(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3950010号
(P3950010)
(45)発行日　平成19年7月25日(2007.7.25)　(24)登録日　平成19年4月27日(2007.4.27)

(51)Int.Cl.　　F I
G06F 21/24 (2006.01)　　G06F 12/14 520A
G06F 21/22 (2006.01)　　G06F 9/06 660Z

請求項の数 9　(全 22 頁)

(21)出願番号　特願2002-143608(P2002-143608)
(22)出願日　平成14年5月17日(2002.5.17)
(65)公開番号　特開2003-332978(P2003-332978A)
(43)公開日　平成15年11月21日(2003.11.21)
審査請求日　平成17年5月16日(2005.5.16)

(73)特許権者　392026693
株式会社エヌ・ティ・ティ・ドコモ
東京都千代田区永田町二丁目１１番１号
(74)代理人　100098084
弁理士　川▲崎▼　研二
(74)代理人　100111763
弁理士　松本　隆
(72)発明者　神谷　大
東京都千代田区永田町二丁目１１番１号
株式会社エヌ・ティ・ティ・ドコモ内
(72)発明者　山田　和宏
東京都千代田区永田町二丁目１１番１号
株式会社エヌ・ティ・ティ・ドコモ内

最終頁に続く

(54)【発明の名称】データ処理装置、プログラムおよび記録媒体

(57)【特許請求の範囲】
【請求項１】
１以上のデータと、１以上のアプリケーションを記憶する記憶手段と、
前記１以上のデータのうち少なくとも１のデータを用いた処理の手順を示し、アプリケーションに含まれるコードに従い呼び出され実行されるメソッドを１以上含むオブジェクトを生成する生成手段と、
前記生成手段により生成されるオブジェクトに含まれるメソッドが呼び出されることなくアプリケーションに含まれるコードに従い前記記憶手段に記憶されている前記１以上のデータを用いた処理が行われることを禁止するデータ利用制限手段と、
前記生成手段が生成しようとするオブジェクトに含まれるメソッドに従った処理において用いられるデータの属性が所定の条件を満たす場合、メソッドを呼び出すコードを含むアプリケーションに含まれるコードに従った処理において前記所定の条件を満たす属性を有するデータを利用可能な状態とするメソッドを含むオブジェクトの生成を禁止するオブジェクト生成制限手段と
を備えるデータ処理装置。
【請求項２】
前記記憶手段は、前記１以上のデータの属性を示す属性情報を記憶し、
前記オブジェクト生成制限手段は、前記属性情報により示されるデータの属性が前記所定の条件を満たすか否かを判定し、
ユーザによる入力操作に応じて前記属性情報を書き換える書換手段を備える

請求項１に記載のデータ処理装置。
【請求項３】
前記生成手段は、オブジェクトを生成しようとした際に前記オブジェクト生成制限手段により当該オブジェクトの生成を禁止された場合、当該生成しようとしたオブジェクトと異なる他のオブジェクトであって、当該他のオブジェクトに含まれるメソッドを呼び出すコードを含むアプリケーションに含まれるコードに従った処理において、当該他のオブジェクトに含まれるメソッドに従った処理において用いられるデータを利用可能な状態とするメソッドを１つも含まないオブジェクトを生成する
請求項１に記載のデータ処理装置。
【請求項４】
アプリケーションの属性が所定の条件を満たすか否かを判定する判定手段を備え、
前記データ利用制限手段は、前記判定手段により所定の条件を満たすと判定されたアプリケーションに従った処理に関しては、前記１以上のデータを用いた処理の禁止を行わず、
前記オブジェクト生成制限手段は、前記判定手段により所定の条件を満たすと判定されたアプリケーションに従った処理に関しては、前記オブジェクトの生成の禁止を行わない
請求項１に記載のデータ処理装置。
【請求項５】
アプリケーションを取得する取得手段を備え、
前記判定手段は、前記判定として、アプリケーションが前記取得手段により取得されたアプリケーションであるか否かを判定する
請求項４に記載のデータ処理装置。
【請求項６】
一のアプリケーションに割り当てられた所定の記憶領域以外の記憶領域に記憶されているデータの読み出しを禁止するデータ読出制限手段を備える
請求項１に記載のデータ処理装置。
【請求項７】
前記１以上のアプリケーションのうち少なくとも１のアプリケーションは実行可能な命令コードである実行コードへの変換が行われない限り実行不可能な中間コードで記述されており、
アプリケーションに含まれる中間コードを実行コードに変換する変換手段を備える
請求項１に記載のデータ処理装置。
【請求項８】
１以上のデータと、１以上のアプリケーションを記憶する記憶手段を有するコンピュータに、
前記１以上のデータのうち少なくとも１のデータを用いた処理の手順を示し、アプリケーションに含まれるコードに従い呼び出され実行されるメソッドを１以上含むオブジェクトを生成する生成処理と、
前記生成処理において生成したオブジェクトに含まれるメソッドが呼び出されることなくアプリケーションに含まれるコードに従い前記記憶手段に記憶されている前記１以上のデータを用いた処理が行われることを禁止するデータ利用制限処理と、
前記生成処理において生成しようとするオブジェクトに含まれるメソッドに従った処理において用いられるデータの属性が所定の条件を満たす場合、メソッドを呼び出すコードを含むアプリケーションに含まれるコードに従った処理において前記所定の条件を満たす属性を有するデータを利用可能な状態とするメソッドを含むオブジェクトの生成を禁止するオブジェクト生成制限処理と
を実行させるプログラム。
【請求項９】
請求項８に記載のプログラムを記録したコンピュータ読み取り可能な記録媒体。
【発明の詳細な説明】

【０００１】
【発明の属する技術分野】
この発明は、通信装置のセキュリティを確保するための技術ならびにコンテンツの著作権を保護するための技術に関する。
【０００２】
【従来の技術】
例えば、パケット通信機能を有する携帯電話機やパーソナルコンピュータなどの通信装置は、インターネットに接続されているサーバから様々なプログラムをダウンロードすることができる。
【０００３】
ところで、インターネットなどのオープンネットワークでは、世界中の様々な人々が自由に情報の公開やプログラムの提供を行うことができる。オープンネットワークは、このような利点を有する反面、例えば、悪意の有る個人や団体が通信装置内に記憶されているデータを密かに盗み出すプログラムを提供していたり、あるいは悪意は無いものの通信装置において動作させると不具合を引き起こしてしまうプログラムが提供されてしまうことがある。
【０００４】
したがって、ネットワークを介して提供されたプログラムに対して通信装置の内部および外部のリソースを何ら制限することなくアクセスできるようにしてしまうと、例えば、通信装置内に記憶されているユーザの電話番号やメールアドレス、銀行口座番号などが勝手に読み出され、通信装置の外へ流出してしまうといった事態が生じ得る。
【０００５】
このため、例えば、Java（登録商標）言語で記述されたプログラムを実行することが可能な通信装置においては、ネットワークを介して提供されたJavaプログラムを実行している場合に、このJavaプログラムの実行過程においてアクセスすることのできるリソースをごく限られたものだけに制限しており、これにより信頼性を完全に保証することのできないプログラムが、例えば、通信装置内のアドレス帳データやユーザの個人情報などにアクセスすることを禁じている。
【０００６】
【発明が解決しようとする課題】
上述したアクセス制限の仕組みは、通信装置におけるセキュリティを確保する上で一定の効果を奏するものの、ネットワークを介して提供されるプログラムに対して様々な動作制限を課すことになる。すなわち、このようなアクセス制限は、ネットワークを介して提供されるプログラムが本来有する、通信装置における機能の変更や追加などを自由に行えるという利便性を損なう要因であった。
【０００７】
しかしながら、ネットワークを介して提供されたプログラムの実行に際し、何らアクセスの制限を行なわない場合、前述した悪意の有る個人や団体が提供するプログラムや、通信装置において動作させると不具合を引き起こしてしまうプログラムなどによる被害が、このプログラムを実行した通信装置のみならず、この通信装置と通信を行った他の電子機器にまで及んでしまうおそれがある。
【０００８】
本発明は、以上説明した事情に鑑みてなされたものであり、ネットワークを介して提供されるプログラムの利便性を損なわせることなく、かつ、このようなプログラムに対するセキュリティを確保することのできる通信装置、プログラムおよび記録媒体を提供することを目的としている。
【０００９】
【課題を解決するための手段】
上記課題を解決するために、本発明は、１以上のデータと、１以上のアプリケーションを記憶する記憶手段と、前記１以上のデータのうち少なくとも１のデータを用いた処理の

手順を示し、アプリケーションに含まれるコードに従い呼び出され実行されるメソッドを１以上含むオブジェクトを生成する生成手段と、前記生成手段により生成されるオブジェクトに含まれるメソッドが呼び出されることなくアプリケーションに含まれるコードに従い前記記憶手段に記憶されている前記１以上のデータを用いた処理が行われることを禁止するデータ利用制限手段と、前記生成手段が生成しようとするオブジェクトに含まれるメソッドに従った処理において用いられるデータの属性が所定の条件を満たす場合、メソッドを呼び出すコードを含むアプリケーションに含まれるコードに従った処理において前記所定の条件を満たす属性を有するデータを利用可能な状態とするメソッドを含むオブジェクトの生成を禁止するオブジェクト生成制限手段とを備えるデータ処理装置を提供する。

【００１０】

本発明の好ましい態様において、前記記憶手段は、前記１以上のデータの属性を示す属性情報を記憶し、前記オブジェクト生成制限手段は、前記属性情報により示されるデータの属性が前記所定の条件を満たすか否かを判定し、ユーザによる入力操作に応じて前記属性情報を書き換える書換手段を備える。

【００１１】

また、本発明の好ましい態様において、前記生成手段は、オブジェクトを生成しようとした際に前記オブジェクト生成制限手段により当該オブジェクトの生成を禁止された場合、当該生成しようとしたオブジェクトと異なる他のオブジェクトであって、当該他のオブジェクトに含まれるメソッドを呼び出すコードを含むアプリケーションに含まれるコードに従った処理において、当該他のオブジェクトに含まれるメソッドに従った処理において用いられるデータを利用可能な状態とするメソッドを１つも含まないオブジェクトを生成する。

【００１２】

また、本発明の好ましい態様において、データ処理装置はアプリケーションの属性が所定の条件を満たすか否かを判定する判定手段を備え、前記データ利用制限手段は、前記判定手段により所定の条件を満たすと判定されたアプリケーションに従った処理に関しては、前記１以上のデータを用いた処理の禁止を行わず、前記オブジェクト生成制限手段は、前記判定手段により所定の条件を満たすと判定されたアプリケーションに従った処理に関しては、前記オブジェクトの生成の禁止を行わない。

また、上記好ましい態様において、前記データ処理装置はアプリケーションを取得する取得手段を備え、前記判定手段は、前記判定として、アプリケーションが前記取得手段により取得されたアプリケーションであるか否かを判定する構成としてもよい。

【００１３】

また、本発明の好ましい態様において、前記データ処理装置は一のアプリケーションに割り当てられた所定の記憶領域以外の記憶領域に記憶されているデータの読み出しを禁止するデータ読出制限手段を備える。

また、本発明の好ましい態様において、前記１以上のアプリケーションのうち少なくとも１のアプリケーションは実行可能な命令コードである実行コードへの変換が行われない限り実行不可能な中間コードで記述されており、前記データ処理装置はアプリケーションに含まれる中間コードを実行コードに変換する変換手段を備える。

【００１４】

また、本発明は、１以上のデータと、１以上のアプリケーションを記憶する記憶手段を有するコンピュータに、前記１以上のデータのうち少なくとも１のデータを用いた処理の手順を示し、アプリケーションに含まれるコードに従い呼び出され実行されるメソッドを１以上含むオブジェクトを生成する生成処理と、前記生成処理において生成したオブジェクトに含まれるメソッドが呼び出されることなくアプリケーションに含まれるコードに従い前記記憶手段に記憶されている前記１以上のデータを用いた処理が行われることを禁止するデータ利用制限処理と、前記生成処理において生成しようとするオブジェクトに含まれるメソッドに従った処理において用いられるデータの属性が所定の条件を満たす場合、メソッドを呼び出すコードを含むアプリケーションに含まれるコードに従った処理におい

て前記所定の条件を満たす属性を有するデータを利用可能な状態とするメソッドを含むオブジェクトの生成を禁止するオブジェクト生成制限処理とを実行させるプログラムを提供する。

また、本発明は、上記プログラムを記録したコンピュータ読み取り可能な記録媒体を提供する。

【００１５】

【発明の実施の形態】

以下、図面を参照して本発明の実施形態について説明する。なお、各図において共通する部分には、同一の符号が付されている。

【００１６】

［Ａ－１．実施形態の構成］

＜１．通信システムの構成＞

図１は、この発明の実施形態に係る通信システム１の構成を例示するブロック図である。同図に示すように通信システム１は、コンテンツサーバ１０と、インターネット２０と、移動パケット通信網３０と、携帯電話機４０とを有している。なお、この通信システム１には、本来、多数の携帯電話機４０が収容されるが、図面が煩雑になることを防ぐため、図１には、１つの携帯電話機４０のみを図示している。また、同様の理由により、図１には、それぞれ１つのコンテンツサーバ１０、ゲートウェイサーバ３１および基地局３２のみを図示している。

【００１７】

コンテンツサーバ１０は、インターネット２０および移動パケット通信網３０を介して携帯電話機４０とパケット通信を行う機能を有している。このコンテンツサーバ１０には、携帯電話機４０に提供するプログラムや画像データ、楽曲データなどの種々のコンテンツが格納されている。これらのコンテンツの中には、携帯電話機４０において実行可能なJavaアプリケーションプログラム（以下、JavaＡＰと略称する）が格納されている。このJavaＡＰは、Javaアプレットや Javaアプリケーションなどの、Javaプログラミング言語で記述された携帯電話機４０用のアプリケーションプログラムである。

【００１８】

移動パケット通信網３０は、当該移動パケット通信網３０に収容される携帯電話機４０に対してパケット通信サービスを提供する通信網であり、ゲートウェイサーバ３１と基地局３２とを有している。なお、通信システム１は、移動パケット通信網３０に加え、図示を省略した移動電話網を有している。この移動電話網は、携帯電話機４０に対して一般的な移動電話の通話サービスを提供する。

【００１９】

ゲートウェイサーバ３１は、移動パケット通信網３０用の通信プロトコルとインターネット２０用の通信プロトコルなど、通信プロトコルの異なるデータを相互に変換し、移動パケット通信網３０とインターネット２０とのデータの授受を中継する。また、基地局３２は、移動パケット通信網３０の通信サービスエリア内に多数設置されており、自局３２がカバーする無線セルに在圏している携帯電話機４０と無線通信を行う。

【００２０】

携帯電話機４０は、自機４０が在圏している無線セルをカバーする基地局３２と無線通信を行う。また、この携帯電話機４０は、移動パケット通信網３０およびインターネット２０を介してコンテンツサーバ１０とパケット通信を行う機能を有しており、コンテンツサーバ１０から任意のコンテンツをダウンロードすることができる。

【００２１】

＜２．携帯電話機の構成＞

図２は、携帯電話機４０のハードウェア構成を例示するブロック図である。同図に示すように携帯電話機４０は、無線通信部４０１と、操作入力部４０２と、通話処理部４０３と、通信インタフェース４０４と、ＣＰＵ４０５と、液晶表示部４０６と、記憶部４０７とを有しており、これらの各部はバス４１１により接続されている。

【００２２】
無線通信部４０１は、アンテナ４０１ａを備え、基地局３２との間で行われる無線通信を制御する。この無線通信部４０１は、ＣＰＵ４０５の制御の下、例えば、送話音声に関するデータやパケット通信用のデータなどを搬送波に重畳して送信信号を生成し、この信号を基地局３２へ送信する。また、無線通信部４０１は、基地局３２から送られてくる無線信号をアンテナ４０１ａを介して受信し、この信号を復調して自機４０宛の受話音声に関するデータやパケット通信用のデータなどを得る。
【００２３】
操作入力部４０２は、数字や文字、操作指示などを入力するための複数のキーを有しており、これらのキーの操作に応じた操作信号をＣＰＵ４０５に出力する。また、通話処理部４０３は、例えば、マイクロフォンやスピーカ、音声処理部などを有しており、ＣＰＵ４０５の制御の下、呼の接続／切断を含む通話処理を行う。
【００２４】
通信インタフェース４０４は、通信ケーブルを介して接続された電子機器との有線通信を制御する。なお、この通信インタフェース４０４は、赤外線通信や、ＨｏｍｅＲＦ（Home Radio Frequency）、Bluetooth（登録商標）などの近距離無線通信を制御するものであってもよい。また、ＣＰＵ４０５は、記憶部４０７に格納されている各種プログラムを実行することにより、バス４１１を介して接続されている装置各部を制御する。また、液晶表示部４０６は、液晶表示パネルと、この液晶表示パネルの表示制御を行う駆動回路とを有している。
【００２５】
記憶部４０７は、ＲＯＭ４０８と、ＲＡＭ４０９と、例えば、ＳＲＡＭ（Static－RAM）やＥＥＰＲＯＭ（Electrically Erasable Programmable－ROM）などの不揮発性メモリ４１０とを有している。ＲＯＭ４０８には、例えば、携帯電話機４０用のオペレーティングシステム（以下、ＯＳと略称する）やＷｅｂ（World Wide Web）ブラウザ、Java実行環境を構築するためのソフトウェアなどが記憶されている。また、ＲＡＭ４０９は、ＣＰＵ４０５のワークエリアとして用いられ、ＣＰＵ４０５により実行される各種のプログラムやデータが一時的に記憶される。
【００２６】
不揮発性メモリ４１０には、携帯電話機４０の製品出荷時点から当該携帯電話機４０に組み込まれているアプリケーションプログラムや、コンテンツサーバ１０からダウンロードされたJavaＡＰなどのコンテンツが格納される。加えて、この不揮発性メモリ４１０には、電話番号やメールアドレスなどの情報が記憶されているアドレス帳データ、受信あるいは送信した電子メールデータ、着信や発信に関する履歴データ、電子決済を行うためのユーザの銀行口座番号やクレジットカード番号などの各種データが格納される。
【００２７】
なお、以下、本明細書では、携帯電話機４０の製品出荷時点において既にＲＯＭ４０８や不揮発性メモリ４１０に格納されているアプリケーションソフトウェアを、ダウンロードされたJavaＡＰと区別するため、ネイティブアプリケーションと記載する。このネイティブアプリケーションには、自身がネイティブアプリケーションであることを示す識別情報が付与されている。
【００２８】
また、不揮発性メモリ４１０は、型指定テーブル４１０ａと、ＪＡＲストレージ４１０ｂと、個別スクラッチパッド４１０ｃと、共通スクラッチパッド４１０ｄとを有している。まず、型指定テーブル４１０ａについて図３を参照して説明する。同図に示すように、型指定テーブル４１０ａには、不揮発性メモリ４１０に格納されている各種のデータのうち、アドレス帳データ、電子メールデータ、着信・発信履歴データ、ユーザデータなどの、ダウンロードされたJavaＡＰを実行した場合に当該JavaＡＰが使用する可能性のあるデータについて、データ名と、当該データを扱うオブジェクトの型を「完全カプセル化」型とするのか、それとも「非完全カプセル化」型とするのかを指定する型指定情報とが対応付

けられて登録されている。なお、上述したユーザデータとは、携帯電話機４０のユーザに関する個人情報であって、例えば、ユーザの氏名や年齢、誕生日、銀行口座番号やクレジットカード番号などである。

【００２９】

同図に示すように、型指定情報は、“１”または“０”の１ビットデータであって、型指定情報の値を“１”にセットした場合は、オブジェクトの型が「完全カプセル化」型に指定される一方、型指定情報の値を“０”にセットした場合は、オブジェクトの型が「非完全カプセル化」型に指定される。

【００３０】

なお、カプセル化オブジェクトとは、カプセル化（情報隠蔽）された１以上のデータと、当該カプセル化された各データに対するオブジェクト外部からの操作を可能とするための１以上のメソッドとを有するオブジェクトである。そして、完全カプセル化オブジェクト（tightly encapsulated object）とは、上記カプセル化オブジェクトのうち、オブジェクト内にカプセル化されたデータ自体を当該オブジェクトに対する操作元のプログラム（例えば、ダウンロードされたJavaＡＰ）へ引き渡すメソッドを一つも持たないように構成したオブジェクトである。また、非完全カプセル化オブジェクトとは、上記カプセル化オブジェクトのうち、オブジェクト内のデータ自体を当該オブジェクトに対する操作元のプログラムへ引き渡すメソッドを少なくとも一つ以上有しているオブジェクトである。完全カプセル化オブジェクトと非完全カプセル化オブジェクトとの差異は、オブジェクト内のカプセル化されたデータ自体を操作元のプログラムへ引き渡すメソッドを有しているか否かである。

【００３１】

すなわち、図３に示した型指定テーブル４１０ａには、各データ毎に、当該データを完全カプセル化オブジェクトとして扱うのか、それとも非完全カプセル化オブジェクトとして扱うのかが登録されている。例えば、同図において、アドレス帳データ、電子メールデータ、着信・発信履歴データ、ユーザデータは、携帯電話機４０に記憶されているデータの中でも特にセキュリティ上の重要度が高いデータである。一方、携帯電話機４０にダウンロードされたJavaＡＰは、信頼性を完全に保証することのできないプログラムであって、万一、悪意のある第３者が作成した、データを盗み出すJavaＡＰが携帯電話機４０にダウンロードされた場合であっても、このようなJavaＡＰを介して上述したセキュリティ上の重要度が高いデータが携帯電話機４０の外部へ流出してしまうような事態は、極力、防がなければならない。

【００３２】

したがって、セキュリティ上の重要度が高いデータは完全カプセル化オブジェクトとして扱い、ダウンロードされたJavaＡＰにデータ自体が引き渡されないようにする必要がある。以上のようなことから、型指定テーブル４１０ａにおいて、アドレス帳データ、電子メールデータ、着信・発信履歴データ、ユーザデータには、型指定情報の値として“１”（完全カプセル化型）が登録されている。

【００３３】

また、同図に示した型指定テーブル４１０ａにおいて、コンテンツＡおよびコンテンツＢは、コンテンツサーバ１０からダウンロードされた画像データや音楽データなどのコンテンツである。これらのコンテンツには、コンテンツの提供事業者（以下、コンテンツプロバイダと記載する）により著作権保護フラグが付与されている。ここで、著作権保護フラグは、“１”または“０”の１ビットデータであり、著作権保護フラグの値が“１”にセットされている場合は、このフラグの付与されているコンテンツの著作権を保護しなければならないことを示している。一方、著作権保護フラグの値が“０”にセットされている場合は、このフラグの付与されているコンテンツの著作権が放棄されていることを示している。

【００３４】

ここで、著作権を保護しなければならないコンテンツの場合、ダウンロードされたJavaＡ

Ｐにコンテンツデータ自体を引き渡してしまうと、コンテンツプロバイダが許可を与えていないような利用形態でコンテンツがJavaＡＰにより利用されてしまうおそれや、コンテンツデータがJavaＡＰを介して携帯電話機４０の外部へ不正に転送されてしまうおそれがある。したがって、著作権保護フラグの値として“１”が付与されているコンテンツデータは完全カプセル化オブジェクトとして扱い、ダウンロードされたJavaＡＰにコンテンツデータ自体が引き渡されないようにする必要がある。以上のようなことから、型指定テーブル４１０ａにおいてコンテンツＡには、型指定情報の値として“１”（完全カプセル化型）が登録されている。

【００３５】

一方、著作権保護フラグの値として“０”が付与されているコンテンツデータは、著作権が放棄されているので、完全カプセル化オブジェクトとして扱う必要がない。この場合、ダウンロードされたJavaＡＰにコンテンツデータ自体を引き渡しても何ら問題がなく、また、コンテンツデータ自体を引き渡せるようにした方がJavaＡＰの利便性が高くなる。このため、型指定テーブル４１０ａにおいてコンテンツＢには、型指定情報の値として“０”（非完全カプセル化型）が登録されている。

【００３６】

なお、型指定テーブル４１０ａにおいて、アドレス帳データ、電子メールデータ、着信・発信履歴データ、ユーザデータには、予め型指定情報の値として“１”がセットされている。また、ダウンロードされたコンテンツについては、コンテンツが携帯電話機４０にダウンロードされた際に、このコンテンツに付与されている著作権保護フラグの値に応じた型指定情報の値がＣＰＵ４０５により決定され、コンテンツを識別するコンテンツ識別情報（データ名）とともに型指定テーブル４１０ａに登録される。

【００３７】

また、携帯電話機４０において作成されたデータやユーザがパソコンなどで作成し、通信インタフェース４０４を介して携帯電話機４０に取り込んだ画像データや音楽データなどに対しては、このデータを扱うオブジェクトの型を「完全カプセル化」型とするのか、それとも「非完全カプセル化」型とするのかをユーザが操作入力により設定することができる。さらに、型指定テーブル４１０ａの内容を液晶画面に表示して、著作権保護フラグが付与されたコンテンツを除く各データの型指定情報を、ユーザ自身が操作入力により変更可能な構成としてもよい。

【００３８】

次に、図２に戻り、不揮発性メモリ４１０は、上述した型指定テーブル４１０ａの他に、ＪＡＲストレージ４１０ｂと、個別スクラッチパッド４１０ｃと、共通スクラッチパッド４１０ｄとを有している。ここで、ＪＡＲストレージ４１０ｂ、個別スクラッチパッド４１０ｃおよび共通スクラッチパッド４１０ｄについて説明する前に、まず、携帯電話機４０にダウンロードされるJavaＡＰについて説明する。JavaＡＰは、JavaＡＰの本体プログラムおよび当該本体プログラムの実行に応じて利用される画像ファイルや音声ファイルなどを１つにまとめたＪＡＲ（Java Archive）ファイルと、このＪＡＲファイルのインストールや起動、ネットワークアクセスなどを制御するための各種制御情報が記述されたＡＤＦ（Application Descriptor File）とを有している。

【００３９】

ＪＡＲストレージ４１０ｂおよび個別スクラッチパッド４１０ｃには、ダウンロードされたJavaＡＰ毎に当該JavaＡＰ用の記憶領域が設けられる。ＪＡＲストレージ４１０ｂ内の各記憶領域には、JavaＡＰのＪＡＲファイルが格納される。また、個別スクラッチパッド４１０ｃ内の各記憶領域には、例えば、JavaＡＰがゲームプログラムである場合、今までの得点データやセーブデータなど、JavaＡＰの利用に応じて発生した当該JavaＡＰ用のデータが格納される。さらに、共通スクラッチパッド４１０ｄには、複数のJavaＡＰが共通して使用するデータが格納される。

【００４０】

また、ダウンロードの後、JavaＡＰが携帯電話機４０において実行される場合、このJava

ＡＰの実行に伴って携帯電話機４０がアクセスすることのできるリソースは、このJavaＡＰのダウンロード元のコンテンツサーバ１０（サイト）と、このJavaＡＰに対して割り当てられたＪＡＲストレージ４１０ｂおよび個別スクラッチパッド４１０ｃ内の記憶領域と、共通スクラッチパッド４１０ｄと、のみに制限され、それ以外のリソースにアクセスすることはできない。

【００４１】

＜３．Java実行環境＞

図４は、携帯電話機４０におけるJavaＡＰの実行環境を説明するための図である。同図に示すように本実施形態に係る携帯電話機４０には、JavaＡＰの実行環境を構築するためのソフトウェアとして、ＫＶＭ（K Virtual Machine）と、コンフィギレーションとしてＣＬＤＣ（Connected Limited Device Configuration）を備えるとともにプロファイルとして通信事業者が独自に策定したオリジナルJava拡張プロファイルを備えたＪ２ＭＥ（Java 2 Micro Edition）とが組み込まれている。

【００４２】

ＫＶＭは、小型電子機器用に設計変更されたＪＶＭ（Java Virtual Machine）であって、JavaＡＰの実行ファイル形式であるバイトコードをＣＰＵ４０５がＯＳを介して解釈／実行可能な命令コードに変換する。また、ＣＬＤＣクラスライブラリは、ＣＬＤＣ用のクラスライブラリである。

【００４３】

オリジナルJava拡張ライブラリは、ＣＬＤＣを基礎として携帯電話機に特化した機能を提供するためのクラスライブラリである。このオリジナルJava拡張ライブラリには、例えば、ユーザインタフェースＡＰＩ（Application Program Interface）、ネットワーキングＡＰＩ、スクラッチパッドＡＰＩ、完全カプセル化ＡＰＩ、非完全カプセル化ＡＰＩなどが含まれている。

【００４４】

ここで、ユーザインタフェースＡＰＩは、携帯電話機４０のユーザインタフェース機能をサポートするＡＰＩであり、ネットワーキングＡＰＩは、ＵＲＬ（Uniform Resource Locator）により指定されたネットワークリソースへのアクセスをサポートするＡＰＩである。また、スクラッチパッドＡＰＩは、個別スクラッチパッド４１０ｃや共通スクラッチパッド４１０ｄに対するデータの書き込みや読み出しをサポートするＡＰＩである。さらに、完全カプセル化ＡＰＩは、完全カプセル化オブジェクトを生成するためのＡＰＩであり、非完全カプセル化ＡＰＩは、非完全カプセル化オブジェクトを生成するためのＡＰＩである。

【００４５】

また、携帯電話機４０は、ＣＬＤＣクラスライブラリおよびオリジナルJava拡張ライブラリに加え、メーカ独自拡張ライブラリを有している。このメーカ独自拡張ライブラリは、携帯電話機４０を製造する各メーカがそれぞれ独自の機能を提供するためのクラスライブラリである。

【００４６】

次に、ＪＡＭ（Java Application Manager）は、ＯＳによる制御の下で、携帯電話機４０にダウンロードされたJavaＡＰや、完全カプセル化オブジェクト、非完全カプセル化オブジェクトなどを管理する機能を有している。例えば、ＪＡＭは、JavaＡＰのインストールや更新、削除を行う機能、不揮発性メモリ４１０に格納されているJavaＡＰをリスト表示する機能、JavaＡＰの実行管理（起動や強制終了など）を行う機能、JavaＡＰの実行に伴う携帯電話機４０のアクセスを制限する機能、完全カプセル化オブジェクトや非完全カプセル化オブジェクトの生成、更新、削除を行なう機能などを有している。

【００４７】

また、同図に示すように、電話帳機能やブラウザ機能、ネットワーク通信機能などを提供するネイティブアプリケーションは、ＯＳによる制御の下で直接動作する。

【００４８】

＜４．カプセル化オブジェクトの構成＞
次に、カプセル化オブジェクトについて説明する。図５は、カプセル化オブジェクトについて説明するための模式図である。同図に示すように、カプセル化オブジェクトとは、カプセル化された１以上のデータと、当該カプセル化された各データに対するオブジェクト外部からの操作を可能とするための１以上のメソッドとを有するオブジェクトである。
【００４９】
同図に示す例では、２つのデータ１，２と、２つのメソッド１，２とを有するカプセル化オブジェクトが例示されている。このカプセル化オブジェクト内のデータ１，２は共にカプセル化されているため、オブジェクトの外部からデータ１，２を直接読み書きすることはできない。したがって、例えば、ダウンロードされたプログラムがカプセル化オブジェクト内のデータ１，２に対してアクセスする場合、プログラムは、メソッド１，２を使用して目的のデータ１またはデータ２に対する操作をカプセル化オブジェクトに指令しなければならない。
【００５０】
ここで、同図に示すメソッド１が、例えば、指定されたデータ自体を操作元のプログラムへ引き渡すメソッドであれば、操作元のプログラムは、メソッド１を使用してカプセル化オブジェクト内の任意のデータ１，２を取得することが可能である。また、同図に示すメソッド２が、例えば、指定されたデータを液晶画面に表示させるメソッドであれば、操作元のプログラムは、メソッド２を使用してカプセル化オブジェクト内の任意のデータ１，２を画面表示させることが可能である。ここで注目すべき点は、メソッド２を使用してカプセル化オブジェクト内の任意のデータ１，２を画面表示させたプログラムは、表示させるデータやメソッドを指定してカプセル化オブジェクトへ指令を出すものの、表示させるデータ自体を取得していない点である。
【００５１】
つまり、データそのものを操作元のプログラムに引き渡すメソッドを１つも有していないカプセル化オブジェクト（完全カプセル化オブジェクト）であれば、操作元のプログラムは、オブジェクト内のデータそのものを取得することはできないが、このオブジェクトに備わるメソッドを使用してオブジェクト内のデータに対する操作を行うことはできる。
【００５２】
したがって、アドレス帳データや電子メールデータなどを完全カプセル化オブジェクトとして扱うようにすれば、操作元のプログラムがダウンロードされたJavaＡＰのように信頼性を完全に保証することのできないプログラムであったとしても、当該プログラムにデータ自体を引き渡すことがないので、携帯電話機４０におけるセキュリティを確保することができる。また、同時に、このようなプログラムであったとしても、アドレス帳データや電子メールデータなど、従来はセキュリティを確保する観点から一切のアクセスを認めていなかったデータに対し、完全カプセル化オブジェクトが有するメソッドを用いて操作（アクセス）を行うことができる。
【００５３】
以上のようなことから本実施形態では、アドレス帳データ、電子メールデータ、着信・発信履歴データ、ユーザデータなどのセキュリティ上の重要度の高いデータや、著作権を保護しなければならないコンテンツなど、前述した型指定情報の値として“１”がセットされているデータを完全カプセル化オブジェクトとして扱う一方、セキュリティの重要度が低いデータや著作権が放棄されたコンテンツなど、型指定情報の値として“０”がセットされているデータを非完全カプセル化オブジェクトとして扱う。また、ダウンロードされたJavaＡＰに対して、完全カプセル化オブジェクトや非完全カプセル化オブジェクトへのアクセスを許可する。
【００５４】
図６は、電話帳データに関する非完全カプセル化オブジェクトについて例示する模式図である。なお、本実施形態において、電話帳データはセキュリティ上の重要度が高いため、本来は必ず完全カプセル化オブジェクトとして扱われるデータであるが、ここでは比較の

ため、あえて非完全カプセル化オブジェクトとして扱った場合について説明する。

【００５５】

Javaプログラミング言語では、「private」というアクセス修飾子を用いてオブジェクト内のフィールドをprivateフィールドに宣言することで、当該privateフィールドに格納されるデータのカプセル化を図る。つまり、オブジェクト内のフィールドが全てprivateフィールドである場合、各privateフィールドに格納されているデータをオブジェクトの外部から直接読み書きすることができなくなる。このようにした場合、各privateフィールドに格納されているデータに対してオブジェクトの外部からアクセスするには、このオブジェクトに備わるメソッドを使用してデータに対する操作を当該オブジェクトに指令しなければならない。

【００５６】

同図に示す非完全カプセル化オブジェクトには、２つのprivateフィールドが設けられ、それぞれprivate char value[１]、private char value[２]という電話帳の文字列データが格納されている。また、この非完全カプセル化オブジェクトは、getBytes()、drawString()という２つのメソッドを有している。ここで、getBytes()は、オブジェクト内のデータをバイト配列の形式で操作元のプログラムへ引き渡すメソッドである。したがって、ダウンロードされたJavaＡＰは、このgetBytes()というメソッドを使用して、非完全カプセル化オブジェクト内の電話帳の文字列データ（private char value[１]、private char value[２]）を取得することが可能である。加えて、JavaＡＰは、取得した電話帳の文字列データを当該JavaＡＰのダウンロード元のサーバ（コンテンツサーバ１０）へ送信することなどができる。

【００５７】

また、drawString()は、オブジェクト内のデータを携帯電話機４０の液晶画面に表示させるメソッドである。JavaＡＰは、このdrawString()というメソッドを使用して、非完全カプセル化オブジェクト内の電話帳の文字列データ（private char value[１]、private char value[２]）を液晶画面に表示させることもできる。

【００５８】

一方、図７は、電話帳データに関する完全カプセル化オブジェクトについて例示する模式図である。同図に示す完全カプセル化オブジェクトが図６に示した非完全カプセル化オブジェクトと異なるのは、完全カプセル化オブジェクトは、上述したgetBytes()のように、オブジェクト内のデータそのものを操作元のプログラムへ引き渡すメソッドを有していない点である。

【００５９】

すなわち、完全カプセル化オブジェクトは、カプセル化された上に、オブジェクト内のデータそのものを操作元のプログラムへ引き渡すメソッドを１つも有していない。したがって、ダウンロードされたJavaＡＰは、drawString()というメソッドを使用してオブジェクト内の電話帳の文字列データ（private char value[１]、private char value[２]）を画面表示させることはできるが、電話帳の文字列データそのものを取得することはできない。以上のようなことから、万一、悪意のある第３者が作成した、データを盗み出すJavaＡＰが携帯電話機４０にダウンロードされた場合であっても、このようなJavaＡＰに電話帳データが引き渡されることはなく、当然、電話帳データがサーバなど携帯電話機４０の外部へ送信されることもない。

【００６０】

ところで、drawString()というメソッドを使用してオブジェクト内の電話帳の文字列データを画面表示させる場合、完全カプセル化オブジェクトや非完全カプセル化オブジェクトは、ネイティブアプリケーションとしてＲＯＭ４０８または不揮発性メモリ４１０に格納されている表示制御プログラムを使用して液晶画面に電話帳の文字列を表示させる。この表示制御プログラムからJavaＡＰが表示データを取得することができてしまうと、完全カプセル化オブジェクトや非完全カプセル化オブジェクトを用いた意味がなくなってしまう。

【００６１】
しかしながら、ダウンロードされたJavaＡＰが実行される場合には、前述したようにＪＡＭのアクセス制限機能により、JavaＡＰの実行中に携帯電話機４０がアクセスすることのできるリソースが制限される。ここで、JavaＡＰの実行中に携帯電話機４０のアクセスが許可されるリソースに表示制御プログラムは含まれていないので、JavaＡＰが表示制御プログラムから表示データを取得するようなことは一切あり得ない。
【００６２】
また、カプセル化は、プログラミング言語レベルでのカプセル化と、実行コード（マシン語またはバイトコード）レベルでのカプセル化とが考えられる。プログラミング言語レベルでのカプセル化が完全であっても、実行コードレベルでのカプセル化が完全でなければ、データを完全にカプセル化したとは言えない。例えば、プログラミング言語であるC++を用いたプログラムでも privateフィールドを有するカプセル化オブジェクトを生成することはできる。しかしながら、C++は、単なるプログラミング言語に過ぎないことから、プログラミング言語レベルでのカプセル化しか達成し得ない。
【００６３】
具体的に説明すると、C++を用いたプログラムにより、オブジェクト内の全てのフィールドを privateフィールドとして宣言し、カプセル化オブジェクトを生成した場合、確かに、このオブジェクト内の privateフィールドに格納されているデータを直接読み書きするようなソースコードはコンパイルされることがないので、当然、実行コードが生成されることもない。
【００６４】
しかしながら、このカプセル化は、コンパイラによって保証されているに過ぎず、例えば、悪意のある第３者がコンパイラを改造することで、オブジェクト内のデータを不正に入手することが可能である。また、コンパイラを改造しなくても、悪意のある第３者がハンドアセンブルなどの手段でオブジェクト内のデータを不正に読み出す実行コードを生成するプログラムを作成することも不可能ではない。加えて、ポインタを用いて直接メモリにアクセスしてしまえば、オブジェクト内のデータを入手することができてしまう。
【００６５】
これに対してJavaの場合、private宣言されたフィールドは、 private属性を有するフィールドであることを示すJavaのバイトコードへコンパイルされる。ＫＶＭがクラスファイルをＲＡＭ４０９などへ展開する際も、フィールドの private属性は保持されている。したがって、仮にコンパイラを改造してオブジェクト内の privateフィールドに格納されているデータを不正に読み出すようなバイトコードを生成したとしても、ＫＶＭまたはＪＡＭがこれを検知するので、オブジェクト内のデータを入手することはできない。また、Javaはポインタをサポートしていないので、ポインタを用いて直接メモリにアクセスし、オブジェクト内のデータを入手することもできない。
【００６６】
以上のようなことから、Javaでは、プログラミング言語レベルのみに止まらず、バイトコードレベルでの完全なカプセル化を達成することが可能である。なお、データのカプセル化に際しては、「private」の他に「protected」や「package」などのアクセス修飾子を用いることもできる。
以上が本実施形態に係る通信システム１の構成である。
【００６７】
［Ａ－２．実施形態の動作］
次に、本実施形態の動作について説明する。
なお、携帯電話機４０が以下に述べる動作を行う前提として、携帯電話機４０は、移動パケット通信網３０およびインターネット２０を介してコンテンツサーバ１０とパケット通信を行い、コンテンツサーバ１０からJavaＡＰをダウンロードして不揮発性メモリ４１０に格納しているものとする。また、不揮発性メモリ４１０には、ダウンロードされたJavaＡＰ（コンテンツ）の他に、アドレス帳データや電子メールデータ、ユーザデータなどが

格納されており、型指定テーブル４１０ａには、これら各データについて型指定情報が登録されているものとする。

【００６８】

また、型指定テーブル４１０ａにおいて、アドレス帳データ、電子メールデータ、着信・発信履歴データ、ユーザデータに対しては、型指定情報の値として“１”がセットされている。また、ダウンロードされたコンテンツに対しては、コンテンツが携帯電話機４０にダウンロードされた際に、このコンテンツに付与されている著作権保護フラグの値に応じた型指定情報の値がＣＰＵ４０５により決定され、当該型指定情報およびコンテンツ名が型指定テーブル４１０ａに登録される。

【００６９】

＜１．オブジェクト生成処理＞

まず、携帯電話機４０においてＣＰＵ４０５により実行されるオブジェクト生成処理について図８を参照して説明する。このオブジェクト生成処理は、ＪＡＭの機能としてＣＰＵ４０５により実行されるものであり、例えば、画面表示されたプログラムの一覧リストの中から、実行するプログラムが操作入力により指定された場合などに実行される。なお、プログラムの実行を指示する形態は、操作入力によるものに限定されず、例えば、予め定められた時間毎にプログラムの実行が指示される場合や、既に実行されている他のプログラムから実行が指示される場合、電子メールなどを用いて携帯電話機４０の外部からプログラムの実行が指示される場合などもある。

【００７０】

同図に示すように、まず、携帯電話機４０のＣＰＵ４０５は、実行するプログラムとして操作入力により指定されたプログラムを特定する（ステップＳ１０１）。次いで、ＣＰＵ４０５は、特定したプログラムがダウンロードされたJavaＡＰであるのか、それともネイティブアプリケーションであるのかを判別する（ステップＳ１０２）。前述したようにネイティブアプリケーションには、自身がネイティブアプリケーションであることを示す識別情報が付与されている。したがって、ＣＰＵ４０５は、プログラムに上記識別情報が付与されているか否かを判別することで、このプログラムがダウンロードされたJavaＡＰであるのか、それともネイティブアプリケーションであるのかを判別することができる。

【００７１】

その結果、ＣＰＵ４０５は、プログラムがネイティブアプリケーションであると判別した場合は（ステップＳ１０２：Ｎｏ）、オブジェクト生成処理を終了するとともに、実行するプログラムとして指定されたネイティブアプリケーションを起動する。そして、ＣＰＵ４０５は、起動させたネイティブアプリケーションに基づく処理を行なう。

【００７２】

ここで、実行するプログラムがネイティブアプリケーションである場合は、信頼性が完全に保証できるプログラムであるので、完全カプセル化オブジェクトや非完全カプセル化オブジェクトを用いたり、あるいはネイティブアプリケーションの実行に伴ってＪＡＭのアクセス制限機能を動作させる必要がない。したがって、ネイティブアプリケーションが実行される場合、ＪＡＭによるアクセス制限は一切行われず、ネイティブアプリケーションは、携帯電話機４０内の任意のリソースおよびネットワーク上の任意のリソースにアクセスすることができる。

【００７３】

一方、ＣＰＵ４０５は、プログラムがダウンロードされたJavaＡＰであると判別した場合は（ステップＳ１０２：Ｙｅｓ）、次いで、不揮発性メモリ４１０に格納されている各種のデータの中から、このJavaＡＰを実行した場合に使用されるデータを、例えば、このJavaＡＰのプログラム内容を解析するなどして特定する（ステップＳ１０３）。なお、JavaＡＰが使用するデータを特定する際には、ＪＡＲストレージ４１０ｂ内の、このJavaＡＰのＪＡＲファイルに格納されているデータは特定の対象から除外する。これは、ＪＡＲファイル内に格納されているデータは、このJavaＡＰを実行する上で必要となるデータとして当該JavaＡＰを提供するコンテンツプロバイダが用意したデータであるためである。

【００７４】
次いで、ＣＰＵ４０５は、型指定テーブル４１０ａ（図３）を参照して上記特定したデータの型指定情報の値に基づいて、このデータを扱うオブジェクトの型を「完全カプセル化」型とするのか、それとも「非完全カプセル化」型とするのかを決定する（ステップＳ１０４）。例えば、JavaＡＰの使用するデータがアドレス帳データの場合、ＣＰＵ４０５は、型指定テーブル４１０ａを参照し、アドレス帳データを扱うオブジェクトの型を「完全カプセル化」型に決定する。また、JavaＡＰの使用するデータがコンテンツＢ（著作権保護フラグ“０”）の場合、ＣＰＵ４０５は、コンテンツＢを扱うオブジェクトの型を「非完全カプセル化」型に決定する。
【００７５】
この後、ＣＰＵ４０５は、上記ステップＳ１０３において特定したデータと、上記ステップＳ１０４において決定したオブジェクトの型とに基づいて、完全カプセル化オブジェクトまたは非完全カプセル化オブジェクトを生成する（ステップＳ１０５）。例えば、上記ステップＳ１０３において特定したデータがアドレス帳データの場合、ＣＰＵ４０５は、オリジナルJava拡張ライブラリ内の完全カプセル化ＡＰＩを起動して、アドレス帳データ用の完全カプセル化オブジェクトを生成する。また、上記ステップＳ１０３において特定したデータがコンテンツＢの場合、ＣＰＵ４０５は、オリジナルJava拡張ライブラリ内の非完全カプセル化ＡＰＩを起動して、コンテンツＢ用の非完全カプセル化オブジェクトを生成する。
【００７６】
次いで、ＣＰＵ４０５は、生成した完全カプセル化オブジェクトまたは非完全カプセル化オブジェクトを共通スクラッチパッド４１０ｄに格納し（ステップＳ１０６）、オブジェクト生成処理を終了する。なお、上記ステップＳ１０５において生成された完全カプセル化オブジェクトや非完全カプセル化オブジェクトは、共通スクラッチパッド４１０ｄではなく、個別スクラッチパッド４１０ｃに格納される形態であってもよい。
【００７７】
また、上記ステップＳ１０３においてJavaＡＰの使用するデータが複数特定された場合は、特定した各データ毎に、当該データ用の完全カプセル化オブジェクトまたは非完全カプセル化オブジェクトを生成して共通スクラッチパッド４１０ｄに格納するため、上記ステップＳ１０４～Ｓ１０６までの処理を各データ毎に繰り返して行う。そして、ＣＰＵ４０５は、オブジェクト生成処理を終了した後、実行するプログラムとして指定されたJavaＡＰを起動し、このプログラムに基づく処理を開始する。
【００７８】
＜２．アクセス管理処理＞
次に、携帯電話機４０においてＣＰＵ４０５により実行されるアクセス管理処理について図９を参照して説明する。このアクセス管理処理は、ＪＡＭの機能としてＣＰＵ４０５により実行されるものであり、ダウンロードされたJavaＡＰの実行過程においてアクセス要求が発生した場合に、割り込み処理として実行される。
【００７９】
同図に示すように、まず、携帯電話機４０のＣＰＵ４０５は、JavaＡＰの実行過程において発生したアクセス要求について、アクセスの要求先が予め許可された範囲内のリソースであるか否かを判別し、アクセス要求を許可するか否かを判定する（ステップＳ２０１）。ここで、アクセス要求の許可有無を判定する仕組みについて具体的に説明すると、ダウンロードされたJavaＡＰが実行される場合、ＣＰＵ４０５は、JavaＡＰの実行に伴ってアクセスすることのできるリソースを、このJavaＡＰのＡＤＦに記述されているＵＲＬにより指定される当該JavaＡＰのダウンロード元のコンテンツサーバ１０（サイト）と、このJavaＡＰに対して割り当てられたＪＡＲストレージ４１０ｂおよび個別スクラッチパッド４１０ｃ内の記憶領域と、共通スクラッチパッド４１０ｄと、のみに制限する。
【００８０】
したがって、ＣＰＵ４０５は、アクセスの要求先が上述したリソースのいずれかである場

合は、このアクセス要求を許可する一方、アクセスの要求先が上述したリソース以外である場合は、このアクセス要求を許可しない。

【００８１】

次いで、ＣＰＵ４０５は、アクセス要求の許可有無を示す判定結果を要求元のJavaＡＰに通知した後（ステップＳ２０２）、アクセス管理処理を終了する。また、実行中のJavaＡＰは、ＪＡＭによる判定結果を受け取ると、この判定結果に従って、アクセス要求が許可された場合は当該アクセス要求に基づく処理を実行する一方、アクセス要求が許可されなかった場合は当該アクセス要求に基づく処理をキャンセルする。

【００８２】

さて、携帯電話機４０のＣＰＵ４０５は、ダウンロードしたJavaＡＰを実行する場合、図８に示したオブジェクト生成処理を行った後にJavaＡＰを起動する。また、ダウンロードしたJavaＡＰの実行過程においてＣＰＵ４０５は、アクセス要求が発生すると、図９に示したアクセス管理処理を行なう。したがって、携帯電話機４０は、ダウンロードしたJavaＡＰの実行中において必ずＪＡＭによるアクセス制限を受けることとなり、例えば、不揮発性メモリ４１０に格納されているアドレス帳データ、電子メールデータ、着信・発信履歴データ、ユーザデータ、コンテンツなどのデータそのものにアクセスすることができなくなる。

【００８３】

このため、携帯電話機４０のＣＰＵ４０５は、上述したオブジェクト生成処理において、起動させるJavaＡＰが使用するデータを特定し、当該データ用の完全カプセル化オブジェクトまたは非完全カプセル化オブジェクトを生成して共通スクラッチパッド４１０ｄに格納する。この共通スクラッチパッド４１０ｄは、前述したように、ＪＡＭによるアクセス制限が行われている場合であっても携帯電話機４０のアクセスが許可されるリソースである。また、携帯電話機４０にダウンロードされるJavaＡＰは、共通スクラッチパッド４１０ｄに格納された完全カプセル化オブジェクトや非完全カプセル化オブジェクトにアクセスし、当該オブジェクトに備わるメソッドを使用してこのオブジェクト内のデータに対する操作を指令するように作成されている。

【００８４】

例えば、アドレス帳データを使用するJavaＡＰが起動される場合、上述したオブジェクト生成処理によりアドレス帳データ用の完全カプセル化オブジェクトが生成され、共通スクラッチパッド４１０ｄに格納される。また、このJavaＡＰは、上記生成されたアドレス帳データ用の完全カプセル化オブジェクトに対して、当該オブジェクトに備わるメソッドを用いてこのオブジェクト内のデータに対する操作を指令する。したがって、完全カプセル化オブジェクトの有するアドレス帳データの一部を画面表示させることなどが可能となる一方、完全カプセル化オブジェクトの有するデータそのものがJavaＡＰに引き渡されることはない。

【００８５】

従来は、ダウンロードされたJavaＡＰに対するセキュリティを確保するため、このようなJavaＡＰについては、アドレス帳データ、電子メールデータ、着信・発信履歴データ、ユーザデータなどに一切アクセスすることができなかった。これに対して本実施形態によれば、完全カプセル化オブジェクトを用いることにより、データそのものがJavaＡＰに引き渡されることがないので、ダウンロードされたJavaＡＰに対するセキュリティを確保しつつ、同時に、従来は一切アクセスできなかったデータについて、完全カプセル化オブジェクトを介して画面表示を行わせることなどができるようになる。したがって、ダウンロードされたJavaＡＰが携帯電話機４０において実現することのできる機能を充実させることができる。

【００８６】

また、著作権が放棄されたコンテンツや、セキュリティ上の重要度が低く、型指定情報として“０”がセットされているデータを使用するJavaＡＰが起動される場合、非完全カプセル化オブジェクトが生成されて共通スクラッチパッド４１０ｄに格納される。この場合

は、完全カプセル化オブジェクトの場合と異なり、非完全カプセル化オブジェクトの有しているデータそのものをJavaＡＰに引き渡すこともできる。

【００８７】

すなわち、ダウンロードされたJavaＡＰは信頼性を完全に保証することのできないプログラムであるが、著作権が放棄されたデータやセキュリティ上の重要度が低いデータについては、非完全カプセル化オブジェクトとして扱うことで、JavaＡＰにデータそのものを引き渡せるようにする。JavaＡＰにデータそのものを引き渡せるようにした方が利便性を高くできることは説明をするまでもなく明かであり、このようにカプセル化するデータのセキュリティ上の重要度や著作権の保護の要否などに応じて完全カプセル化オブジェクトと非完全カプセル化オブジェクトとを使い分けるようにすると、完全カプセル化オブジェクトのみを用いた場合と比較して、さらに利便性を高めることができる。

【００８８】

＜３．JavaＡＰ終了処理＞

次に、携帯電話機４０においてＣＰＵ４０５により実行されるJavaＡＰ終了処理について図１０を参照して説明する。このJavaＡＰ終了処理は、ＪＡＭの機能としてＣＰＵ４０５により実行されるものであり、JavaＡＰの実行終了要求が発生した場合に、割り込み処理として実行される。

【００８９】

同図に示すように、携帯電話機４０のＣＰＵ４０５は、JavaＡＰの実行終了要求が発生すると、共通スクラッチパッド４１０ｄに格納されている完全カプセル化オブジェクトや非完全カプセル化オブジェクトを削除する（ステップＳ３０１）。このステップＳ３０１において削除される完全カプセル化オブジェクトや非完全カプセル化オブジェクトは、JavaＡＰを起動させる際に、上述したオブジェクト生成処理（図８参照）において生成され、共通スクラッチパッド４１０ｄに格納されたものである。ＣＰＵ４０５は、共通スクラッチパッド４１０ｄからオブジェクトを削除すると、JavaＡＰ終了処理を終える。

【００９０】

このようにダウンロードされたJavaＡＰを起動する際に、完全カプセル化オブジェクトや非完全カプセル化オブジェクトを生成して共通スクラッチパッド４１０ｄに格納する一方、このJavaＡＰの実行が終了する際に、共通スクラッチパッド４１０ｄから完全カプセル化オブジェクトや非完全カプセル化オブジェクトを削除するようにすると、完全カプセル化オブジェクトや非完全カプセル化オブジェクトを共通スクラッチパッド４１０ｄに常時保持しておく必要がない。したがって、携帯電話機４０のメモリ資源を効率的に活用することができる。

【００９１】

なお、携帯電話機４０は、ＲＯＭ４０８または不揮発性メモリ４１０に記憶されているプログラムに従って、本発明に係る処理（オブジェクト生成処理、アクセス管理処理、JavaＡＰ終了処理）を実行するが、このような処理を実行するためのプログラムを携帯電話機４０に対して通信により提供する形態としてもよい。さらに、このような処理を実行するためのプログラムを、例えば、光記録媒体や磁気記録媒体、半導体メモリなどの記録媒体を用いて携帯電話機４０へ提供するようにしてもよい。但し、プログラムを記録媒体により携帯電話機４０へ提供する場合、携帯電話機４０は、記録媒体からプログラムを読み出すための記録媒体ドライブを有する。

【００９２】

［Ｂ．変形例］

以上、本発明の実施形態について説明したが、本発明はその主要な特徴から逸脱することなく他の様々な形態で実施することが可能である。上述した実施形態は、本発明の一態様を例示したものに過ぎず、本発明の範囲は、特許請求の範囲に示す通りであって、また、特許請求の範囲の均等範囲に属する変形や変更は、全て本発明の範囲内に含まれる。なお、変形例としては、例えば、以下のようなものが考えられる。

【００９３】

＜変形例１＞
上述した実施形態では、完全カプセル化オブジェクトと非完全カプセル化オブジェクトとを用いる場合について説明したが、完全カプセル化オブジェクトのみを用いるようにしてもよい。すなわち、ダウンロードされたJavaＡＰが使用するデータを全て完全カプセル化オブジェクトとして扱うようにしてもよい。この場合、型指定情報は不要となる。また、上述した実施形態では、型指定テーブル４１０ａを用いる構成としたが、アドレス帳データ、電子メールデータ、コンテンツなどのデータ自体に型指定情報を付与するようにすれば、指定テーブル４１０ａを用いる必要はない。

【００９４】
＜変形例２＞
上述した実施形態では、ダウンロードされたJavaＡＰの実行が指示された場合に、完全カプセル化オブジェクトまたは非完全カプセル化オブジェクトを生成するようにした。しかしながら、完全カプセル化オブジェクトまたは非完全カプセル化オブジェクトが生成されるタイミングは、JavaＡＰの実行が指示されたときに限定されるものではない。

【００９５】
例えば、携帯電話機４０の電源投入時などに、型指定テーブル４１０ａ（図３）を参照し、各データ用の完全カプセル化オブジェクトや非完全カプセル化オブジェクトを生成して共通スクラッチパッド４１０ｄに格納しておくようにしてもよい。但し、この場合、アドレス帳データ、電子メールデータ、着信・発信履歴データ、ユーザデータなどの元データが更新された場合に、その更新内容に応じて共通スクラッチパッド４１０ｄに格納されている完全カプセル化オブジェクトや非完全カプセル化オブジェクト内のデータを更新する必要がある。

【００９６】
したがって、このような制御を行う場合には、元データの更新に応じてオブジェクト内にカプセル化されているデータを更新するＡＰＩをオリジナル拡張ライブラリに備えるようにする。また、完全カプセル化オブジェクトや非完全カプセル化オブジェクトを共通スクラッチパッド４１０ｄに常時保持しておく場合は、当然、JavaＡＰの実行終了に応じて共通スクラッチパッド４１０ｄから完全カプセル化オブジェクトや非完全カプセル化オブジェクトを削除する必要はない。

【００９７】
＜変形例３＞
上述した実施形態において、例えば、“１（重要度「高」）”～“５（重要度「低」）”までのセキュリティ上の重要度を示すセキュリティレベル情報をデータに対して付与し、このセキュリティレベル情報が“１”～“３”までのデータを完全カプセル化オブジェクトとして扱う一方、セキュリティレベル情報が“４”および“５”のデータを非完全カプセル化オブジェクトとして扱うようにしてもよい。

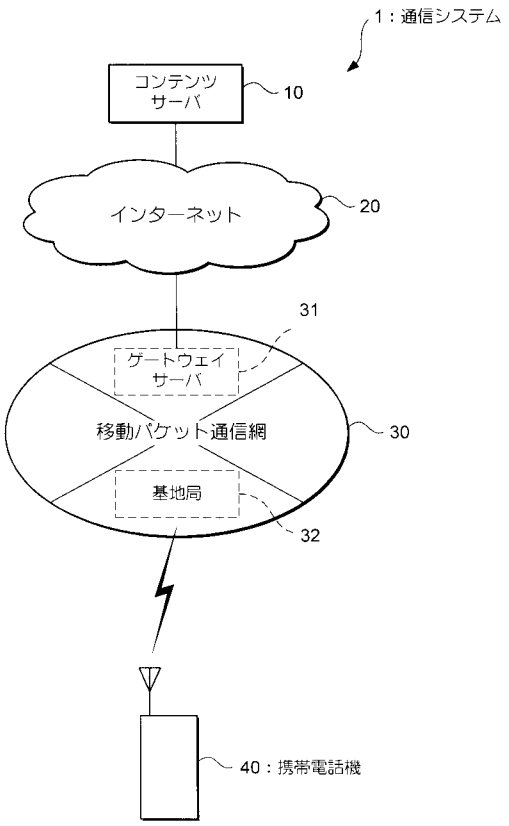
【００９８】
＜変形例４＞
携帯電話機４０にダウンロードされたJavaＡＰであっても、例えば、移動パケット通信網３０を運営する通信事業者やＣＡ（Certificate Authority）ような公正な第３者機関によりJavaＡＰの内容が審査され、一定の動作基準を満たしていると認定されたJavaＡＰであれば、ネイティブアプリケーションと同様に、信頼性を完全に保証することのできるプログラムとみなすことができる。

【００９９】
したがって、第３者機関によって認定されたJavaＡＰであれば、ダウンロードされたJavaＡＰであってもネイティブアプリケーションと同様に、使用するデータを不揮発性メモリ４１０から直接取得できるようにしてもよい。なお、第３者機関により認定されたJavaＡＰには、当該JavaＡＰが第３者機関の認定プログラムであることを示す識別情報が付与されている。したがって、携帯電話機４０において、ダウンロードされたJavaＡＰが第３者機関による認定プログラムであるか否かを識別する場合には、上記識別情報の有無を判別

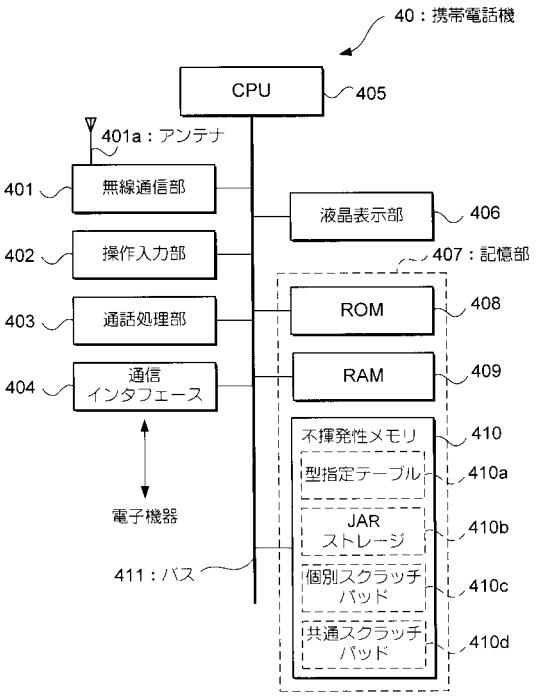
すればよい。
【０１００】
＜変形例５＞
上述した実施形態においてコンテンツサーバ１０は、インターネット２０に接続されている構成とした。しかしながら、コンテンツサーバ１０は、専用線を介して移動パケット通信網３０のゲートウェイサーバ３１に直接接続されている構成であってもよい。また、ゲートウェイサーバ３１がコンテンツサーバ１０の機能を有する構成であってもよい。さらに、コンテンツサーバ１０が移動パケット通信網３０内に設置されている構成であってもよい。
【０１０１】
＜変形例６＞
上述した実施形態では、図１１においてハッチングで示すように、ＫＶＭと、コンフィギレーションとしてＣＬＤＣを備えるとともにプロファイルとしてオリジナルJava拡張プロファイルを備えるＪ２ＭＥとが組み込まれた携帯電話機４０に本発明を適用した場合について説明した。しかしながら、Java実行環境は、上述したＫＶＭとＪ２ＭＥの組み合わせに限定されるものではない。また、本発明が適用可能な通信装置は、携帯電話機に限定されるものではない。
【０１０２】
例えば、同図に示すように、Ｊ２ＭＥのプロファイルとして、オリジナルJava拡張プロファイルの代わりにＭＩＤＰ（Mobile Information Device Profile）を有する構成であってもよい。また、ＫＶＭの代わりにＪＶＭを有し、Ｊ２ＭＥのコンフィギレーションとしてＣＬＤＣの代わりにＣＤＣ（Connected Device Configuration）を、また、Ｊ２ＭＥのプロファイルとして、例えば、液晶付電話機用プロファイル、ＴＶ用プロファイル、カーナビゲーション用プロファイルなどを有する構成であってもよい。さらには、ＨｏｔＳｐｏｔと、Ｊ２ＳＥ（Java 2 Standard Edition）またはＪ２ＥＥ（Java 2 Enterprise Edition）とを有する構成であってもよい。
【０１０３】
また、以上説明したJava実行環境の変形例から明らかなように、本発明は、例えば、ＰＨＳ（Personal Handyphone System：登録商標）端末やＰＤＡ（Personal Digital Assistant）、カーナビゲーション装置、パーソナルコンピュータなどの、通信機能を有する各種電子機器に適用可能である。また、本発明は、移動パケット通信網３０に収容される通信装置に限定されるものではない。例えば、図１２に示すような通信システム２において、ＬＡＮ５０内に設けられたパーソナルコンピュータ７０Ａ～７０Ｃに本発明を適用することもできる。
【０１０４】
また、上述した実施形態では、Javaプログラミング言語により記述されたJavaＡＰを用いた場合について説明したが、プログラミング言語はJavaに限定されるものではない。
【０１０５】
【発明の効果】
以上説明したように本発明によれば、受信したプログラムに対する通信装置のセキュリティを確保しつつ、このようなプログラムが通信装置において実現することのできる機能を従来と比較して充実させることができる。
【図面の簡単な説明】
【図１】　本発明の実施形態に係る通信システムの構成を例示するブロック図である。
【図２】　同実施形態に係る携帯電話機のハードウェア構成を例示するブロック図である。
【図３】　同実施形態に係る携帯電話機において、不揮発性メモリに格納されている型指定テーブルのデータ構成を例示する図である。
【図４】　同実施形態に係る携帯電話機において、JavaＡＰの実行環境を説明するための図である。

【図５】　同実施形態に係る携帯電話機において、カプセル化オブジェクトを説明するための模式図である。
【図６】　同実施形態に係る携帯電話機において、非完全カプセル化オブジェクトについて例示する模式図である。
【図７】　同実施形態に係る携帯電話機において、完全カプセル化オブジェクトについて例示する模式図である。
【図８】　同実施形態に係る携帯電話機において、ＣＰＵにより実行されるオブジェクト生成処理の動作を説明するフローチャートである。
【図９】　同実施形態に係る携帯電話機において、ＣＰＵにより実行されるアクセス管理処理の動作を説明するフローチャートである。
【図１０】　同実施形態に係る携帯電話機において、ＣＰＵにより実行されるJavaＡＰ終了処理の動作を説明するフローチャートである。
【図１１】　本発明の変形例６に係り、Java実行環境の変形例を説明するための図である。
【図１２】　本発明の変形例６に係り、通信システムの変形例を例示するブロック図である。
【符号の説明】
１，２……通信システム、１０……コンテンツサーバ、２０……インターネット、３０……移動パケット通信網、３１……ゲートウェイサーバ、３２……基地局、４０……携帯電話機、５０……ＬＡＮ、６０……ゲートウェイサーバ、７０Ａ，７０Ｂ，７０Ｃ……パーソナルコンピュータ、４０１……無線通信部、４０１ａ……アンテナ、４０２……操作入力部、４０３……通話処理部、４０４……通信インタフェース、４０５……ＣＰＵ、４０６……液晶表示部、４０７……記憶部、４０８……ＲＯＭ、４０９……ＲＡＭ、４１０……不揮発性メモリ、４１０ａ……型指定テーブル、４１０ｂ……ＪＡＲストレージ、４１０ｃ……個別スクラッチパッド、４１０ｄ……共通スクラッチパッド、４１１……バス。

【図１】



【図２】

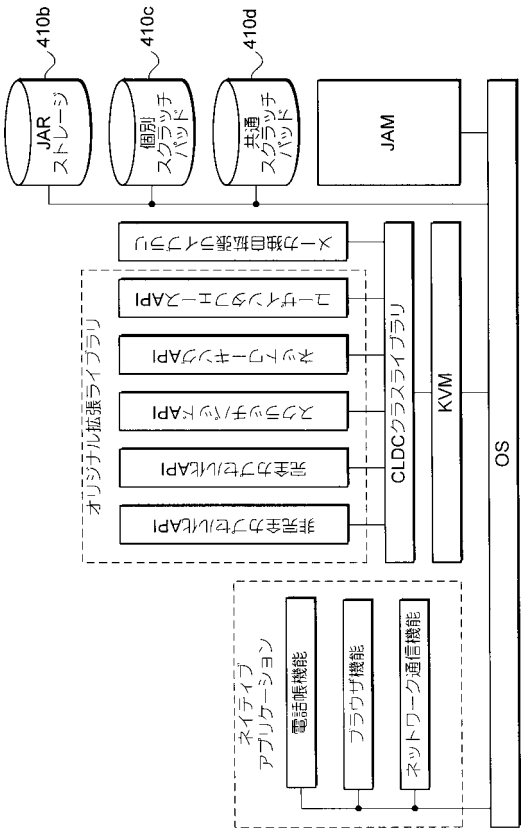


【図３】

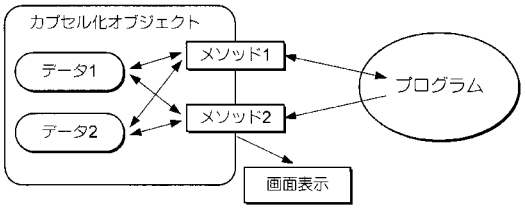
410a：型指定テーブル

データ名	型指定情報
アドレス帳データ	1 （完全カプセル化型）
電子メールデータ	1
着信・発信履歴データ	1
ユーザデータ	1
コンテンツA （著作権保護フラグ“1”）	1
コンテンツB （著作権保護フラグ“0”）	0 （非完全カプセル化型）
自作画像データ	0
⋮	⋮

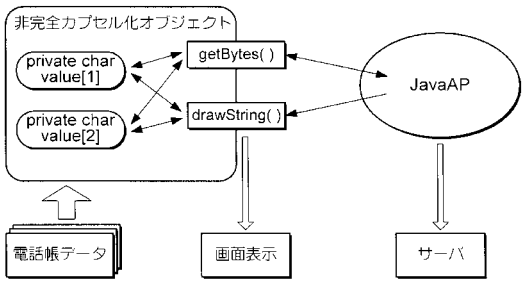
【図４】



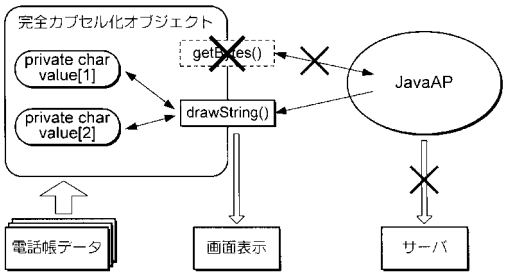
【図５】



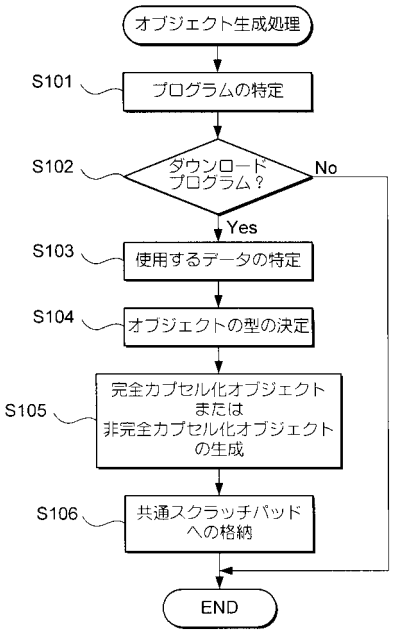
【図６】



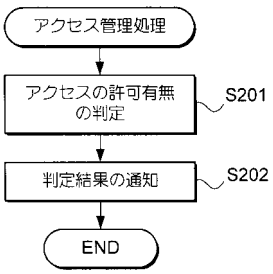
【図７】



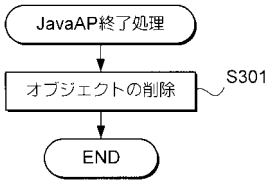
【図８】



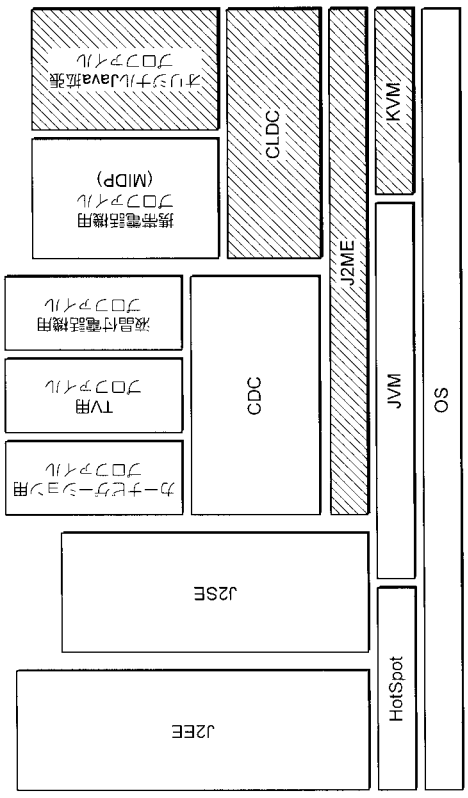
【図９】



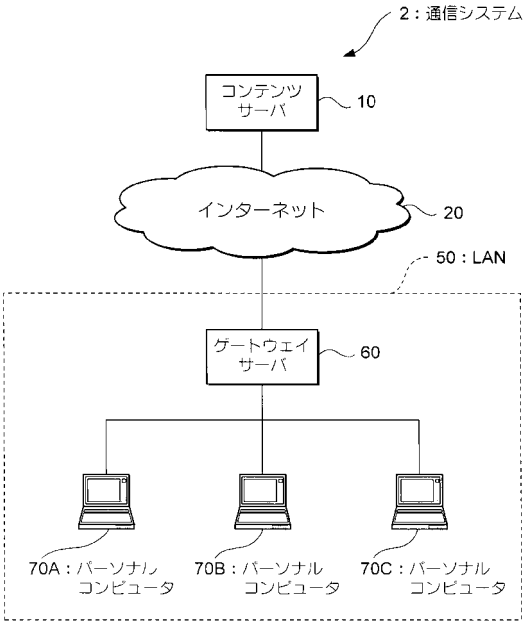
【図１０】



【図１１】



【図１２】



フロントページの続き

(72)発明者　鷲見　豊
東京都港区虎ノ門４－３－２０　神谷町ＭＴビル２Ｆ　株式会社ケイ・ラボラトリー内

審査官　高橋　克

(56)参考文献　特開２００１－３５０６６４（ＪＰ，Ａ）
特開２００１－０４３１７６（ＪＰ，Ａ）
（株）コネクト 加来 徹也，山田 昌宏，伊藤 広明，“はじめてのｉモードＪａｖａプログラミング”，日経ＢＰ社，２００１年　4月２５日，1版，ｐ．３０－４１
Scott Oaks 著，島田 秋雄 監訳，“Ｊａｖａセキュリティ”，株式会社オライリー・ジャパン，２００１年１１月２８日，初版，ｐ．１９－３３，６９－９６

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F　21/00　- 21/24
G06F　9/44
H04B　7/26
H04Q　7/38
H04M　1/00
H04M　1/725