGELOG.md-60D67ED18C1946A93A47B7D30FF010024C843A16",
   "SPDXRef-File--LICENSE.txt-42D586FD523FE822C198E99AF5866BC26BBA4EB5",
   "SPDXRef-File--bin-setup-38AE71152A2D22A6B6418E0ADD993DF2268CCE81",
   "SPDXRef-File--lib-hello-version.rb-B5BE44EF7A6F7F6DED5CC28EC27BD3CB9A67CCB2",
   "SPDXRef-File--Rakefile-7D82872A906A22904A3419329427B957EDF26E7F",
   "SPDXRef-File--bin-console-BCED536ADBE99BDE1B1D0D0039F57BA83F10D727",
   "SPDXRef-File--hello.gemspec-7FD905D8BA81BEB7E4194BDEF067E68620C42A91",
   "SPDXRef-File--lib-hello.rb-F84D3A18EAF5AA1D343BE06A8C456B2E2A614D4C",
   "SPDXRef-File--README.md-F5A5B10F3E0FEEAB03D48E9D377692DAD9B7BD52"
```

図１　hello パッケージの SPDX（一部抜粋）

SPDX バージョン	SPDX-2.2
SPDXID	SPDXRef-DOCUMENT
作成日	2022-11-04T11:11:52Z
作成者	DocFest
SPDX 作成ツール	Microsoft.SBOMTool-0.2.7
ファイル一覧	Gemfile.lock，Gemfile，sig-hello.rbs，CHANGELOG.md，LICENSE.txt，Binsetup，libhelloversion.rb，Rakefile，hello.gemspec，lib-hello.rb，README.md

図２　図１の SPDX ファイルの簡約化したもの

このように簡約した理由としては，SPDX ファイル自体の作成者とファイル元となるパッケージ作成者を記すことで誰が作成したか把握しやすくなり，問題が生じた際には容易に責任者を特定することができる．ファイル一覧，バージョン，ツールに関しても同様の理由である．

4　ChatGPT を用いた簡約化

上記のような SPDX ファイルの簡約化を ChatGPT を用いて行うことを考える．SPDX ファイルの情報と簡約化命令を ChatGPT にプロンプトとして送り，その結果の問題点を改善するためにプロンプトを送る．これを繰り返し行うことで見やすく簡約化された表示を目指す．

簡約化する SPDX の対象のパッケージはソースコードやライブラリ，バイナリなども対象とする．また，ここで用いる SPDX は，すべて github 上にある SPDX サンプルファイル（spdx-example1～spdx-example12）の中から使用する．

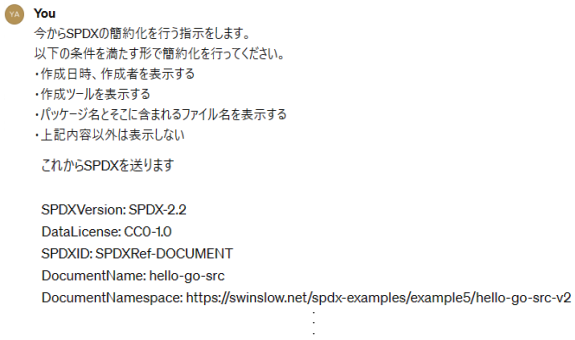

図 3　1 回目の指示

図 4　1 回目の指示の結果

今回，簡約化は以下の方針で行なった．

- できるだけプロンプトは短く，また，少ない対話回数にする
- 簡約化の方針で挙げた 6 種類の要素を表示する
- 簡約化した情報は，見やすい表形式にする．また，その表示順は一定になるようにする

ChatGPT は文書の要約やプログラム理解やリファクタリングなどで実績があるが，SPDX 記述など特定の分野で用いられる形式的な記述の意味理解や要約に関してはその効果は不明であった．しかし，各種のプログラミング言語に対する実績をもとに，SPDX においても意味の理解や要約に用いることができるのではないかと考えた．

5　簡約化の例

ChatGPT を用いて，サンプルの SPDX を簡約化した例を示す．ここではプロンプトを 2 回与えて，安定した結果が得られた．

図 3 は，1 回目の指示である．ここでは，表示すべき項目と，対象となる SPDX 全体を送っている．図 4 はその結果である．パッケージ名や SPDX のバージョン，SPDXID などは正しく抽出されているが，不要な Data License や Document Namespace などの情報が含まれている．

図 5 は，2 回目の指示とその結果である．ここでは，表示すべき項目を再度指示している．その結果，不要な情報は削除されて表示されるようになった．ただし，本来希望していた表形式による表示ではなく，リスト形式で表示されている．

同様な対話で簡約化できるか検証のために，20 種類の SPDX ファイルを簡約化したところ，19 個が 2 回のやり取りで簡約化に成功した．

6　考察とまとめ

ChatGPT を用いる簡約化方法では，ひとつひとつのファイルをコピーペーストし，命令を 2 回行う必要がある．また，字数上限になった場合，簡約化を行えない，大量に簡約化が必要になった際に時間効率が極端に悪いというデメリットがみられる．仮に業務などに組み込むとするならば，自動簡約化プログラムを用いて自動的に大量のファイルを取り扱える環境が最適であろう．ChatGPT も手動も，ケアレスミスなど人間特有の間違いが絡む可能性があり，不確定要素も多い．

図 5　2 回目の指示とその結果

今後 SPDX が日常的に普及し，SPDX に接する機会が増えるので，CharGPT により簡約化された SPDX でその概要が理解できるようになることは重要なろう．

参考文献

[1] 経済産業省商務情報政策局サイバーセキュリティ課：ソフトウェア管理に向けた SBOM（Software Bill of Materials）の導入に関する手引 Ver. 1.0, https://www.meti.go.jp/press/2023/07/20230728004/20230728004-1-2.pdf.

[2] Linux Foundation and its Contributors：A common software package data exchange format，2.0 edition, 2015.

[3] National Telecommunications and Information Administration：The minimum elements for a software bill of materials (sbom), https://www.ntia.doc.gov/files/ntia/publications/sbomminimumlements report，2021-7.