(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4669934号
(P4669934)
(45)発行日　平成23年4月13日(2011.4.13)　(24)登録日　平成23年1月28日(2011.1.28)

(51)Int.Cl.　F I
G06F 21/22 (2006.01)　G06F 9/06 660L

請求項の数 18 (全 14 頁)

(21)出願番号　特願2005-171372(P2005-171372)
(22)出願日　平成17年6月10日(2005.6.10)
(65)公開番号　特開2006-344160(P2006-344160A)
(43)公開日　平成18年12月21日(2006.12.21)
審査請求日　平成20年5月28日(2008.5.28)

(73)特許権者　504143441
国立大学法人　奈良先端科学技術大学院大学
奈良県生駒市高山町８９１６－５
(74)代理人　100064746
弁理士　深見　久郎
(74)代理人　100085132
弁理士　森田　俊雄
(74)代理人　100083703
弁理士　仲村　義平
(74)代理人　100096781
弁理士　堀井　豊
(74)代理人　100098316
弁理士　野田　久登

最終頁に続く

(54)【発明の名称】プログラム変換装置、実行支援装置、それらの方法およびそれらのコンピュータ・プログラム

(57)【特許請求の範囲】
【請求項１】
本体プログラムを難読化するプログラム変換装置であって、
前記本体プログラムのメソッドの呼出しの記述を動的呼出しの記述に変更するための呼出変更手段と、
前記呼出変更手段によって変更された後の動的呼出しを指定する文字列を暗号化して前記本体プログラムを変更するための暗号化手段とを含む、プログラム変換装置。
【請求項２】
前記プログラム変換装置が、フィールドを参照する記述またはフィールドに値を代入する記述を、それぞれフィールドを動的に参照する記述またはフィールドに値を動的に代入する記述に変更するためのフィールド変更手段と、
前記フィールド変更手段によって変更された後の記述に含まれるフィールドの名称を暗号化して前記本体プログラムを変更するための暗号化手段とを含む、請求項１記載のプログラム変換装置。
【請求項３】
前記呼出変更手段は、前記動的呼出しの記述を別のクラスのメソッドを経由して呼出すように変更し、
前記フィールド変更手段は、フィールドを動的に参照する記述またはフィールドに値を動的に代入する記述を前記別のクラスのメソッドを経由して行なうように変更する、請求項２記載のプログラム変換装置。

【請求項４】
前記プログラム変換装置はさらに、前記本体プログラムに含まれる変数の型を変更するための変数型変更手段を含む、請求項１～３のいずれかに記載のプログラム変換装置。
【請求項５】
難読化された本体プログラムの実行を支援する実行支援装置であって、
前記難読化された本体プログラムからのメソッドの動的呼出しに応じて、暗号化後の値から文字列を復号するための復号手段と、
前記復号手段によって復号された文字列を用いてメソッドを実行するための実行手段とを含む、実行支援装置。
【請求項６】
前記実行支援装置が、前記難読化された本体プログラムからのフィールドの動的参照またはフィールドへの動的代入の要求に応じて、前記フィールドの名称を復号するための復号手段と、
前記復号手段によって復号された名称を用いてフィールドの動的参照またはフィールドの動的代入を行なうためのフィールドアクセス手段とを含む、請求項５記載の実行支援装置。
【請求項７】
コンピュータに本体プログラムを難読化させるプログラム変換方法であって、
前記本体プログラムのメソッドの呼出しの記述を動的呼出しの記述に変更するステップと、
前記変更された後の動的呼出しを指定する文字列を暗号化して前記本体プログラムを変更するステップとを含む、プログラム変換方法。
【請求項８】
前記プログラム変換方法が、フィールドを参照する記述またはフィールドに値を代入する記述を、それぞれフィールドを動的に参照する記述またはフィールドに値を動的に代入する記述に変更するステップと、
前記変更された後の記述に含まれるフィールドの名称を暗号化して前記本体プログラムを変更するステップとを含む、請求項７記載のプログラム変換方法。
【請求項９】
前記プログラム変換方法が、動的呼出しの記述を別のクラスのメソッドを経由して呼出すように変更するステップと、
フィールドを動的に参照する記述またはフィールドに値を動的に代入する記述を前記別のクラスのメソッドを経由して行なうように変更するステップとを含む、請求項８記載のプログラム変換方法。
【請求項１０】
前記プログラム変換方法はさらに、前記本体プログラムに含まれる変数の型を変更するステップを含む、請求項７～９のいずれかに記載のプログラム変換方法。
【請求項１１】
コンピュータに難読化された本体プログラムの実行を支援させる実行支援方法であって、
前記難読化された本体プログラムからのメソッドの動的呼出しに応じて、前記呼出しに含まれる暗号化後の値から文字列を復号するステップと、
前記復号された文字列を用いてメソッドを実行するステップとを含む、実行支援方法。
【請求項１２】
前記実行支援方法が、前記難読化された本体プログラムからのフィールドの動的参照またはフィールドへの動的代入の要求に応じて、前記フィールドの名称を復号するステップと、
前記復号された名称を用いてフィールドの動的参照またはフィールドの動的代入を行なうステップとを含む、請求項１１記載の実行支援方法。
【請求項１３】

本体プログラムを難読化するプログラム変換方法をコンピュータに実行させるためのコンピュータ・プログラムであって、
前記プログラム変換方法は、前記本体プログラムのメソッドの呼出しの記述を動的呼出しの記述に変更するステップと、
前記変更された後の動的呼出しを指定する文字列を暗号化して前記本体プログラムを変更するステップとを含む、コンピュータ・プログラム。
【請求項１４】
前記コンピュータ・プログラムが、フィールドを参照する記述またはフィールドに値を代入する記述を、それぞれフィールドを動的に参照する記述またはフィールドに値を動的に代入する記述に変更するステップと、
前記変更された後の記述に含まれるフィールドの名称を暗号化して前記本体プログラムを変更するステップとを含む、請求項１３記載のコンピュータ・プログラム。
【請求項１５】
前記コンピュータ・プログラムが、前記動的呼出しの記述を別のクラスのメソッドを経由して呼出すように変更するステップと、
フィールドを動的に参照する記述またはフィールドに値を動的に代入する記述を前記別のクラスのメソッドを経由して行なうように変更するステップとを含む、請求項１４記載のコンピュータ・プログラム。
【請求項１６】
前記プログラム変換方法はさらに、前記本体プログラムに含まれる変数の型を変更するステップを含む、請求項１３～１５のいずれかに記載のコンピュータ・プログラム。
【請求項１７】
難読化された本体プログラムの実行を支援する実行支援方法をコンピュータに実行させるためのコンピュータ・プログラムであって、
前記実行支援方法は、前記難読化された本体プログラムからのメソッドの動的呼出しに応じて、前記呼出しに含まれる暗号化後の値から文字列を復号するステップと、
前記復号された文字列を用いてメソッドを実行するステップとを含む、コンピュータ・プログラム。
【請求項１８】
前記コンピュータ・プログラムが、前記難読化された本体プログラムからのフィールドの動的参照またはフィールドへの動的代入の要求に応じて、前記フィールドの名称を復号するステップと、
前記復号された名称を用いてフィールドの動的参照またはフィールドの動的代入を行なうステップとを含む、請求項１７記載のコンピュータ・プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、ソフトウェアプロテクション技術に関し、特に、与えられたプログラムをより解析の困難なプログラムに変換するプログラム変換装置、変換されたプログラムの実行を支援する実行支援装置、それらの方法およびそれらのコンピュータ・プログラムに関する。
【背景技術】
【０００２】
近年、ソフトウェアプロテクションに関する研究が盛んに行なわれている。このソフトウェアプロテクションとは、ソフトウェアの改ざん、解析、コピー、再利用（盗用）などの攻撃からソフトウェアシステムを保護する技術の総称であり、難読化、暗号化、多様化、電子透かし、バースマークなどの要素技術がある。
【０００３】
これらのソフトウェアプロテクション技術の中で、攻撃のしにくさという観点から、追加のプロテクション機構を持たず、プログラム自身を解析しにくくする方式であるプログ

ラムの難読化が特に注目されている。

【０００４】

難読化とは、与えられたプログラムをより複雑なプログラムに変換する技術であり、レイアウト難読化、データ難読化、制御フロー難読化などがある。難読化されたプログラムは、難読化前のプログラムと同一の機能を持つが、理解や解析がより困難となっている。これに関連する技術として、以下に示すような特許文献１～３と、非特許文献１～２とがある。

【特許文献１】特開２００５－４９９２５号公報

【特許文献２】特開２００４－１９２０６８号公報

【特許文献３】米国特許第６，１０２，９６６号

【非特許文献１】Toshio Ogiso and Yusuke Sakabe and Masakazu Soushi and Mitsuko Miyaji, "Software obfuscation on a theoretical basis and its implementation," IEICE Transactions on Fundamentals, Vol.E86-A, No.1, pp 176-186 Jan 2003.

【非特許文献２】Yusuke Sakabe and Masakazu Soshi and Atsuko Miyaji, "Java（登録商標） Obfuscation with a Theoretical Basis for Building Secure Mobile Agents," Lecture Notes in Computer Science, Vol. 2828, pp. 89-103, 2003.

【発明の開示】

【発明が解決しようとする課題】

【０００５】

近年作成されるプログラムは、関数やプロシージャ、メソッドといった単位に処理内容が分けられることが多い。特に、オブジェクト指向言語で書かれたプログラムは、メソッドとフィールドとを持つクラスという単位にプログラムが分割され、多くのクラスがまとまり、協調することで一つのアプリケーションが構成される。

【０００６】

このようなオブジェクト指向言語で書かれたプログラムを理解するには各クラスの役割、各クラス間の関係を把握することが必要となる。その理解のためには、クラスの各メソッドでどのような処理が行なわれているのかを知る必要がある。オブジェクト指向言語で書かれたプログラムではライブラリメソッドを含む様々なクラスのメソッドを呼出すことで処理が進むため、呼出されるメソッドをわからなくすることで、クラスの役割や各クラス間の関係の理解を妨げることが可能である。

【０００７】

非特許文献１は、メソッドの呼出し処理を隠蔽する技術であるが、呼出される可能性のあるメソッド群の名前を隠蔽することができない。非特許文献２は、クラスのメソッドにオーバロードの関係を持たせることにより、呼出されるメソッドを隠蔽する技術であるが、ライブラリメソッドの呼出しに対してはこの処理を行なうことができない。また、いずれも呼出されるメソッドの引数の型および返り値の型も隠蔽することができない。当該引数等の型に関する情報から、呼出そうとするメソッドを類推することが可能であるため、これら型を隠蔽しないと、難読化の程度を高めることができない。しかし、単に型を変更するのみでは、難読化後のプログラムの実行に支障をきたすため、型の変更または削除は困難であった。

【０００８】

したがって、上述した特許文献１～３または非特許文献１～２の技術を用いたとしても、メソッドの呼出しの隠蔽、モジュールの名前の隠蔽および、呼出されるモジュールの引数の型および返り値の型の隠蔽のすべてを同時に行なうことはできず、プログラムの難読化の程度を一定以上に高めることは不可能であった。

【０００９】

本発明は、上記問題点を解決するためになされたものであり、第１の目的は、メソッド呼出し処理を隠蔽して、プログラムを解析困難なプログラムに変換するプログラム変換装置、その方法およびそのコンピュータ・プログラムを提供することである。

【００１０】

第２の目的は、プログラム変換装置などによって解析困難に変換された後の本体プログラムの実行を支援することが可能な実行支援装置、その方法およびそのコンピュータ・プログラムを提供することである。

【課題を解決するための手段】

【００１１】

本発明のある局面に従えば、本体プログラムを難読化するプログラム変換装置であって、本体プログラムのメソッドの呼出しの記述を動的呼出しの記述に変更するための呼出変更手段と、呼出変更手段によって変更された後の動的呼出しを指定する文字列を暗号化して本体プログラムを変更するための暗号化手段とを含む。

【００１２】

好ましくは、プログラム変換装置はさらに、フィールドを参照する記述またはフィールドに値を代入する記述を、それぞれフィールドを動的に参照する記述またはフィールドに値を動的に代入する記述に変更するためのフィールド変更手段と、フィールド変更手段によって変更された後の記述に含まれるフィールドの名称を暗号化して本体プログラムを変更するための暗号化手段とを含む。

【００１３】

さらに好ましくは、呼出変更手段は、動的呼出しの記述を別のクラスのメソッドを経由して呼出すように変更し、フィールド変更手段は、フィールドを動的に参照する記述またはフィールドに値を動的に代入する記述を別のクラスのメソッドを経由して行なうように変更する。

【００１４】

さらに好ましくは、プログラム変換装置は、本体プログラムに含まれる変数の型を削除するための変数型変更手段を含む。

【００１５】

さらに好ましくは、プログラム変換装置はさらに、暗号化手段によって暗号化される前の文字列を別のクラスに保存するための文字列保存手段を含む。

【００１６】

本発明の別の局面に従えば、難読化された本体プログラムの実行を支援する実行支援装置であって、難読化された本体プログラムからのメソッドの動的呼出しに応じて、呼出しに含まれる暗号化後の値から文字列を復号するための復号手段と、復号手段によって復号された文字列を用いてメソッドを実行するための実行手段とを含む。

【００１７】

好ましくは、実行支援装置はさらに、難読化された本体プログラムからのフィールドの動的参照またはフィールドへの動的代入の要求に応じて、フィールドの名称を復号するための復号手段と、復号手段によって復号された名称を用いてフィールドの参照またはフィールドへの代入を行なうためのフィールドアクセス手段とを含む。

【００１８】

本発明のさらに別の局面に従えば、コンピュータに本体プログラムを難読化させるプログラム変換方法であって、本体プログラムに記述されたメソッドの呼出しを動的呼出しに変更するステップと、変更された後の動的呼出しを指定する文字列を暗号化して本体プログラムを変更するステップとを含む。

【００１９】

本発明のさらに別の局面に従えば、コンピュータに難読化された本体プログラムの実行を支援させる実行支援方法であって、難読化された本体プログラムからのメソッドの動的呼出しに応じて、呼出しに含まれる暗号化後の値から文字列を復号するステップと、復号された文字列を用いてメソッドを実行するステップとを含む。

【００２０】

本発明のさらに別の局面に従えば、本体プログラムを難読化するプログラム変換方法をコンピュータに実行させるためのコンピュータ・プログラムであって、プログラム変換方法は、本体プログラムに記述されたメソッドの呼出しを動的呼出しに変更するステップと

、変更された後の動的呼出しを指定する文字列を暗号化して本体プログラムを変更するステップとを含む。

【００２１】

本発明のさらに別の局面に従えば、難読化された本体プログラムの実行を支援する実行支援方法をコンピュータに実行させるためのコンピュータ・プログラムであって、実行支援方法は、難読化された本体プログラムからのメソッドの呼出しに応じて、呼出しに含まれる暗号化後の値から文字列を復号するステップと、復号された文字列を用いてメソッドを実行するステップとを含む。

【発明の効果】

【００２２】

本発明のある局面によれば、本体プログラム中のメソッド呼出しを動的に行なうように変更した後、動的呼出しで指定されるメソッドの名前を暗号化するため、どのメソッドの呼出しであるのかを隠蔽することができ、プログラムの理解を困難にすることが可能となる。

【００２３】

また、本体プログラム中のフィールドの参照／代入の記述を別のクラスのメソッド経由で行なうように変更した後、フィールドの名称を暗号化して扱うため、どのフィールドの参照または代入であるのかを隠蔽することができ、プログラムの理解を困難にすることが可能となる。

【００２４】

また、変数型変更手段が、本体プログラムに含まれる変数の型を変更するので、引数や返り値の型などのプログラム中に現れる全ての変数の型情報を隠蔽することができ、プログラムの理解をさらに困難にすることが可能となる。

【００２５】

また、呼出し変更手段およびフィールド変更手段が、動的に行なうように変更された呼出し、またはフィールドの参照もしくはフィールドへの代入の記述を別のクラスのメソッドを経由して実行するように変更するので、この別のクラスに、難読化された本体プログラムの実行を支援させることが可能となる。

【００２６】

本発明の別の局面によれば、復号手段が、難読化された本体プログラムからのメソッドの呼出しに応じて、呼出しに含まれる暗号化後の値から文字列を復号し、実行手段が復号された文字列を用いてメソッドを実行するので、プログラム変換装置などによって難読化された後の本体プログラムの実行を支援することが可能となる。

【００２７】

また、復号手段が、難読化された本体プログラムからのフィールドの参照またはフィールドへの代入の要求に応じて、フィールドの名称を復号し、フィールドアクセス手段が復号された名称を用いてフィールドの参照またはフィールドの代入を行なうので、プログラム変換装置などによって変換された後の本体プログラムの実行の際の支援がさらに容易となる。

【発明を実施するための最良の形態】

【００２８】

本発明の実施の形態におけるプログラム変換装置は、Ｊａｖａ（登録商標）などのオブジェクト指向言語で記述されたプログラムのメソッドの呼出しおよびフィールドの参照／代入の隠蔽、変数の型を変更するように、元のプログラムを動的呼び出しを利用したプログラムに変換する。そして、実行支援装置がこの変換されたプログラムの呼出しなどを解釈し、メソッドの実行などを行なう。なお、この実行支援装置は、動的呼出しなど、本体プログラムの実行を支援するクラスとして提供され、以下DynamicCallerクラスとも呼ぶ。さらに、プログラム変換装置中で、呼出しなどが経由するように変更される別のクラスのメソッドが実行支援装置の役割を果たす実施形態も可能である。

【００２９】

図１は、本発明の実施の形態におけるプログラム変換装置および実行支援装置の構成例を示すブロック図である。プログラム変換装置および実行支援装置は、コンピュータ本体１、ディスプレイ装置２、ＦＤ（Flexible Disk）４が装着されるＦＤドライブ３、キーボード５、マウス６、ＣＤ－ＲＯＭ（Compact Disc-Read Only Memory）８が装着されるＣＤ－ＲＯＭ装置７、およびネットワークに接続されるネットワーク通信装置９を含む。プログラム変換プログラムおよび実行支援プログラムは、ＦＤ４またはＣＤ－ＲＯＭ８等の記録媒体によって供給される。プログラム変換プログラムおよび実行支援プログラムがコンピュータ本体１によって実行されることによって、プログラム変換装置および実行支援装置全体の制御が行なわれる。また、プログラム変換プログラムおよび実行支援プログラムは他のコンピュータより通信回線を経由し、コンピュータ本体１に供給されてもよい。

【００３０】

図１に示すコンピュータ本体１は、ＣＰＵ（Central Processing Unit）１０と、ＲＯＭ（Read Only Memory）１１と、ＲＡＭ（Random Access Memory）１２と、ハードディスク１３とを含む。ＣＰＵ１０は、ディスプレイ装置２、ＦＤドライブ３、キーボード５、マウス６、ＣＤ－ＲＯＭ装置７、ネットワーク通信装置９、ＲＯＭ１１、ＲＡＭ１２またはハードディスク１３との間でデータを入出力しながら処理を行う。ＦＤ４またはＣＤ－ＲＯＭ８に記録されたプログラム変換プログラムおよび実行支援プログラムは、ＣＰＵ１０によりＦＤドライブ３またはＣＤ－ＲＯＭ装置７を介してハードディスク１３に格納される。ＣＰＵ１０は、ハードディスク１３から適宜プログラム変換プログラムおよび実行支援プログラムをＲＡＭ１２にロードして実行することによって、プログラム変換装置および実行支援装置全体の制御が行なわれる。

【００３１】

図２は、本発明の実施の形態におけるプログラム変換装置の機能的構成を示すブロック図である。このプログラム変換装置２１は、本体プログラム２３に記述されたアクセス制限を変更するアクセス制限変更部３１と、本体プログラム２３中に存在するメソッドをDynamicCallerクラス２２経由で呼出すように変更するメソッド呼出変更部３２と、本体プログラム２３中に存在するフィールドの値を参照／代入する記述（get/set）をDynamicCallerクラス２２経由で行なうように変更するフィールド参照／代入変更部３３と、本体プログラム２３中に存在するフィールド、ローカル変数の型をObjectクラスに変更する変数型変更部３４と、クラス名、メソッド名およびフィールド名などの文字列を暗号化し、暗号化後の名称に変更する暗号化部３５とを含む。

【００３２】

オブジェクト指向言語のクラスのメンバ（フィールド、メソッド）には、アクセス修飾子が存在する。このアクセス修飾子は、フィールドやメソッドに対して、どのような範囲からアクセス可能であるかを定義するものであり、public、protected、privateが存在する。アクセス制限変更部３１は、クラスのメンバのアクセス修飾子を全てpublicにする。これにより、メソッドが全てDynamicCallerクラス経由で呼出されるようになるため、DynamicCallerクラスはどのようなクラスのメンバにもアクセスできるようになる。

【００３３】

一般に、オブジェクト指向言語で記述されたプログラムにおいては、クラスをインスタンス化してオブジェクトを作成し、そのオブジェクトに対してメッセージを送ることで動作する。クラスの定義は、メタクラスを用いることによってプログラムの実行中に動的に変更することが可能である。本発明の実施の形態においては、このような動的呼出しを利用する。

【００３４】

たとえば、Ｊａｖａ（登録商標）言語であれば、java（登録商標）.lang.Classクラスがメタクラスに相当し、Classクラスのインスタンスは ClassクラスのforName staticメソッドにクラス名を示す文字列を与えることで取得することができる。そして、得られたClassクラスのインスタンスに対してnewInstanceメソッドを実行することによって、対応す

るクラスのオブジェクトを生成することができる。さらに、Classクラスのインスタンスからjava（登録商標）.lang.reflect.Methodオブジェクトを得ることにより、実行するメソッドを与えられた文字列から決定することができる。これにより、メソッドの動的呼出しが実現される。

【００３５】

メソッド呼出し部３２は、このようなメタクラスを用いることにより、オブジェクトの生成からメソッドの実行までを文字列を与えることで行なうように本体プログラム２３を変更する。

【００３６】

また、本発明の実施の形態においては、オブジェクト指向の重要な特徴の１つである多態性を用いてプログラムの解析をより困難にする。多態性とは、同じメッセージに対して、異なるクラスのオブジェクトは異なる動作をすることをいう。すなわち、同じ型の変数に異なるクラスのオブジェクトが入っている場合でも、変数に入っているオブジェクトによって動作が変わり、実行結果が異なる場合がある。

【００３７】

本実施の形態においては、オブジェクトからメタクラスを取得し、そのメタクラスからメソッドやフィールドの定義を得て、それらに対して処理を行なう。したがって、型情報が隠蔽されていても、その変数に入っているオブジェクトがプログラム変換前（難読化前）と同じであれば、本体プログラムの動作に影響を与えない。そこで、変数型変更部３４が、クラスのメンバの型をクラス階層のルートクラスに変更することにより、本体プログラムの型情報が削除され、プログラムの解析をより困難にさせることができる。

【００３８】

暗号化部３５は、メソッド呼出変更部３２およびフィールド参照／代入変更部３３によって変更されたクラス名、メソッド名などの文字列を暗号化し、暗号化後の名称に変更する。なお、暗号化部３５はメソッド呼出変更部３２によって変更されたメソッド名を暗号化する第１暗号化部と、フィールド参照／代入変更部３３によって変更されたフィールド名称を暗号化する第２暗号化部とに分けてもよい。なお、本実施の形態においては、暗号化の方法として共通鍵暗号を用いているが、対称鍵暗号、またはハッシュ関数を用いても実現可能である。共通鍵方式および対称鍵方式を用いた場合には、プログラム変換装置または実行支援装置に復号鍵を保存することが必要である。ハッシュ関数を用いた場合には、ハッシュ化前の文字列を保持するテーブルを実行支援装置２２に保存することにより、ハッシュ化前の文字列の復号を可能とせしめる必要がある。

【００３９】

図３は、本発明の実施の形態におけるプログラム変換装置２１の処理手順を説明するためのフローチャートである。プログラム変換装置２１の処理手順を、図３に示すフローチャートおよび図４～図７に示すプログラム変換例を参照しながら説明する。

【００４０】

まず、アクセス制限変更部３１は、本体プログラム２３を取得し（Ｓ１１）、全てのメソッド、フィールドのアクセス制限をpublicに変更する（Ｓ１２）。なお、図４～図７に示す変換前の本体プログラム１００は、アクセス制限がpublicに変更された後のプログラムを示している。

【００４１】

次に、メソッド呼出変更部３２は、全てのメソッド呼出しをDynamicCallerクラス２２経由の呼出しに変更する（Ｓ１３）。なお、本発明の実施の形態においては、動的呼出しを別のクラスのメソッド、つまりDynamicCallerクラスを経由せずに実行することもできる。その場合、DynamicCallerが行なう処理機能を本体プログラムに持たせる必要がある。

【００４２】

図４は、変換前の本体プログラム１００とメソッド呼出しを変更した後の本体プログラム２００とを示す図である。このプログラムはHelloWorldクラスを定義して実行するもの

であり、メソッド呼出変更部３２によって変換前の本体プログラム１００の記述１０１および１０２のそれぞれが、変更後の本体プログラム２００の記述２０１および２０２に変更されている。記述２０１に示すように、新たなオブジェクトとして“Hello World”が変数oa1に代入され、DynamicCallerクラス経由でjava（登録商標）.io.PrintStreamクラスのprintlnメソッドが呼出されるように変更される。このjava（登録商標）.io.PrintStreamは、上述したメタクラスを用いてメソッドの実行を行うときに指定される文字列である。この文字列を用いて動的呼出しが行なわれる。

【００４３】

また、記述２０２に示すように、DynamicCallerクラス２２経由でnewInstanceメソッドを実行することにより、HelloWorldクラスのインスタンスが生成され、生成されたインスタンスが変数o1に代入される。そして、DynamicCallerクラス２２経由でHelloWorldクラスのhelloメソッドが呼出される。このとき、メソッドを呼出すためのインスタンスが変数o1である。

【００４４】

次に、フィールド参照／代入変更部３３は、本体プログラム２３中に存在するフィールドの値の参照／代入（get/set）を全てDynamicCaller２２経由で行なうように変更する（Ｓ１４）。

【００４５】

図５は、変換前のプログラム１００とフィールドの値の参照／代入を変更した後の本体プログラム３００とを示す図である。フィールド参照／代入変更部３３によって、図４に示す記述２０１の最後の行が、図５に示す記述３０１および３０２に変更されている。図５の記述３０２に示すように、図４の記述２０１の最後の行のSystem.outが変数o1に置換される。また、図５の記述３０１が追加され、DynamicCallerクラス２２経由でjava（登録商標）.lang.Systemクラスのoutフィールドの値が参照されて変数o1に代入される。

【００４６】

次に、変数型変更部３４は、全てのフィールド、ローカル変数の型をクラス階層のルートクラスであるObjectクラスに変更する（Ｓ１５）。

【００４７】

図６は、変換前の本体プログラム１００と変数の型がObjectクラスに変更された後の本体プログラム４００とを示す図である。変数型変更部３４によって、図５に示す記述３０１と図４に示す記述２０２の最初の行とが、それぞれ図６に示す記述４０１および４０２に変更されている。図６の記述４０１に示すように、変数の型がPrintStreamである変数o1の型が、Objectクラスに変更されている。また、図６の記述４０２に示すように、変数の型がHelloWorldである変数o1の型が、Objectクラスに変更されている。

【００４８】

次に、暗号化部３５は、クラス名、メソッド名などを暗号化し、本体プログラム２３内の名称を暗号化後の名称に変更する（Ｓ１６）。

【００４９】

図７は、変換前の本体プログラム１００と暗号化された名称に変更された後の本体プログラム５００とを示す図である。図７に示すように、暗号化部３５によってクラス名、フィールド名およびメソッド名が暗号化されて、記述５０１～５０３のように変更される。図７の記述５０１はクラス名java（登録商標）.lang.Systemおよびフィールド名outが変更されたものであり、記述５０２はクラス名java（登録商標）.io.PrintStreamおよびメソッド名printlnが変更されたものであり、記述５０３はクラス名HelloWorldおよびメソッド名helloが変更されたものである。

【００５０】

次に、暗号部３５は、図７に示すような変換後の本体プログラム２４を出力して（Ｓ１７）、処理を終了する。

【００５１】

図８は、本発明の実施の形態における実行支援装置２２の機能的構成を示すブロック図

である。この実行支援装置２２は、プログラム変換装置２１によって変換された後の本体プログラム２４が実行されるときに、変換後本体プログラム２４からの呼出しに応じて動作するものであり、インスタンス生成部４１と、メソッド実行部４２と、フィールド参照部４３と、フィールド代入部４４と、復号部４５とを含む。なお、インスタンス生成部４１、メソッド実行部４２、フィールド参照部４３およびフィールド代入部４４は、それぞれメソッドとして提供される。復号部には、あらかじめ、暗号化部で用いた暗号鍵に対応する復号鍵を格納する。なお、ハッシュ関数を用いて暗号化した場合には、プログラム変換装置における暗号化時に復号部にハッシュ前の名称（文字列）を保存する必要がある。実行支援装置のインスタンス生成部４１、メソッド実行部４２、フィールド参照部４３およびフィールド代入部４４はいずれもDynamicCallerクラス２２内に包含することができる。

【００５２】

インスタンス生成部４１は、変換後本体プログラム２４からの要求に応じて、メタクラスから対象クラスのインスタンスを生成する。すなわち、インスタンス生成部４１は、変換後本体プログラム２４からの要求に含まれる暗号化された値を抽出し、復号部４５は当該値を復号して暗号化前の名称を得る。そして、インスタンス生成部４１は、暗号化前の名称に基づいてインスタンスを生成し、生成したインスタンスを変換後本体プログラム２４に返す。

【００５３】

メソッド実行部４２は、変換後本体プログラム２４からの呼出しに応じて、メタクラスを参照し、対象クラスの指定されたメソッドを実行する。すなわち、メソッド実行部４２は、変換後本体プログラム２４からの呼出しに含まれる暗号化された値を抽出し、復号部４５により復号して暗号化前の名称を得る。そして、メソッド実行部４２は、暗号化前の名称に基づいてメソッドを実行し、実行されたメソッドの返り値を変換後本体プログラム２４に返す。

【００５４】

フィールド参照部４３は、変換後本体プログラム２４からの要求に応じて、メタクラスを参照し、対象クラスの指定されたフィールドを参照する。すなわち、フィールド参照部４３は、変換後本体プログラム２４からの要求に含まれる暗号化された値を抽出し、復号部４５により復号して暗号化前の名称を得る。そして、フィールド参照部４３は、暗号化前の名称に基づいて標準ライブラリに格納されるフィールドの値を参照し、このフィールドの値を変換後本体プログラム２４に返す。

【００５５】

フィールド代入部４４は、変換後本体プログラム２４からの要求に応じて、メタクラスを参照し、対象クラスの指定されたフィールドに値を代入する。すなわち、フィールド代入部４４は、変換後本体プログラム２４からの要求に含まれる暗号化された値を抽出し、復号部４５により復号して暗号化前の名称を得る。そして、フィールド代入部４４は、暗号化前の名称に基づいて標準ライブラリなどに格納されるフィールドに値を代入する。なお、このフィールド代入部４４は、変換後本体プログラム２４に何も返さない。

【００５６】

以上説明したように、本実施の形態におけるプログラム変換装置によれば、全てのクラスのメソッド呼出しを動的呼出しに変換し、動的呼出しに用いる文字列を暗号化するようにしたので、本体プログラム内部に存在するメソッド呼出しを隠蔽することができ、プログラムの理解を困難にすることが可能となった。

【００５７】

また、フィールドの値を参照／代入する際に用いるフィールド名を暗号化するようにしたので、フィールドへのアクセスを隠蔽することができ、プログラムの理解をより困難にすることが可能となった。

【００５８】

また、変数の型（引数の型、返り値の型）を全てクラス階層のルートクラスに変更する

ようにしたので、変数の型を隠蔽することができ、プログラムの理解をより困難にすることが可能となった。

【００５９】

本実施の形態における実行支援装置によれば、復号部４５によって暗号化された名称の元の名称を得て、得られた元の名称を用いてインスタンスの生成、メソッドの実行、フィールドの参照およびフィールドの代入を行なうようにしたので、プログラム変換装置によって変換された後の本体プログラム２４の実行を支援することが可能となった。

【００６０】

今回開示された実施の形態は、すべての点で例示であって制限的なものではないと考えられるべきである。本発明の範囲は上記した説明ではなくて特許請求の範囲によって示され、特許請求の範囲と均等の意味および範囲内でのすべての変更が含まれることが意図される。

【図面の簡単な説明】

【００６１】

【図１】本発明の実施の形態におけるプログラム変換装置および実行支援装置の構成例を示すブロック図である。

【図２】本発明の実施の形態におけるプログラム変換装置の機能的構成を示すブロック図である。

【図３】本発明の実施の形態におけるプログラム変換装置２１の処理手順を説明するためのフローチャートである。

【図４】変換前の本体プログラム１００とメソッド呼出しを変更した後の本体プログラム２００とを示す図である。

【図５】変換前のプログラム１００とフィールドの値の参照／代入を変更した後の本体プログラム３００とを示す図である。

【図６】変換前の本体プログラム１００と変数の型がobjectクラスに変更された後の本体プログラム４００とを示す図である。

【図７】変換前の本体プログラム１００と暗号化された名称に変更された後の本体プログラム５００とを示す図である。

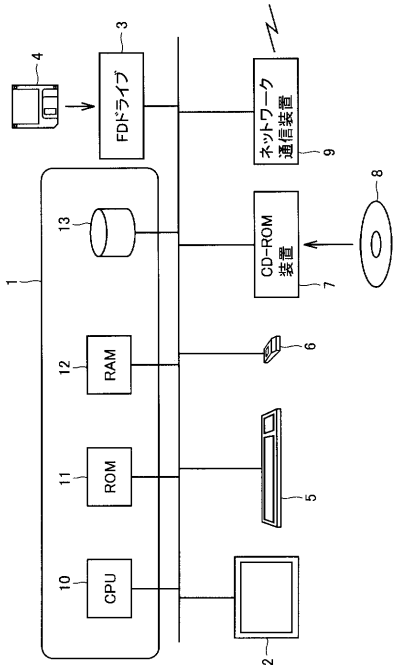
【図８】本発明の実施の形態における実行支援装置２２の機能的構成を示すブロック図である。

【符号の説明】

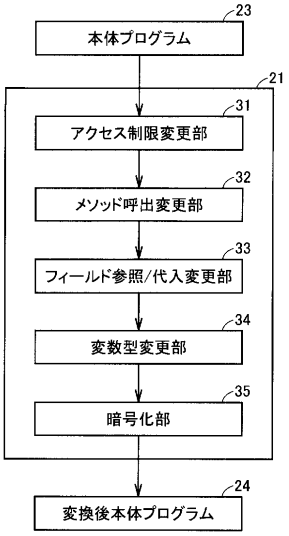
【００６２】

１　コンピュータ本体、２　ディスプレイ装置、３　ＦＤドライブ、４　ＦＤ、５　キーボード、６　マウス、７　ＣＤ－ＲＯＭ装置、８　ＣＤ－ＲＯＭ、９　ネットワーク通信装置、１０　ＣＰＵ、１１　ＲＯＭ、１２　ＲＡＭ、１３　ハードディスク、２１　プログラム変換装置、２２　実行支援装置、２３　本体プログラム、２４　変換後本体プログラム、３１　アクセス制限変更部、３２　メソッド呼出変更部、３３　フィールド参照／代入変更部、３４　変数型変更部、３５　暗号化部、３６　文字列保存部、４１　インスタンス生成部、４２　メソッド実行部、４３　フィールド参照部、４４　フィールド代入部、４５　復号部。

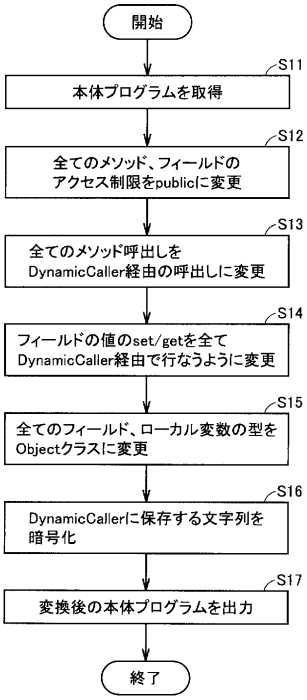
【図１】


FDドライブ
ネットワーク通信装置
CD-ROM装置
RAM
ROM
CPU

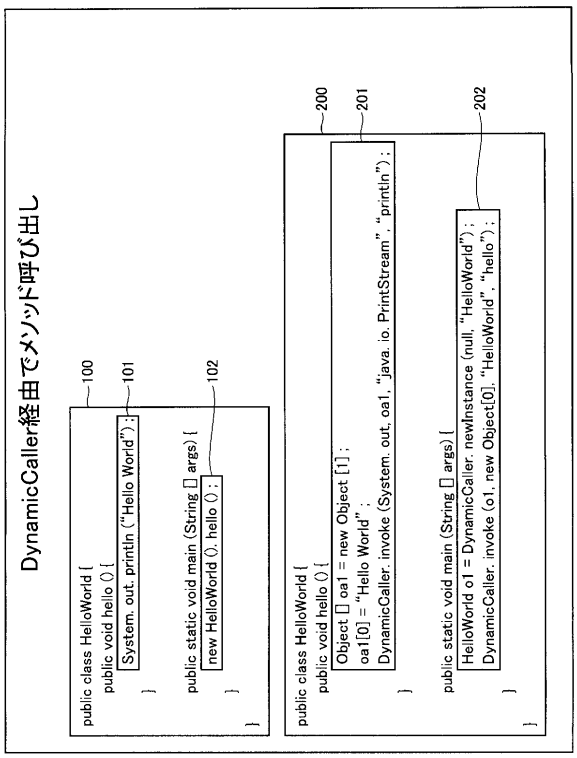
【図２】


本体プログラム
アクセス制限変更部
メソッド呼出変更部
フィールド参照/代入変更部
変数型変更部
暗号化部
変換後本体プログラム

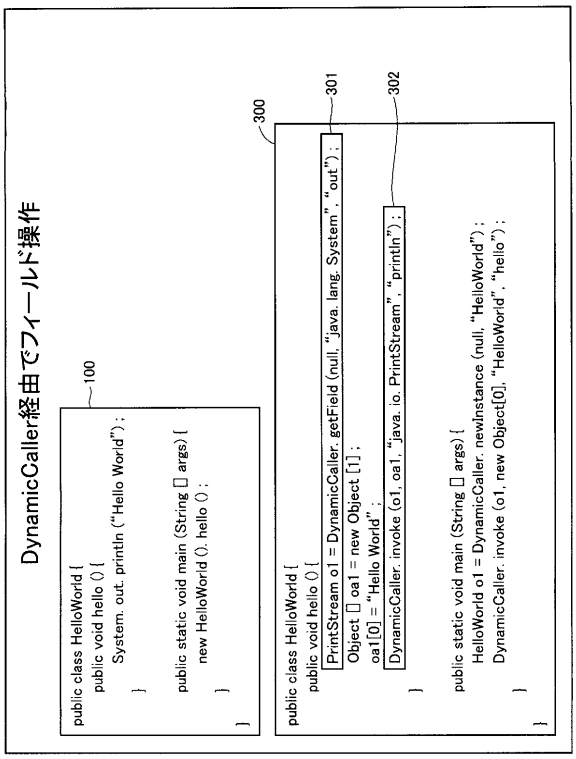
【図３】


開始
本体プログラムを取得
全てのメソッド、フィールドのアクセス制限をpublicに変更
全てのメソッド呼出しをDynamicCaller経由の呼出しに変更
フィールドの値のset/getを全てDynamicCaller経由で行なうように変更
全てのフィールド、ローカル変数の型をObjectクラスに変更
DynamicCallerに保存する文字列を暗号化
変換後の本体プログラムを出力
終了

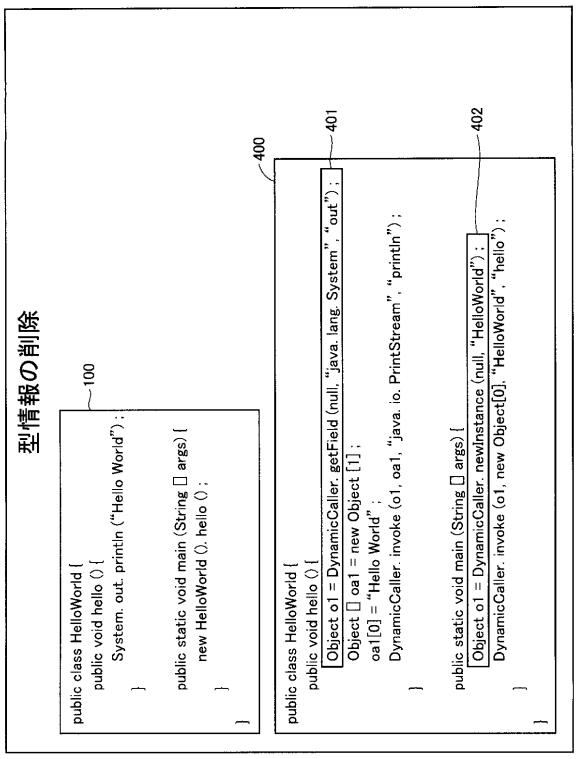
【図４】


DynamicCaller経由でメソッド呼び出し

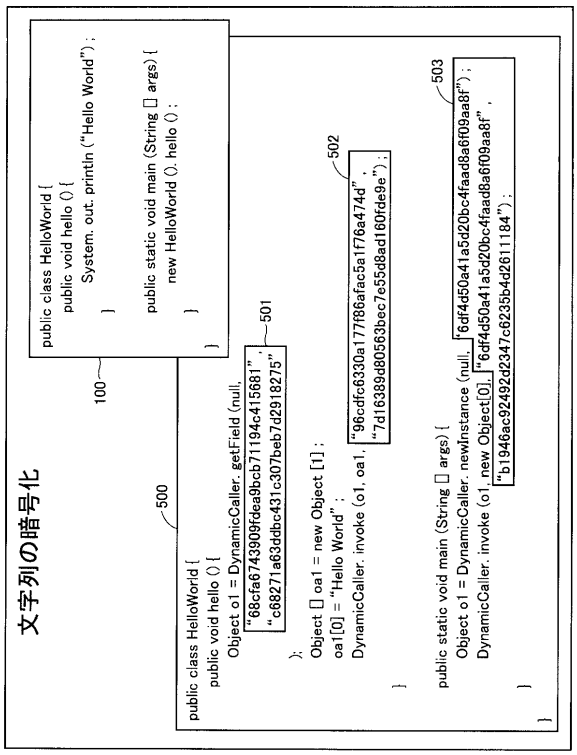
【図５】


DynamicCaller経由でフィールド操作
100
300
301
302

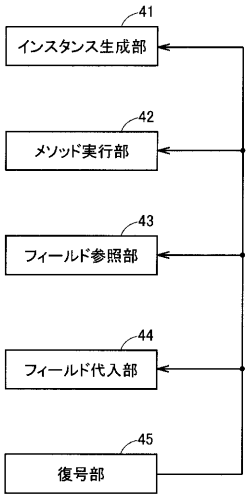
【図６】


型情報の削除
100
400
401
402

【図７】


文字列の暗号化
100
500
501
502
503

【図８】


41
インスタンス生成部
42
メソッド実行部
43
フィールド参照部
44
フィールド代入部
45
復号部

フロントページの続き

(74)代理人 100109162
弁理士 酒井 將行
(72)発明者 玉田 春昭
奈良県生駒市高山町８９１６－５ 奈良先端科学技術大学院大学内
(72)発明者 門田 暁人
奈良県生駒市高山町８９１６－５ 奈良先端科学技術大学院大学内
(72)発明者 中村 匡秀
奈良県生駒市高山町８９１６－５ 奈良先端科学技術大学院大学内
(72)発明者 松本 健一
奈良県生駒市高山町８９１６－５ 奈良先端科学技術大学院大学内

審査官 深沢 正志

(56)参考文献 特開２００３－３８０７５５（ＪＰ，Ａ）
福島 和英，櫻井幸一，メソッド分散によるＪａｖａ言語の難読化手法の提案，コンピュータセキュリティシンポジウム２００２論文集，日本，社団法人情報処理学会，２００２年１０月３０日，ｐ．１９１－１９６

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ２１／２０ － ２１／２４