(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7322963号
(P7322963)

(45)発行日　令和5年8月8日(2023.8.8)

(24)登録日　令和5年7月31日(2023.7.31)

(51)国際特許分類

Ｇ０６Ｆ	*8/77*	*(2018.01)*	Ｇ０６Ｆ　8/77
Ｇ０６Ｆ	*21/57*	*(2013.01)*	Ｇ０６Ｆ　21/57

ＦＩ

請求項の数　9　（全25頁）

(21)出願番号　特願2021-553261(P2021-553261)
(86)(22)出願日　令和1年10月25日(2019.10.25)
(86)国際出願番号　PCT/JP2019/041929
(87)国際公開番号　WO2021/079496
(87)国際公開日　令和3年4月29日(2021.4.29)
審査請求日　令和4年4月18日(2022.4.18)

(73)特許権者　000004237
日本電気株式会社
東京都港区芝五丁目７番１号
(74)代理人　100141519
弁理士　梶田 邦之
(72)発明者　西岡 淳
東京都港区芝五丁目７番１号 日本電気株式会社内
(72)発明者　榮 純明
東京都港区芝五丁目７番１号 日本電気株式会社内
(72)発明者　磯山 和彦
東京都港区芝五丁目７番１号 日本電気株式会社内
(72)発明者　小林 佑嗣

最終頁に続く

(54)【発明の名称】　評価装置、評価方法及びプログラム

(57)【特許請求の範囲】

【請求項１】

ソースコードに記載された第１のライブラリについての互いに性質が異なる情報に基づいて、前記第１のライブラリのリスクに関する評価値を生成する、生成手段と、

前記第１のライブラリが前記ソースコードの全体に与える影響を示す影響度を算出する、算出手段と、

前記異なる情報ごとに生成された評価値の合計値と前記算出された影響度とに基づき、前記第１のライブラリの危険度を算出し、前記算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、前記算出されたリスク値の時系列データを出力する、出力手段と、

を備え、

前記生成手段は、前記ソースコードに前記第１のライブラリから呼び出される第２のライブラリが含まれる場合には、前記第２のライブラリについての互いに性質が異なる情報に基づいて、前記第２のライブラリのリスクに関する評価値を生成し、

前記出力手段は、前記第２のライブラリに関して生成された前記異なる情報ごとの評価値のうち値の最も大きい評価値を特定し、前記第１のライブラリに関して生成された評価値の合計値に前記特定された評価値を加算することで、前記第１のライブラリの危険度を算出する、評価装置。

【請求項２】

前記出力手段は、前記評価値の合計値に前記影響度を乗算することで、前記第１のライ

ブラリの危険度を算出する、請求項１に記載の評価装置。
【請求項３】
前記生成手段は、前記第１のライブラリに関する静的な情報に基づき静的評価値を生成し、前記第１のライブラリに関する動的な情報に基づき動的評価値を生成し、前記第１のライブラリのソースコードに基づき実体評価値を生成する、請求項１または２に記載の評価装置。
【請求項４】
前記出力手段は、前記ソースコードに含まれるライブラリについて算出された危険度の合計値を前記ソースコード全体のリスク値として算出し、前記算出されたソースコード全体のリスク値の時系列データを表示する、請求項１乃至３のいずれか一項に記載の評価装置。
【請求項５】
前記出力手段は、前記第１のライブラリの危険度の時系列データを表示する、請求項４に記載の評価装置。
【請求項６】
前記出力手段は、リスク値の時系列データを表示する項目を切り替えるためのＧＵＩ（Graphical User Interface）を生成する、請求項５に記載の評価装置。
【請求項７】
前記算出されたリスク値が所定の条件を満たす場合に、前記ソースコードにリスクが含まれる旨を外部に通知する、通知手段をさらに備える、請求項１乃至６のいずれか一項に記載の評価装置。
【請求項８】
評価装置において、
ソースコードに記載された第１のライブラリについての互いに性質が異なる情報に基づいて、前記第１のライブラリのリスクに関する評価値を生成するステップと、
前記第１のライブラリが前記ソースコードの全体に与える影響を示す影響度を算出するステップと、
前記異なる情報ごとに生成された評価値の合計値と前記算出された影響度とに基づき、前記第１のライブラリの危険度を算出し、前記算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、前記算出されたリスク値の時系列データを出力するステップと、
を含み、
前記生成するステップでは、前記ソースコードに前記第１のライブラリから呼び出される第２のライブラリが含まれる場合には、前記第２のライブラリについての互いに性質が異なる情報に基づいて、前記第２のライブラリのリスクに関する評価値を生成し、
前記出力するステップでは、前記第２のライブラリに関して生成された前記異なる情報ごとの評価値のうち値の最も大きい評価値を特定し、前記第１のライブラリに関して生成された評価値の合計値に前記特定された評価値を加算することで、前記第１のライブラリの危険度を算出する、評価方法。
【請求項９】
評価装置に搭載されたコンピュータに、
ソースコードに記載された第１のライブラリのリスクについての互いに性質が異なる情報に基づいて、前記第１のライブラリに関する評価値を生成する処理と、
前記第１のライブラリが前記ソースコードの全体に与える影響を示す影響度を算出する処理と、
前記異なる情報ごとに生成された評価値の合計値と前記算出された影響度とに基づき、前記第１のライブラリの危険度を算出し、前記算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、前記算出されたリスク値の時系列データを出力する処理と、
を実行させ、

前記生成する処理では、前記ソースコードに前記第１のライブラリから呼び出される第２のライブラリが含まれる場合には、前記第２のライブラリについての互いに性質が異なる情報に基づいて、前記第２のライブラリのリスクに関する評価値を生成し、
前記出力する処理では、前記第２のライブラリに関して生成された前記異なる情報ごとの評価値のうち値の最も大きい評価値を特定し、前記第１のライブラリに関して生成された評価値の合計値に前記特定された評価値を加算することで、前記第１のライブラリの危険度を算出する、プログラム。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、評価装置、評価方法及びプログラムに関する。
【背景技術】
【０００２】
近年、ソフトウェア（アプリケーションプログラム）の規模は増加の一途であり、全てのソースコードを自前で開発することは困難な状況にある。そのため、オープンソースソフトウェア（ＯＳＳ；Open Source Software）等と称される第３者が開発したソースコードを利用する開発が盛んに行われている。例えば、アプリケーションプログラムの一部機能が、ＯＳＳとして公開されているライブラリにより実現される。
【０００３】
特許文献１には、並行開発で予測不能な任意のタイミングで依存関係が生成されるような開発工程における変更影響把握の困難さを解決する、と記載されている。特許文献２には、ソフトウェアの構造的な障害の発生箇所を予測することができる危険度判定プログラムを提供する、と記載されている。特許文献３には、ソフトウェア開発、保守、および修復ライフサイクルにおける重要な側面を自動化する、大量のソフトウェアファイルを活用することが可能なシステム及び方法を提供する、と記載されている。特許文献４には、ソフトウェア開発効率の低下とソフトウェアエージングの検出精度の低下を招かないような適切な頻度で負荷テストを実行する、と記載されている。
【先行技術文献】
【特許文献】
【０００４】
【文献】特開２００８－０４０７６０号公報
特開２０１０－２５７０９１号公報
特表２０１７－５２０８４２号公報
国際公開第２０１５／０２２８９０号
【発明の概要】
【発明が解決しようとする課題】
【０００５】
上述のように、アプリケーションプログラムの開発にオープンソースソフトウェアとして公開されたライブラリが用いられることがある。
【０００６】
ライブラリをアプリケーションプログラムに組み込み際、通常、当該ライブラリに対する検証が行われる。具体的には、使用予定のライブラリに関する脆弱性情報の有無等やソースコードに組み込んだ場合の振る舞いが検証される。
【０００７】
また、ライブラリ自身も他のライブラリに依存（他のライブラリを呼び出し）するなど多段の依存関係を有し、ライブラリの持つ複雑度が増している。そのため、ライブラリ自身がバグやセキュリティホール等の脆弱性を有していることも珍しくない。
【０００８】
ここで、使用予定のライブラリに関し、その導入時には念入りに当該ライブラリの検証が行われるが、導入後には動作確認等の簡単な確認に留まることが多い。即ち、アプリケ

ーションプログラム自体の開発に重点が置かれ、導入後のライブラリに対する検証は重要視されないことも多い。

【０００９】

しかし、ライブラリを使用する側（呼び出し側）のソースコードはプロジェクトの進行と共に改版され、ライブラリの使用態様も変化する。例えば、ライブラリの導入時には１つの関数が当該ライブラリを呼び出すに過ぎなかったものが、開発の進行と共に複数の関数が上記ライブラリを呼び出すこともある。

【００１０】

この場合、ライブラリの脆弱性がアプリケーションプログラム全体に及ぼす影響が拡大していることになる。しかし、上述のように、ライブラリの導入時にはその検証が念入りに行われるが、ライブラリの導入後には同等の検証が行われることは少ない。その結果、ライブラリの有する脆弱性がシステム（アプリケーションプログラム全体）に及ぼす深刻な影響が見落とされる可能性がある。即ち、ライブラリが有する脆弱性がアプリケーションプログラムに及ぼす影響は時間と共に変化するが、現状のシステム開発体制では当該ライブラリの影響度を適切に評価することができない。

【００１１】

本発明は、時間の経過と共に変化するソースコードのリスクを適切に評価することに寄与する、評価装置、評価方法及びプログラムを提供することを主たる目的とする。

【課題を解決するための手段】

【００１２】

本発明の第１の視点によれば、ソースコードに記載された、第１のライブラリのリスクに関する評価値を生成する、生成部と、少なくとも前記生成された評価値に基づき、前記第１のライブラリの危険度を算出し、前記算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、前記算出されたリスク値の時系列データを出力する、出力部と、を備える、評価装置が提供される。

【００１３】

本発明の第２の視点によれば、評価装置において、ソースコードに記載された、第１のライブラリのリスクに関する評価値を生成するステップと、少なくとも前記生成された評価値に基づき、前記第１のライブラリの危険度を算出し、前記算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、前記算出されたリスク値の時系列データを出力するステップと、を含む評価方法が提供される。

【００１４】

本発明の第３の視点によれば、評価装置に搭載されたコンピュータに、ソースコードに記載された、第１のライブラリのリスクに関する評価値を生成する処理と、少なくとも前記生成された評価値に基づき、前記第１のライブラリの危険度を算出し、前記算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、前記算出されたリスク値の時系列データを出力する処理と、を実行させるプログラムが提供される。

【発明の効果】

【００１５】

本発明の各視点によれば、時間の経過と共に変化するソースコードのリスクを適切に評価することに寄与する、評価装置、評価方法及びプログラムが提供される。なお、本発明により、当該効果の代わりに、又は当該効果と共に、他の効果が奏されてもよい。

【図面の簡単な説明】

【００１６】

【図１】一実施形態の概要を説明するための図である。

【図２】第１の実施形態に係る評価装置を説明するための図である。

【図３】第１の実施形態に係る評価装置の処理構成の一例を示す図である。

【図４】第１の実施形態に係る評価装置が取得するソースコードの一例を示す図である。

【図５】ライブラリ構成情報の一例を示す図である。

【図６】リスク評価値生成部が生成する情報の一例を示す図である。
【図７】ライブラリ評価結果の一例を示す図である。
【図８】ライブラリ影響度算出部が算出するライブラリ影響度の一例を示す図である。
【図９】ライブラリ評価情報の一例を示す図である。
【図１０】記憶部に格納されたリスク値の時系列データの一例を示す図である。
【図１１】第１の実施形態に係る評価装置による表示の一例を示す図である。
【図１２】第１の実施形態に係る評価装置による表示の一例を示す図である。
【図１３】第１の実施形態に係る評価装置による表示の一例を示す図である。
【図１４】第１の実施形態に係る評価装置による表示の一例を示す図である。
【図１５】第１の実施形態に係る評価装置による表示の一例を示す図である。
【図１６】第１の実施形態に係る評価装置による表示の一例を示す図である。
【図１７】第１の実施形態に係る評価装置による表示の一例を示す図である。
【図１８】第１の実施形態に係る評価装置の動作の一例を示すフローチャートである。
【図１９】第２の実施形態に係る評価装置の処理構成の一例を示す図である。
【図２０】第２の実施形態に係る評価装置による表示の一例を示す図である。
【図２１】第２の実施形態に係る評価装置による表示の一例を示す図である。
【図２２】第２の実施形態に係る評価装置による表示の一例を示す図である。
【図２３】評価装置のハードウェア構成の一例を示す図である。
【図２４】ライブラリ評価結果の一例を示す図である。
【発明を実施するための形態】
【００１７】
はじめに、一実施形態の概要について説明する。なお、この概要に付記した図面参照符号は、理解を助けるための一例として各要素に便宜上付記したものであり、この概要の記載はなんらの限定を意図するものではない。なお、本明細書及び図面において、同様に説明されることが可能な要素については、同一の符号を付することにより重複説明が省略され得る。
【００１８】
一実施形態に係る評価装置１００は、生成部１０１と、出力部１０２と、を備える（図１参照）。生成部１０１は、ソースコードに記載された、第１のライブラリのリスクに関する評価値を生成する。出力部１０２は、少なくとも生成された評価値に基づき、第１のライブラリの危険度を算出し、算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、算出されたリスク値の時系列データを出力する。
【００１９】
評価装置１００は、ソースコードに記載された各ライブラリ自身の評価を行い、ライブラリの持つリスクを危険度として数値化する。評価装置１００は、ライブラリの観点からソースコードが有するリスクを可視化し、その時系列変化を表示する。当該表示に接したユーザは、時間の経過と共に変化するソースコードのリスクを適切に評価することができる。
【００２０】
以下に具体的な実施形態について、図面を参照してさらに詳しく説明する。
【００２１】
［第１の実施形態］
第１の実施形態について、図面を用いてより詳細に説明する。
【００２２】
図２は、第１の実施形態に係る評価装置１０を説明するための図である。評価装置１０は、アプリケーションプログラム等のソースコードに内在するリスクを評価する装置である。評価装置１０は、リスクの評価対象となるソースコードを取得する。
【００２３】
評価装置１０は、当該ソースコードが有するリスクを数値化し、当該数値化したリスクをユーザ（アプリケーションプログラムの管理者等）に提示する。その際、評価装置１０

は、ソースコード（当該ソースコードから生成されたプロジェクト）のリスクに関する時系列変化をユーザに提示する。

【００２４】

例えば、評価装置１０は、上記リスクの時系列データを液晶モニタ等に表示する。あるいは、評価装置１０は、当該時系列データを所定のアドレス等に送信してもよいし、プリンター等を用いて印刷してもよい。

【００２５】

図３は、第１の実施形態に係る評価装置１０の処理構成（処理モジュール）の一例を示す図である。図３を参照すると、評価装置１０は、ソースコード取得部２０１と、ライブラリ情報生成部２０２と、リスク情報出力部２０３と、記憶部２０４と、を備える。

【００２６】

ソースコード取得部２０１は、リスク評価の対象となるソースコードを取得する手段である。例えば、ソースコード取得部２０１は、管理者等が評価対象となるソースコードを入力（指定）するためのＧＵＩ（Graphical User Interface）情報を生成し、当該ＧＵＩを用いてソースコードを取得してもよい。あるいは、ソースコード取得部２０１は、ＵＳＢ（Universal Serial Bus）メモリ等の外部記憶装置を介してソースコードを取得してもよいし、外部のデータベースサーバ等にアクセスしてソースコードを取得してもよい。

【００２７】

例えば、ソースコード取得部２０１は、図４に示すようなソースコードを取得する。図４Ａは、メイン関数を含むソースコードの一例を示し、図４Ｂ及び図４Ｃは、ライブラリのソースコードの一例を示す。なお、ソースコード取得部２０１が取得するソースコードには、ソースコードに付随するＲＥＡＤＭＥファイルやソースコードのコンパイルにより生成されるファイル、ソースコードが格納されたディレクトリ等が含まれる。

【００２８】

ソースコード取得部２０１は、取得したソースコードとその識別情報（例えば、プロジェクト名）を対応付けて記憶部２０４に格納する。

【００２９】

ライブラリ情報生成部２０２は、上記取得したソースコードに記載されたライブラリを評価するための情報（以下、ライブラリ評価情報と表記する）を生成する手段である。

【００３０】

図３に示すように、ライブラリ情報生成部２０２は、ライブラリ抽出部２１１と、リスク評価値生成部２１２と、ライブラリ影響度算出部２１３と、を含む。

【００３１】

ライブラリ抽出部２１１は、取得したソースコードに記載されたライブラリを抽出する手段である。ライブラリ抽出部２１１は、ソースコードを走査し、当該ソースコードに記載されたライブラリを抽出する。具体的には、ライブラリ抽出部２１１は、使用ライブラリを宣言する領域（例えば、ｉｍｐｏｒｔ宣言が記載されたファイルの先頭領域）を確認し、当該領域に記載されたライブラリを抽出する。

【００３２】

図４の例では、ライブラリ抽出部２１１は、図４Ａの１行目、２行目に記載されたライブラリ「Ｌ１２３」、「Ｌ３４５」を抽出する。ライブラリ抽出部２１１は、図４Ｂの１行目、２行目に記載されたライブラリ「Ｌ９１１」、「Ｌ９１２」を抽出する。ライブラリ抽出部２１１は、図４Ｃの１行目に記載されたライブラリ「Ｌ８１１」を抽出する。

【００３３】

ソースコードに記載されたライブラリの抽出が終了すると、ライブラリ抽出部２１１は、当該抽出したライブラリ間の依存関係を検出する。具体的には、ライブラリ抽出部２１１は、各ライブラリが使用している（各ライブラリから呼び出している）子ライブラリ（従属ライブラリ）を検出し、ライブラリ間の依存関係（従属関係）を検出する。

【００３４】

ライブラリ抽出部２１１は、当該検出したライブラリ間の依存関係を「ライブラリ構成

情報」としてまとめる。例えば、図４の例では、ライブラリ「Ｌ１２３」には、ライブラリ「Ｌ９１１」、「Ｌ９１２」が含まれる。従って、ライブラリ「Ｌ１２３」とライブラリ「Ｌ９１１」及び「Ｌ９１２」の間には依存関係が認められる。また、ライブラリ「Ｌ９１１」には、ライブラリ「Ｌ８１１」が含まれる。従って、これらのライブラリの間には依存関係が認められる。

【００３５】

ライブラリ抽出部２１１は、上記検出したライブラリ間の依存関係に基づき、図５に示すようなライブラリ構成情報を生成する。

【００３６】

図５を参照すると、ライブラリ「Ｌ１２３」を基準として、当該ライブラリには子ライブラリ「Ｌ９１１」、「Ｌ９１２」が含まれ、さらに、孫ライブラリ「Ｌ８１１」が含まれることが分かる。

【００３７】

なお、図５には、ライブラリの階層として３階層（親、子、孫）の場合が記載されているが、当該階層構造は例示であることは勿論である。評価対象のソースコードが３階層以上の構造を有していてもよいことは当然である。

【００３８】

リスク評価値生成部２１２は、ソースコードに記載されたライブラリのリスクに関する評価値を生成する手段である。リスク評価値生成部２１２は、抽出されたライブラリ自身のリスクを示す評価値を生成する。具体的には、リスク評価値生成部２１２は、ライブラリ単体でのリスクを評価値として数値化する。

【００３９】

なお、以降の説明において、リスク評価値生成部２１２が実施するライブラリ単体のリスク評価を「ライブラリ単体評価」と表記する。

【００４０】

リスク評価値生成部２１２は、様々な観点からライブラリ単体評価を実施する。具体的には、リスク評価値生成部２１２は、各ライブラリを互いに性質の異なる情報に基づき評価し、異なる情報ごとに評価値を生成する。

【００４１】

リスク評価値生成部２１２は、評価対象となるライブラリに関する時間的な変化が少ない安定した指標を用いてライブラリを評価する。例えば、リスク評価値生成部２１２は、評価対象ライブラリのディレクトリ構成、ＲＥＡＤＭＥファイルの有無、テストディレクトリの有無等、静的な情報に基づき評価値を算出する。

【００４２】

例えば、リスク評価値生成部２１２は、評価対象ライブラリのディレクトリ構成がシンプル（例えば、階層数が所定値よりも少ない等）であれば小さな評価値を与え、ディレクトリ構成が複雑であれば大きな評価値を与える。当該評価値の決定は、ディレクトリ構成が複雑であれば対応するライブラリのリスク（危険度）は相対的に高くなるという知見に基づく。

【００４３】

リスク評価値生成部２１２は、ＲＥＡＤＭＥファイルやテストディレクトリが存在すれば小さな評価値を与え、ＲＥＡＤＭＥファイルやテストディレクトリが非存在であれば大きな評価値を与えてもよい。当該評価値の決定は、ＲＥＡＤＭＥファイルやテストディレクトリが存在すれば、対応するライブラリの管理や評価（デバッグ）が十分である判断できるためである。

【００４４】

リスク評価値生成部２１２は、上記項目（例えば、ディレクトリ構成、ＲＥＡＤＭＥファイルの有無、テストディレクトリの有無）の全部又は一部を用いて評価値を算出する。

【００４５】

なお、以降の説明において、ディレクトリ構成等の静的な情報に基づき定められた評価

値を「静的評価値」と表記する。
【００４６】
リスク評価値生成部２１２は、時間的に変化する情報（例えば、評価対象ライブラリに関するメタデータ）を用いてライブラリを評価してもよい。例えば、リスク評価値生成部２１２は、評価対象ライブラリに関する「問題解決率（Ｉｓｓｕｅ解決率）」、「コミット数」、「コード生成者（作者）」等のメタデータに基づき評価値を算出する。
【００４７】
例えば、リスク評価値生成部２１２は、Ｉｓｓｕｅ解決率やコミット数に対して、閾値処理を実行し、その結果に応じて評価値を算出してもよい。より具体的には、リスク評価値生成部２１２は、Ｉｓｓｕｅ解決率等が閾値以上であれば小さな評価値を与え、Ｉｓｓｕｅ解決率等が閾値よりも小さければ大きな評価値を与える。あるいは、リスク評価値生成部２１２は、Ｉｓｓｕｅ解決率等が属する数値範囲に応じて評価値を決定してもよい。上記評価値の決定は、Ｉｓｓｕｅ解決率やコミット数が所定の値よりも大きければ対応するライブラリの開発は継続されておりリスクを低く見積もることが妥当であるという知見に基づく。
【００４８】
リスク評価値生成部２１２は、コード生成者が予め定められた人物に該当するか否かに応じて評価値を決定してもよい。例えば、リスク評価値生成部２１２は、脆弱性の少ない安全なライブラリを多数生成した人物のリスト（予め用意されたホワイトリスト）を参照する。リスク評価値生成部２１２は、当該リストに評価対象ライブラリの生成者が含まれる場合には、対応するライブラリに小さい評価値を与える。
【００４９】
リスク評価値生成部２１２は、脆弱性が多数報告され、不完全なライブラリを多数生成した人物のリスト（予め用意されたブラックリスト）を参照してもよい。当該リストに評価対象ライブラリの生成者が含まれる場合、リスク評価値生成部２１２は対応するライブラリに大きな評価値を与えてもよい。
【００５０】
リスク評価値生成部２１２は、上記項目（例えば、Ｉｓｓｕｅ解決率、コミット数、コード作成者）の全部又は一部を用いて評価値を算出する。
【００５１】
なお、以降の説明において、Ｉｓｓｕｅ解決率等の動的な情報（メタデータ）に基づき定められた評価値を「動的評価値」と表記する。上記Ｉｓｓｕｅ解決率等はメタデータの一例であって、上記以外の情報、例えば、ＣＶＥ（Common Vulnerabilities and Exposures）、コミットサイズ、更新頻度等を用いて動的評価値が算出されてもよい。あるいは、使用ライブラリのバージョンと最新バージョン（最新のコミット）の距離（乖離度）を用いて動的評価値が算出されてもよい。例えば、最新バージョンと使用ライブラリのバージョンの差分が計算され、当該差分が含まれる数値範囲に基づいて動的評価値が算出されてもよい。また、メジャーバージョンとマイナーバージョンで重みを変え、メジャーバージョンが大きく離れている場合に動的評価値が大きく算出（リスクが高く算出）されてもよい。
【００５２】
ここで、使用ライブラリがオープンソースソフトウェアとして公開されているライブラリであれば、上記動的評価値の算出する際の基礎となるメタデータはネットワーク上で公開されていることが多い。リスク評価値生成部２１２は、当該ネットワーク上で公開されているメタデータを取得し、上記動的評価値を算出すればよい。上記公開されているメタデータの一例として下記の参考情報１が例示される。
＜参考情報１＞
ＵＲＬ；https://developer.github.com/
【００５３】
リスク評価値生成部２１２は、評価対象ライブラリのソースコードを用いてライブラリ

を評価してもよい。例えば、リスク評価値生成部２１２は、評価対象ライブラリのソースコードに記載されたコメントの数、コードの複雑度、ソースコードに記載された関数の名称等に基づき評価値を算出する。

【００５４】

例えば、リスク評価値生成部２１２は、上記コメント数やコードの複雑度に対して、閾値処理を実行し、その結果に応じて評価値を決定してもよい。より具体的には、リスク評価値生成部２１２は、コメント数等が閾値以上であれば小さい評価値を与え、コメント数等が閾値よりも小さければ大きな評価値を与える。

【００５５】

リスク評価値生成部２１２は、評価対象ライブラリのソースコードに記載された関数の名称が所定の規則に合致しているか否か（あるいは、所定の規則からどの程度乖離しているか否か）に応じて評価値を決定してもよい。

【００５６】

リスク評価値生成部２１２は、上記項目（例えば、コメント数、コード複雑度、関数の名称）の全部又は一部を用いて評価値を算出する。

【００５７】

なお、以降の説明において、コード複雑度等の評価対象ライブラリのソースコードに記載された情報に基づき定められた評価値を「実体評価値」と表記する。

【００５８】

リスク評価値生成部２１２は、下記の参考情報２に記載された情報に基づき上記実体評価値を算出してもよい。参考情報２には、ソースコードのライン数、コードの循環的複雑度のメトリクスが記載されている。

＜参考情報２＞

ＵＲＬ；https://www.techmatrix.co.jp/product/understand/function/metrics.html

【００５９】

上記説明したように、リスク評価値生成部２１２は、種々の観点から評価対象ライブラリに関する単体評価を行う。即ち、リスク評価値生成部２１２は、評価対象ライブラリのディレクトリ構成等の静的情報、メタデータ等の動的情報、ソースコードによる実体情報等を用いた多種多様な観点からライブラリ単体評価を実行する。

【００６０】

上記３つの単体評価は例示であって、リスク評価値生成部２１２のライブラリ単体評価の実施内容を限定する趣旨でない。リスク評価値生成部２１２は、上記３つの単体評価のうち少なくとも１つ以上の評価を実施すればよく、上記３つの単体評価とは異なる評価をさらに実施してもよい。

【００６１】

リスク評価値生成部２１２は、上記説明したライブラリ単体評価を抽出されたライブラリについて実行する。例えば、リスク評価値生成部２１２は、図６に示すような情報を生成し、記憶部２０４に格納する。

【００６２】

ライブラリの中で他のライブラリが呼び出されている場合には、リスク評価値生成部２１２は、当該他のライブラリに関するライブラリ単体評価の結果を加味してライブラリを評価する。つまり、ライブラリに子ライブラリが含まれる場合には、リスク評価値生成部２１２は、子ライブラリに関する単体評価の結果を加味して親ライブラリを評価する（親ライブラリを評価するための情報を生成する）。

【００６３】

図５の例では、ライブラリ「Ｌ１２３」が親ライブラリ、「Ｌ９１１」、「Ｌ９１２」が子ライブラリの関係にあり、子ライブラリ「Ｌ９１１」、「Ｌ９１２」の単体評価の結果が親ライブラリ「Ｌ１２３」の評価に反映される。

【００６４】

リスク評価値生成部２１２は、評価対象ライブラリに子ライブラリが含まれる場合には、当該子ライブラリに関する単体評価の評価値のうち値が最も大きい（リスクが最も大きい）評価値を特定する。例えば、図５の例では、ライブラリ「Ｌ１２３」には２つの子ライブラリ「Ｌ９１１」と「Ｌ９１２」が含まれるので、これら２つの子ライブラリに関する単体評価による評価値のうち値が最も大きい評価値が特定される。

【００６５】

図６の例では、評価値Ａ１３、Ｂ１３、Ｃ１３、Ａ１４、Ｂ１４、Ｃ１４のうち評価値Ａ１３の値が最大であれば、当該評価値Ａ１３が特定される。リスク評価値生成部２１２は、当該特定された評価値に対応するライブラリ（子ライブラリ）を評価対象ライブラリ（親ライブラリ）のリスク要因として取り扱う。上記の例では、評価値Ａ１３に対応するライブラリ「Ｌ９１１」がリスク要因として扱われる。

【００６６】

このように、リスク評価値生成部２１２は、ライブラリ間に依存関係が認められれば、従属ライブラリを親ライブラリのリスク要因として抽出する。ライブラリ単体評価及び依存関係に基づくリスク要因の特定が終了すると、リスク評価値生成部２１２は、上記単体評価及びリスク要因をまとめ「ライブラリ評価結果」を生成する。例えば、リスク評価値生成部２１２は、図７に示すようなライブラリ評価結果を生成する。

【００６７】

図３に説明を戻す。ライブラリ影響度算出部２１３は、ソースコードに含まれる各ライブラリがソースコード全体（プロジェクト全体）に与える影響を示す影響度を算出する手段である。

【００６８】

例えば、ライブラリ影響度算出部２１３は、ソースコードに含まれるライブラリごとに、ライブラリを呼び出している関数の数を計数する。図４の例では、ライブラリ「Ｌ１２３」に関しては、ＦｕｎｃｔｉｏｎＡで２回（７、８行目）、ＦｕｎｃｔｉｏｎＢで１回（１０行目）呼び出されているので、ライブラリ「Ｌ１２３」を呼び出す関数の数は「２」となる。ライブラリ「Ｌ３４５」に関しては、ＦｕｎｃｔｉｏｎＢで１回（１１行目）呼び出されているので、ライブラリ「Ｌ１２３」を呼び出す関数の数は「１」となる。

【００６９】

ライブラリ影響度算出部２１３は、各ライブラリを呼び出している関数の数を「ライブラリ影響度」として算出する。

【００７０】

ライブラリ影響度算出部２１３は、ソースコードに含まれる関数の依存関係を考慮してライブラリ影響度を算出してもよい。例えば、ライブラリＡを呼び出す関数Ａが存在し、当該関数Ａをさらに呼び出す関数Ｂ～Ｆがソースコードに記載されている場合を考える。この場合、ライブラリＡを直接呼び出す関数の数は「１」であるが、５つの関数Ｂ～ＦもライブラリＡを間接的に呼び出すので、ライブラリＡの影響度が「６＝（１＋５）」と算出されてもよい。即ち、ライブラリ影響度算出部２１３は、ライブラリを直接呼び出している関数の数と当該ライブラリを間接的に呼び出している関数の数に基づきライブラリ影響度を算出してもよい。

【００７１】

なお、ソースコードによっては関数が再帰的に呼び出されることがある。このように再帰的に呼び出される関数で使用されるライブラリに関しては、ライブラリ影響度算出部２１３は、当該ライブラリの影響度に極めて大きな値を与えてもよい。

【００７２】

あるいは、ライブラリ影響度算出部２１３は、各ライブラリが呼び出されている数を「ライブラリ影響度」として算出してもよい。例えば、上記の例では、ライブラリ「Ｌ１２３」は全体として３回呼び出されているのでライブラリ影響度が「３」、ライブラリ「Ｌ３４５」は全体として１回呼び出されているのでライブラリ影響度が「１」となる。

【００７３】

ライブラリ影響度算出部２１３は、各ライブラリを呼び出す関数の数、及び、各ライブラリが呼び出されている数のいずれかを使用してライブラリ影響度を算出してもよいし、これらの情報を用いてライブラリ影響度を算出してもよい。例えば、ライブラリ影響度算出部２１３は、上記関数の数と呼び出し回数の平均（例えば、相加平均、相乗平均、加重平均）を計算し、当該計算された平均値をライブラリ影響度として算出してもよい。

【００７４】

ライブラリ影響度算出部２１３は、ソースコードに含まれる各ライブラリについてライブラリ影響度を算出する（図８参照）。

【００７５】

ライブラリ情報生成部２０２は、リスク評価値生成部２１２が生成したライブラリ評価結果とライブラリ影響度算出部２１３が算出したライブラリ影響度を結合し、「ライブラリ評価情報」を生成する（図９参照）。

【００７６】

リスク情報出力部２０３は、ライブラリの評価値に基づき、当該ライブラリの危険度を算出し、算出された危険度に基づきソースコードに内在するリスクを示すリスク値を算出する手段である。さらに、リスク情報出力部２０３は、当該算出されたリスク値の時系列データを出力する。即ち、リスク情報出力部２０３は、ライブラリ情報生成部２０２が生成したライブラリ評価情報に基づき、評価対象となるソースコードに関するリスク情報を生成し、出力する。

【００７７】

具体的には、リスク情報出力部２０３は、管理者等から指定された項目のリスク情報を生成する。リスク情報出力部２０３は、上記生成するリスク情報を入力するためのＧＵＩを生成し、液晶モニタ等に表示する。

【００７８】

例えば、生成するリスク情報として、ソースコード全体（プロジェクト全体）のリスクが例示される。この場合、リスク情報出力部２０３は、評価対象ソースコードに含まれる各ライブラリについて危険度を算出し、当該算出された危険度の合計値をソースコード全体のリスク値とする。

【００７９】

例えば、リスク情報出力部２０３は、各ライブラリについての単体評価値（静的評価値、動的評価値、実体評価値）の合計を各ライブラリの危険度として算出する。図９の例では、ライブラリ「Ｌ１２３」について「Ａ１１＋Ｂ１１＋Ｃ１１」が当該ライブラリの危険度として算出される。同様に、ライブラリ「Ｌ３４５」について「Ａ１２＋Ｂ１２＋Ｃ１２」が当該ライブラリの危険度として算出される。このように、リスク情報出力部２０３は、異なる情報ごとに生成された評価値（静的評価値、動的評価値、実体評価値）の合計値を各ライブラリの危険度として算出する。

【００８０】

リスク情報出力部２０３は、各ライブラリの危険度を合算し、アプリケーションプログラム全体のリスク値として算出する。上記の例では、「Ａ１１＋Ｂ１１＋Ｃ１１＋Ａ１２＋Ｂ１２＋Ｃ１２＋Ａ１３・・・」がソースコード全体のリスク値として算出される。

【００８１】

リスク情報出力部２０３は、各ライブラリの危険度を算出する際、リスク要因やライブラリ影響度を反映してもよい。例えば、リスク要因を反映する場合には、リスク情報出力部２０３は、上記単体評価値の合計にリスク要因の評価値を加算する。上記の例では、ライブラリ「Ｌ１２３」について「Ａ１１＋Ｂ１１＋Ｃ１１＋Ａ１３」が当該ライブラリの危険度として算出される。

【００８２】

ライブラリ影響度を反映する場合には、リスク情報出力部２０３は、上記単体評価値の合計値に対応するライブラリ影響度を乗算する。例えば、ライブラリ「Ｌ１２３」について「（Ａ１１＋Ｂ１１＋Ｃ１１）×Ｅ１１」が当該ライブラリの危険度として算出される

。あるいは、リスク要因も反映され、ライブラリ「Ｌ１２３」について「（Ａ１１＋Ｂ１１＋Ｃ１１＋Ａ１３）×Ｅ１１」が当該ライブラリの危険度として算出されてもよい。

【００８３】

リスク情報出力部２０３は、生成したリスク情報（リスク値）をその生成時刻と共に記憶部２０４に格納する。例えば、図１０に示すようなリスク値の時系列データ（リスク値の履歴）が記憶部２０４に格納される。

【００８４】

リスク情報出力部２０３は、記憶部２０４に格納された最新のリスク値（現在のリスク値）を表示してもよい。あるいは、リスク情報出力部２０３は、記憶部２０４に格納された時系列データからリスク値の時間推移を示すグラフ（図１０に示すようなグラフ）生成し、当該生成されたグラフを表示してもよい。

【００８５】

例えば、リスク情報出力部２０３は、図１１に示すような表示を行う。図１１では、ソースコード全体のリスク値に関する時系列データが表示されている。図１１に示すように、リスク情報出力部２０３は、ソースコードに含まれるライブラリについて算出された危険度の合計値をソースコード全体のリスク値として算出し、当該算出されたソースコード全体のリスク値の時系列データを表示してもよい。

【００８６】

図１１に示すようなリスク情報に接した管理者は、プロジェクトの運用開始時に比べ、現状のリスクが上昇していることを認識する。なお、図１１では、リスク値が折れ線グラフの形式で表示されているが、棒グラフ等の他の形状によりリスク値が表示されてもよい。

【００８７】

リスク情報出力部２０３は、個別のライブラリに関するリスク情報を生成してもよい。例えば、管理者は、リスクを評価したいライブラリを、ＧＵＩを用いて指定する。リスク情報出力部２０３は、指定されたライブラリの単体評価値を合計し危険度を計算する。

【００８８】

リスク情報出力部２０３は、当該計算された危険度をリスク情報（リスク値）として表示する。例えば、リスク情報出力部２０３は、図１２に示すような表示をしてもよい。図１１及び図１２に示すように、リスク情報出力部２０３は、リスク値の時系列データを表示する項目（プロジェクト全体、個別ライブラリ）を切り替えるＧＵＩを生成してもよい。

【００８９】

リスク情報出力部２０３は、各ライブラリの単体評価に関する個別データ（静的評価値、動的評価値、実体評価値）を表示してもよい。例えば、リスク情報出力部２０３は、図１３に示すような表示をしてもよい。

【００９０】

図１３Ａでは、ライブラリ「Ｌ１２３」の単体評価により得られる３つの評価値（各評価値の時系列データ）が同時に表示されている。図１３Ｂでは、ライブラリ「Ｌ１２３」の単体評価を構成する３つの評価値のうち動的評価値に関する時系列データが表示されている。あるいは、単体評価により得られる３つの評価値のうち２つが選択され、表示されてもよい。

【００９１】

リスク情報出力部２０３は、各ライブラリのリスク値の一覧を表示してもよい。例えば、リスク情報出力部２０３は、図１４Ａに示すように、ライブラリと当該ライブラリのリスク値の一覧を表示してもよい。

【００９２】

リスク情報出力部２０３は、図１４Ｂに示すように、リスク値に対して閾値処理を実行し、その結果の一覧を表示してもよい。

【００９３】

リスク情報出力部２０３は、リスク値の内訳を表示してもよいし、当該内訳を時系列データの付随情報として表示してもよい。例えば、リスク情報出力部２０３は、図１５に示

すように、指定された時刻における単体評価の内訳を表示してもよい。

【００９４】

リスク情報出力部２０３は、ライブラリ構成情報（図５参照）に基づき、ライブラリ間の依存関係（従属関係）を表示してもよい。例えば、リスク情報出力部２０３は、図１６に示すような表示をしてもよい。

【００９５】

リスク情報出力部２０３は、単体評価の基礎となった情報（例えば、メタデータ）を表示してもよい。例えば、リスク情報出力部２０３は、図１７に示すような情報を表示してもよい。

【００９６】

第１の実施形態に係る評価装置１０の動作をまとめると図１８に示すフローチャートのとおりとなる。

【００９７】

評価装置１０は、評価対象となるソースコードを取得する（ステップＳ１０１）。

【００９８】

評価装置１０は、ソースコードに記載されたライブラリを抽出する（ステップＳ１０２）。

【００９９】

評価装置１０は、当該抽出されたライブラリの単体評価を実施する（ステップＳ１０３）。

【０１００】

評価装置１０は、上記抽出されたライブラリに関する影響度（ライブラリ影響度）を算出する（ステップＳ１０４）。

【０１０１】

評価装置１０は、ライブラリ評価情報に基づきリスク情報を生成し、出力する（ステップＳ１０５）。

【０１０２】

なお、評価装置１０を用いたソースコードのリスク評価は、定期的又は所定のタイミングで実施され、その評価結果が履歴情報として記憶部２０４に格納される。例えば、あるプロジェクトのソースコードがリリースされるタイミング等に上記リスク評価が実行される。

【０１０３】

以上のように、第１の実施形態に係る評価装置１０は、ソースコードに記載された各ライブラリの単体評価を実施し、当該評価に基づきソースコードのリスク変化を管理者等に提示する。当該提示されたリスク変化に接した管理者は、プロジェクト（アプリケーション）のリスクが許容できるのか否か等を判断することができる。また、評価装置１０は、ソースコードに記載された各ライブラリについてそのリスク変化を算出し、管理者等に提供できる。その結果、管理者等は、いずれのライブラリに関するリスクが大きく変化した等の情報を正確に得ることができる。

【０１０４】

また、評価装置１０は、ソースコードを取得するたびに、ライブラリの評価を行うので、ライブラリのバージョン違い等により生じる潜在的なリスクを明らかにすることができる。例えば、運用開始時に使用されているライブラリのバージョンが進んでいるにも関わらず、評価対象のソースコードに記載されたライブラリのバージョンが運用開始時と同じであると言った状況が起こりうる。この場合、評価装置１０によるライブラリ単体評価のうち、動的評価（メタデータを用いた評価）の結果が悪化し、リスク値が上昇する。評価装置１０は、このようなライブラリのバージョンアップが放置されたような状況を検出し、当該放置により生じるリスクを管理者等に通知する。

【０１０５】

また、評価装置１０は、ソースコードを取得するたびに、当該ソースコードを走査し、

当該ソースコードに含まれるライブラリの影響度を算出するので、ライブラリを使用する側の変化により生じるリスクも明らかにすることができる。例えば、運用開始時にライブラリＡが１つの関数により呼び出され、評価時には当該ライブラリＡが３つの関数により呼び出されるような改訂が行われることがある。この場合、ライブラリＡ自身のリスク値は変化していなくとも、ライブラリＡがソースコード（プロジェクト）全体に与える影響度は増加（上記の例では３倍）している。評価装置１０は、ライブラリがソースコード全体に与える影響も考慮しつつ、リスク値を算出するので上記ライブラリを使用する側の変化により生じるリスク値を算出できる。

【０１０６】

［第２の実施形態］

続いて、第２の実施形態について図面を参照して詳細に説明する。

【０１０７】

第１の実施形態では、評価装置１０がリスク情報を管理者等に提供し、当該リスク情報に接した管理者が、ソースコード（ソースコード全体、各ライブラリ）のリスクを判断している。第２の実施形態では、評価装置１０が、ソースコードのリスクを自動的に判断し、その判断結果を管理者等に提示する場合について説明する。

【０１０８】

図１９は、第２の実施形態に係る評価装置１０の処理構成（処理モジュール）の一例を示す図である。図３及び図１９を参照すると、第１の実施形態に係る評価装置１０に対してリスク通知部２０５が追加されている。

【０１０９】

以下、第１及び第２の実施形態の相違点を中心に説明する。

【０１１０】

リスク通知部２０５は、算出されたリスク値が所定の条件を満たす場合に、ソースコードにリスクが含まれる旨を外部に通知する手段である。具体的には、リスク通知部２０５は、管理者に通知する必要のあるリスクがソースコードに認められればその旨を通知する。

【０１１１】

例えば、リスク通知部２０５は、リスク情報出力部２０３が生成したリスク値（アプリケーションプログラムのリスク値、個別のライブラリに関するリスク値）に対して閾値処理を実行する。リスク通知部２０５は、リスク値が所定の閾値以上であれば、「ソースコードにリスクあり」を管理者等に通知する。具体的には、リスク通知部２０５は、図２０に示すような表示を行う。

【０１１２】

リスク通知部２０５は、特定のライブラリについてのリスク値が閾値よりも高い場合には、その旨を管理者等に通知してもよい。例えば、リスク通知部２０５は、図２１に示すような表示を行う。

【０１１３】

リスク通知部２０５は、運用開始時と現時点におけるリスク上昇率を計算し、当該計算されたリスク上昇率に応じた通知をしてもよい。当該リスク上昇率の計算は、アプリケーションプログラム全体に関してでもよいし、個別のライブラリに関してでもよい。

【０１１４】

リスク通知部２０５は、ライブラリの単体評価を構成する評価値（静的評価値、動的評価値、実体評価値）それぞれについてリスク上昇率を計算してもよい。例えば、リスク通知部２０５は、特定のライブラリに関する動的評価値が運用開始時と比較して極端に上昇していればその旨を管理者等に通知する。この場合、リスク通知部２０５は、図２２に示すような表示を行ってもよい。図２２には、ライブラリ「Ｌ１２３」の動的評価値が上昇している旨と、運用開始時及び現在時刻を含む動的評価値の時系列データが表示されている。

【０１１５】

以上のように、第２の実施形態に係る評価装置１０は、ソースコードが持つリスクの変

動を検出し、当該リスクの変動を管理者等に通知する必要がある場合には、その旨を管理者等に通知する。その結果、管理者等は、ソースコードに記載されたライブラリのリスク変化を網羅的に確認する必要がなく、重大なリスクを見落とすことが防止できる。

【０１１６】

続いて、評価装置１０のハードウェアについて説明する。図２３は、評価装置１０のハードウェア構成の一例を示す図である。

【０１１７】

評価装置１０は、情報処理装置（所謂、コンピュータ）により構成可能であり、図２３に例示する構成を備える。例えば、評価装置１０は、プロセッサ３１１、メモリ３１２、入出力インターフェイス３１３及び通信インターフェイス３１４等を備える。上記プロセッサ３１１等の構成要素は内部バス等により接続され、相互に通信可能に構成されている。

【０１１８】

但し、図２３に示す構成は、評価装置１０のハードウェア構成を限定する趣旨ではない。評価装置１０は、図示しないハードウェアを含んでもよいし、必要に応じて入出力インターフェイス３１３を備えていなくともよい。また、評価装置１０に含まれるプロセッサ３１１等の数も図２３の例示に限定する趣旨ではなく、例えば、複数のプロセッサ３１１が評価装置１０に含まれていてもよい。

【０１１９】

プロセッサ３１１は、例えば、ＣＰＵ（Central Processing Unit）、ＭＰＵ（Micro Processing Unit）、ＤＳＰ（Digital Signal Processor）等のプログラマブルなデバイスである。あるいは、プロセッサ３１１は、ＦＰＧＡ（Field Programmable Gate Array）、ＡＳＩＣ（Application Specific Integrated Circuit）等のデバイスであってもよい。プロセッサ３１１は、オペレーティングシステム（ＯＳ；Operating System）を含む各種プログラムを実行する。

【０１２０】

メモリ３１２は、ＲＡＭ（Random Access Memory）、ＲＯＭ（Read Only Memory）、ＨＤＤ（Hard Disk Drive）、ＳＳＤ（Solid State Drive）等である。メモリ３１２は、ＯＳプログラム、アプリケーションプログラム、各種データを格納する。

【０１２１】

入出力インターフェイス３１３は、図示しない表示装置や入力装置のインターフェイスである。表示装置は、例えば、液晶ディスプレイ等である。入力装置は、例えば、キーボードやマウス等のユーザ操作を受け付ける装置である。

【０１２２】

通信インターフェイス３１４は、他の装置と通信を行う回路、モジュール等である。例えば、通信インターフェイス３１４は、ＮＩＣ（Network Interface Card）等を備える。

【０１２３】

評価装置１０の機能は、各種処理モジュールにより実現される。当該処理モジュールは、例えば、メモリ３１２に格納されたプログラムをプロセッサ３１１が実行することで実現される。また、当該プログラムは、コンピュータが読み取り可能な記憶媒体に記録することができる。記憶媒体は、半導体メモリ、ハードディスク、磁気記録媒体、光記録媒体等の非トランジェント（non-transitory）なものとすることができる。即ち、本発明は、コンピュータプログラム製品として具現することも可能である。また、上記プログラムは、ネットワークを介してダウンロードするか、あるいは、プログラムを記憶した記憶媒体を用いて、更新することができる。さらに、上記処理モジュールは、半導体チップにより実現されてもよい。

【０１２４】

［変形例］

なお、上記実施形態にて説明した評価装置１０の構成、動作等は例示であって、当該装置の構成等を限定する趣旨ではない。例えば、上記説明した評価装置１０の機能が異なる装置により実現されてもよい。具体的には、ライブラリ情報生成部２０２とリスク情報出

力部２０３の機能が異なる装置に実装されていてもよい。
【０１２５】
上記実施形態では、リスク評価値生成部２１２がソースコードを走査し、その記載内容を解析する「静的コード解析」により単体評価を行うことを説明した。しかし、評価対象のソースコードから生成された実行ファイルを実行し、評価対象のソースコードを解析する「動的プログラム解析」が実行されてもよい。例えば、各ライブラリが呼び出されるごとにフラグを立てるようなテストプログラムを用意し、単位時間あたりに呼び出されるライブラリの回数を評価値としてもよい。
【０１２６】
上記実施形態では、ソースコードに記載されたライブラリの観点（視点）から上記ソースコードのリスクを評価しているが、上記ソースコード自身の評価結果を加味して当該ソースコードのリスクを評価してもよい。例えば、ライブラリのコード解析（ライン数、複雑度等を用いた解析）と同様の解析をソースコードに対して実行し、その結果がソースコード全体のリスク値に反映されてもよい。
【０１２７】
上記実施形態では、ライブラリの危険度を算出する際、ライブラリの単体評価（静的評価値、動的評価値、実体評価値）の加算を行うことを説明した。しかし、当該危険度を算出する際、各項目に予め与えられた重みを用いた加重平均により危険度が算出されてもよい。例えば、上記３つの評価値のうち、動的評価値を重要視するような重みを決定し、ライブラリの危険度が算出されてもよい。
【０１２８】
上記３つの評価値（静的評価値、動的評価値、実体評価値）を算出する際、全部又は一部に関して機械学習の手法が適用されてもよい。例えば、ソースコードを用いた実体評価値の算出では、脆弱性が少ない優良なライブラリのソースコードを多数収集し、当該収集したソースコードに高い評価を与えるラベルを付して教師データとして用意する。当該教師データを用いて学習モデルが生成され、当該学習モデルに評価対象のライブラリのソースコードを入力することで、教師データとの差分がスコアとして出力される。当該学習モデルにより出力されたスコアが実体評価値として扱われてもよい。
【０１２９】
上記実施形態では、親ライブラリの評価に子ライブラリの評価値をリスク要因として含ませる場合を説明した。親ライブラリの評価に子ライブラリ、孫ライブラリの評価値を含ませてもよい。この場合、リスク評価値生成部２１２は、図２４に示すようなライブラリ評価結果を生成する。図２４では、親ライブラリ「Ｌ１２３」の子ライブラリ「Ｌ９１１」が第１のリスク要因として抽出され、孫ライブラリ「Ｌ８１１」が第２のリスク要因として抽出されている。なお、当該リスク要因の抽出は上記説明した内容と同一とすることができるので説明を省略する。
【０１３０】
上記説明した評価装置１０は、複数のプロジェクトに関するソースコードを取り扱う。そのため、プロジェクトは異なるが使用されているライブラリは共通しているという状況が起こりうる。その場合、評価装置１０は、既に単体評価が終了しているライブラリと同じライブラリが他のプロジェクト（ソースコード）で使用されている場合には、当該ライブラリの単体評価の結果を再利用してもよい。
【０１３１】
コンピュータの記憶部に評価プログラムをインストールすることにより、コンピュータを評価装置として機能させることができる。また、評価プログラムをコンピュータに実行させることにより、コンピュータにより評価方法を実行することができる。
【０１３２】
また、上述の説明で用いたフローチャートでは、複数の工程（処理）が順番に記載されているが、各実施形態で実行される工程の実行順序は、その記載の順番に制限されない。各実施形態では、例えば各処理を並行して実行する等、図示される工程の順番を内容的に

支障のない範囲で変更することができる。また、上述の各実施形態は、内容が相反しない範囲で組み合わせることができる。

【０１３３】

上記の実施形態の一部又は全部は、以下の付記のようにも記載され得るが、以下には限られない。

［付記１］

ソースコードに記載された、第１のライブラリのリスクに関する評価値を生成する、生成部（１０１、２１２）と、

少なくとも前記生成された評価値に基づき、前記第１のライブラリの危険度を算出し、前記算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、前記算出されたリスク値の時系列データを出力する、出力部（１０２、２０３）と、

を備える、評価装置（１０、１００）。

［付記２］

前記生成部（１０１、２１２）は、前記ソースコードに前記第１のライブラリから呼び出される第２のライブラリが含まれる場合には、前記第２のライブラリのリスクに関する評価値を生成し、

前記出力部（１０２、２０３）は、前記第１のライブラリの危険度に前記第２のライブラリの評価値を反映する、付記１に記載の評価装置（１０、１００）。

［付記３］

前記第１のライブラリが前記ソースコードの全体に与える影響を示す影響度を算出する、算出部（２１３）をさらに備え、

前記出力部（１０２、２０３）は、前記算出された影響度を前記第１のライブラリの危険度に反映する、付記２に記載の評価装置（１０、１００）。

［付記４］

前記生成部（１０１、２１２）は、前記第１のライブラリを互いに性質の異なる情報に基づき評価し、前記異なる情報ごとに評価値を生成する、付記３に記載の評価装置（１０、１００）。

［付記５］

前記出力部（１０２、２０３）は、前記異なる情報ごとに生成された評価値の合計値を前記第１のライブラリの危険度として算出する、付記４に記載の評価装置（１０、１００）。

［付記６］

前記出力部（１０２、２０３）は、前記第２のライブラリに関して生成された前記異なる情報ごとの評価値のうち値の最も大きい評価値を特定し、前記第１のライブラリに関して生成された評価値の合計値に前記特定された評価値を加算することで、前記第１のライブラリの危険度を算出する、付記５に記載の評価装置（１０、１００）。

［付記７］

前記出力部（１０２、２０３）は、前記評価値の合計値に前記影響度を乗算することで、前記第１のライブラリの危険度を算出する、付記５又は６に記載の評価装置（１０、１００）。

［付記８］

前記生成部（１０１、２１２）は、前記第１のライブラリに関する静的な情報に基づき静的評価値を生成し、前記第１のライブラリに関する動的な情報に基づき動的評価値を生成し、前記第１のライブラリのソースコードに基づき実体評価値を生成する、付記４乃至７のいずれか一つに記載の評価装置（１０、１００）。

［付記９］

前記出力部（１０２、２０３）は、前記ソースコードに含まれるライブラリについて算出された危険度の合計値を前記ソースコード全体のリスク値として算出し、前記算出されたソースコード全体のリスク値の時系列データを表示する、付記１乃至７のいずれか一つ

に記載の評価装置（１０、１００）。
［付記１０］
前記出力部（１０２、２０３）は、前記第１のライブラリの危険度の時系列データを表示する、付記９に記載の評価装置（１０、１００）。
［付記１１］
前記出力部（１０２、２０３）は、リスク値の時系列データを表示する項目を切り替えるためのＧＵＩ（Graphical User Interface）を生成する、付記１０に記載の評価装置（１０、１００）。
［付記１２］
前記算出されたリスク値が所定の条件を満たす場合に、前記ソースコードにリスクが含まれる旨を外部に通知する、通知部をさらに備える、付記１乃至１１のいずれか一つに記載の評価装置（１０、１００）。
［付記１３］
評価装置（１０、１００）において、
ソースコードに記載された、第１のライブラリのリスクに関する評価値を生成するステップと、
少なくとも前記生成された評価値に基づき、前記第１のライブラリの危険度を算出し、前記算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、前記算出されたリスク値の時系列データを出力するステップと、
を含む評価方法。
［付記１４］
評価装置（１０、１００）に搭載されたコンピュータ（３１１）に、
ソースコードに記載された、第１のライブラリのリスクに関する評価値を生成する処理と、
少なくとも前記生成された評価値に基づき、前記第１のライブラリの危険度を算出し、前記算出された危険度に基づき前記ソースコードに内在するリスクを示すリスク値を算出すると共に、前記算出されたリスク値の時系列データを出力する処理と、
を実行させるプログラム。
なお、付記１３の形態及び付記１４の形態は、付記１の形態と同様に、付記２の形態～付記１２の形態に展開することが可能である。
【０１３４】
なお、引用した上記の先行技術文献の各開示は、本書に引用をもって繰り込むものとする。以上、本発明の実施形態を説明したが、本発明はこれらの実施形態に限定されるものではない。これらの実施形態は例示にすぎないということ、及び、本発明のスコープ及び精神から逸脱することなく様々な変形が可能であるということは、当業者に理解されるであろう。
【符号の説明】
【０１３５】
１０、１００　評価装置
１０１　生成部
１０２　出力部
２０１　ソースコード取得部
２０２　ライブラリ情報生成部
２０３　リスク情報出力部
２０４　記憶部
２０５　リスク通知部
２１１　ライブラリ抽出部
２１２　リスク評価値生成部
２１３　ライブラリ影響度算出部
３１１　プロセッサ

３１２　メモリ
３１３　入出力インターフェイス
３１４　通信インターフェイス
【図面】
【図１】

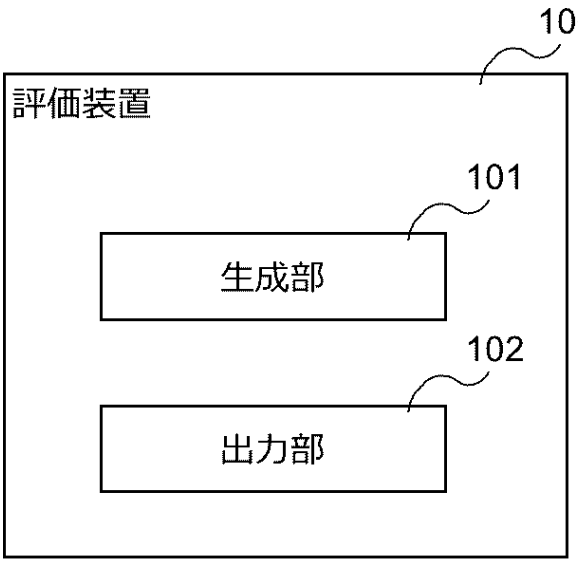

Fig.1

【図２】

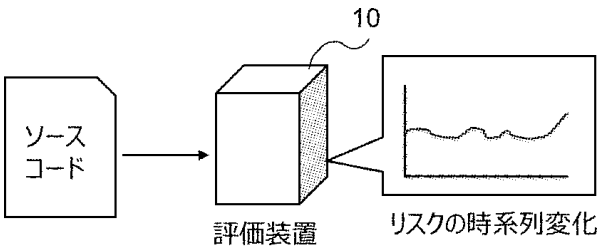

Fig.2

【図３】

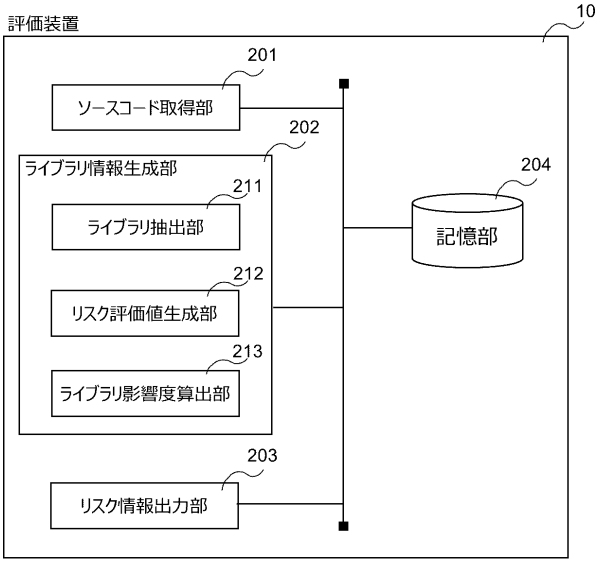

Fig.3

【図４】

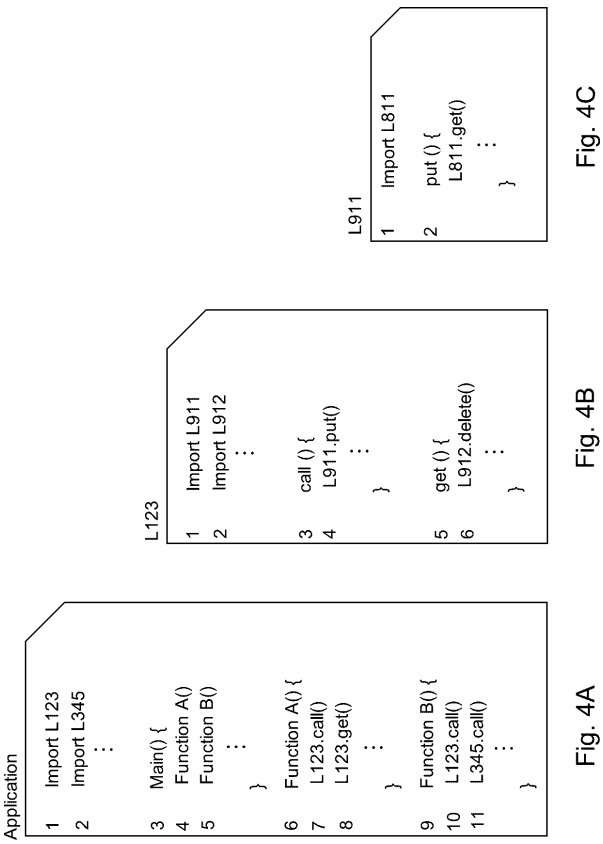

Fig. 4C

Fig. 4B

Fig. 4A

【図５】

ライブラリ構成情報

ライブラリ	従属ライブラリ
L123	L911,L912
L345	None
L911	L811
L912	None
⋮	⋮

Fig. 5

【図６】

ライブラリ	静的評価値	動的評価値	実体評価値
L123	A11	B11	C11
L345	A12	B12	C12
L911	A13	B13	C13
L912	A14	B14	C14
⋮	⋮	⋮	⋮

Fig. 6

【図７】

ライブラリ評価結果

ライブラリ	静的評価値	動的評価値	実体評価値	リスク要因
L123	A11	B11	C11	L911(A13)
L345	A12	B12	C12	None
L911	A13	B13	C13	None
L912	A14	B14	C14	None
⋮	⋮	⋮	⋮	⋮

Fig. 7

【図８】

ライブラリ	ライブラリ影響度
L123	E11
L345	E12
L911	E13
L912	E14
⋮	⋮

Fig. 8

【図９】

ライブラリ評価情報

ライブラリ	静的評価	動的評価	実体評価	リスク要因	ライブラリ影響度
L123	A11	B11	C11	L911(A13)	E11
L345	A12	B12	C12	None	E12
L911	A13	B13	C13	None	E13
L912	A14	B14	C14	None	E14
…	…	…	…	…	…

Fig. 9

【図１０】

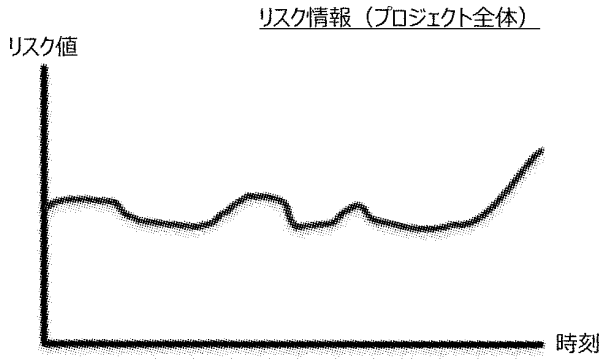

Fig. 10

【図１１】

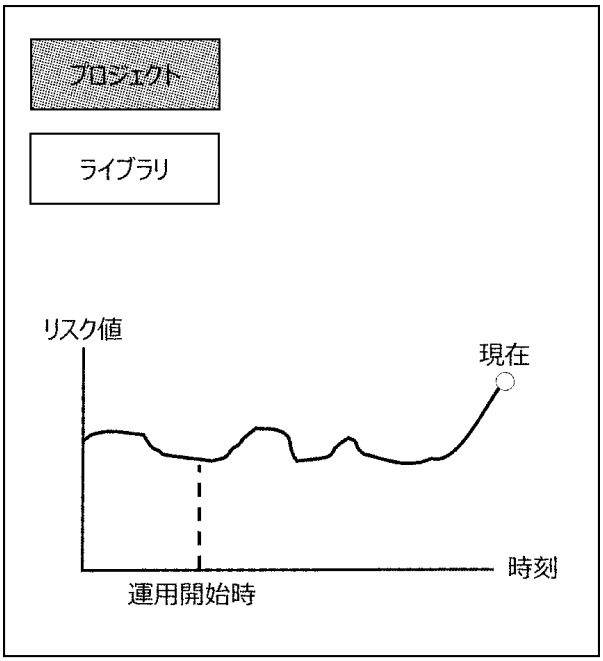

Fig. 11

【図１２】

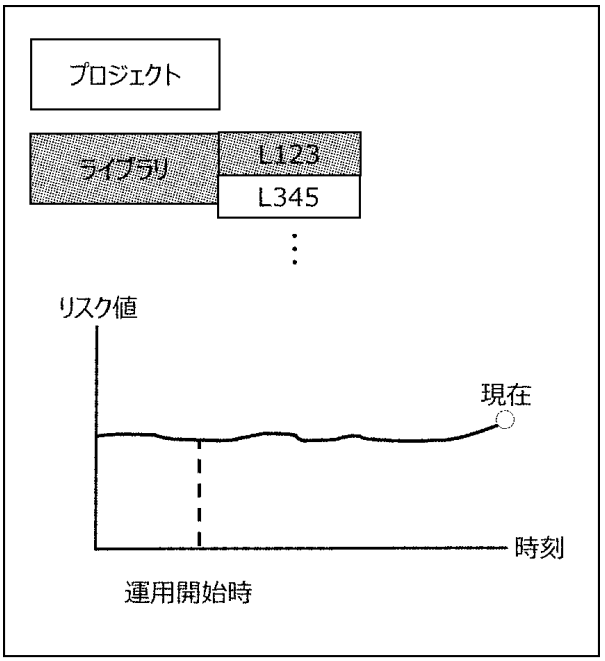

Fig. 12

【図１３】

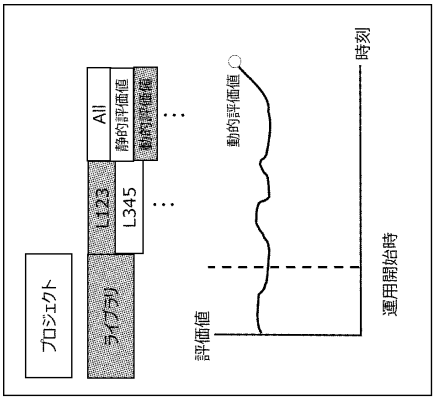

Fig. 13B

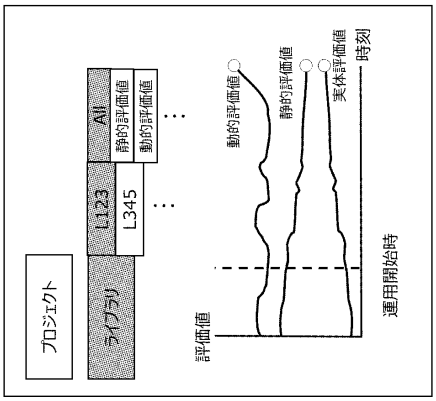

Fig. 13A

【図１４】

ライブラリのリスク一覧

ライブラリ	リスク
L123	高
L345	低
L911	低
L912	低
…	…

Fig. 14B

リスク値一覧

ライブラリ	リスク値
L123	F11
L345	F12
L911	F13
L912	F14
…	…

Fig. 14A

【図１５】

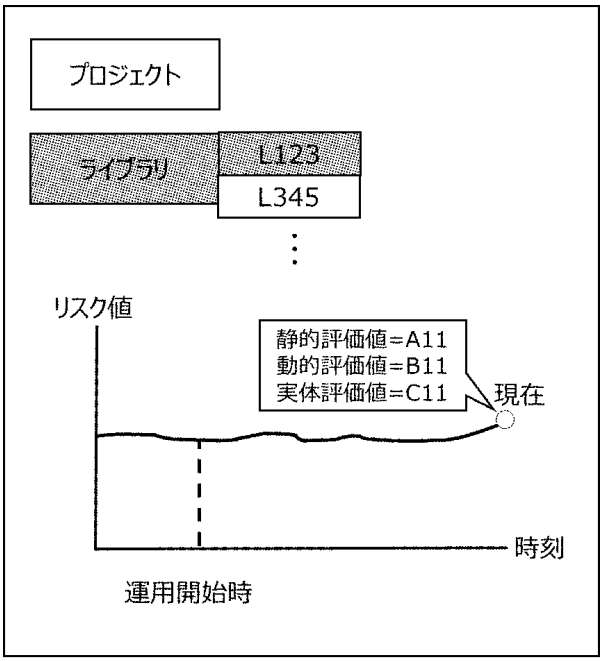

Fig. 15

【図１６】

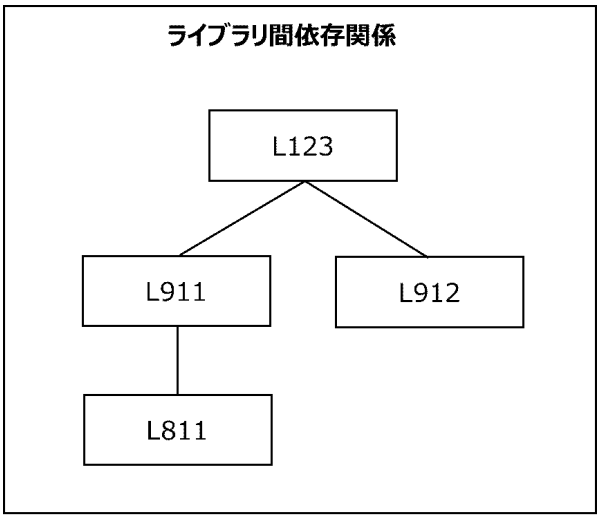

Fig. 16

【図１７】

メタデータの詳細

メタデータ（指標）	値
最新コミットとの差	3コミット遅れ
コミット数/Day	0.1
コミッター人数	4
平均コミットサイズ	30L
平均Issue解決期間	10 Day
Issue発生率	3/Week
⋮	⋮

Fig. 17

【図１８】

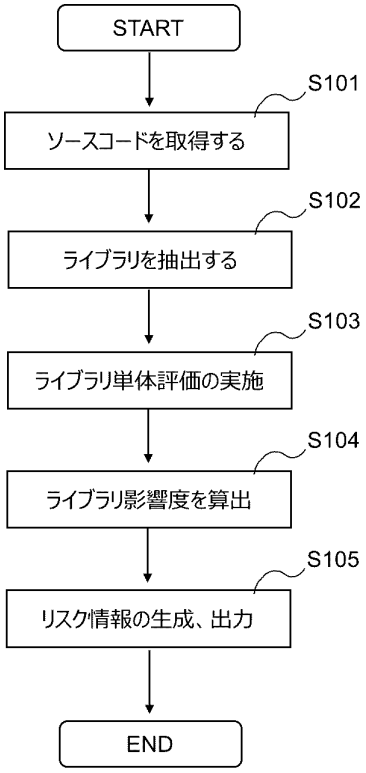

Fig. 18

【図１９】

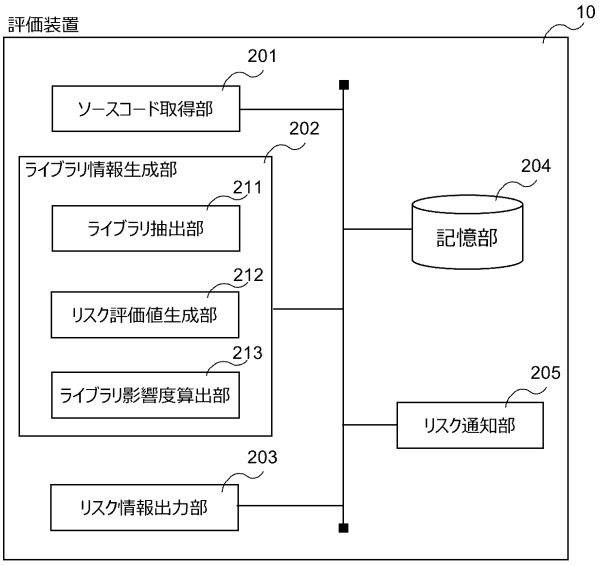

Fig.19

【図２０】

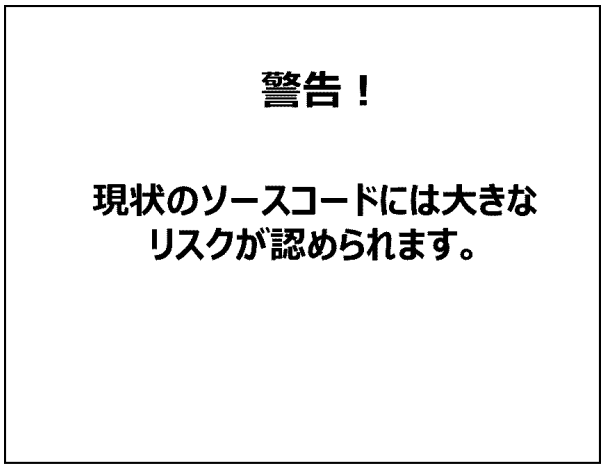

Fig. 20

【図２１】

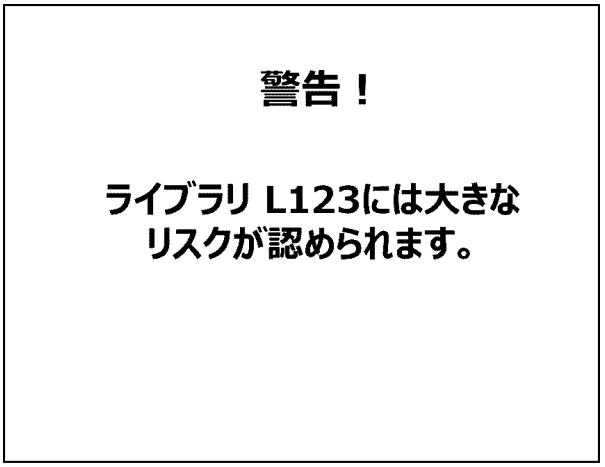

Fig. 21

【図２２】

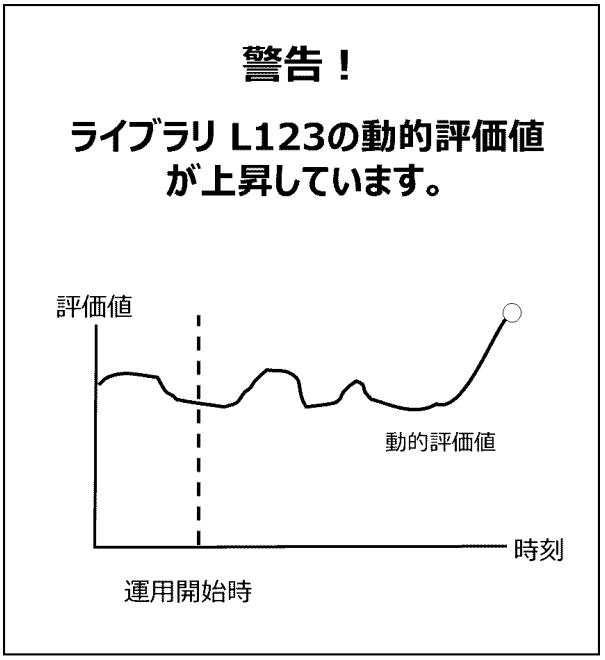

Fig. 22

【図２３】

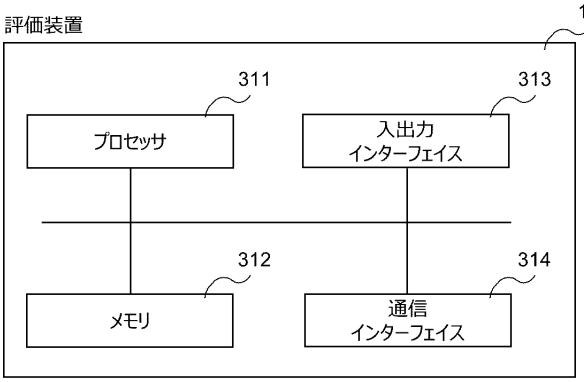

Fig. 23

【図２４】

ライブラリ評価結果

ライブラリ	静的評価値	動的評価値	実体評価値	リスク要因１	リスク要因２
L123	A11	B11	C11	L911(A13)	L811(A15)
L345	A12	B12	C12	None	None
L911	A13	B13	C13	None	None
L912	A14	B14	C14	None	None
⋮	⋮	⋮	⋮	⋮	⋮

Fig. 24

フロントページの続き

東京都港区芝五丁目７番１号 日本電気株式会社内

審査官　岸野 徹

(56)参考文献　米国特許出願公開第２０１５／０００７３３０（ＵＳ，Ａ１）
米国特許出願公開第２０１９／０２２７９０２（ＵＳ，Ａ１）
米国特許出願公開第２０１５／０２６８９４８（ＵＳ，Ａ１）
米国特許出願公開第２０１８／０３４９６１４（ＵＳ，Ａ１）
特開２００１－１８４４４１（ＪＰ，Ａ）
特開２０１０－１９１９２２（ＪＰ，Ａ）
OSSリスク管理ツール Checkmarx CxOSA，Prometech Simulation Conference 2018，2018年12月13日，pp.1－2

(58)調査した分野　(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　８／７７
Ｇ０６Ｆ　２１／５７