(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2007-34813
(P2007-34813A)
(43)公開日　平成19年2月8日(2007.2.8)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 9/44	(2006.01)	G06F	9/06	620A	5B009
G06F 17/21	(2006.01)	G06F	17/21	592A	5B076
		G06F	17/21	501T	5B176

審査請求 未請求　請求項の数 12　ＯＬ　（全 22 頁）

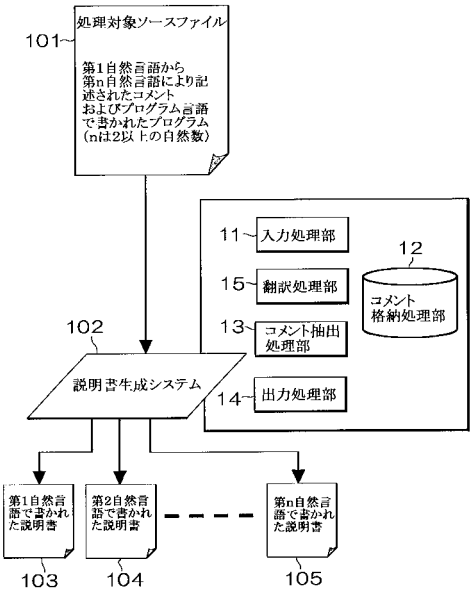
(21)出願番号　特願2005-218993 (P2005-218993)
(22)出願日　平成17年7月28日 (2005.7.28)

(71)出願人　301021533
独立行政法人産業技術総合研究所
東京都千代田区霞が関１－３－１
(72)発明者　平野　聡
茨城県つくば市東１－１－１　独立行政法人産業技術総合研究所つくばセンター内
(72)発明者　大川　猛
茨城県つくば市東１－１－１　独立行政法人産業技術総合研究所つくばセンター内
(72)発明者　曲　潤涛
茨城県つくば市東１－１－１　独立行政法人産業技術総合研究所つくばセンター内

Ｆターム(参考)　5B009 QB02　TB01　VB01
5B076 DD04　DF01　DF02　DF04
5B176 DD04　DF01　DF02　DF04

(54)【発明の名称】複数自然言語のソフトウェア説明書生成システム

(57)【要約】
【課題】　複数の自然言語で書かれたソフトウェア説明書を生成できるソフトウェア説明書生成システムを提供する。
【解決手段】　ソフトウェア説明書生成システムは、入力手段により、プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内においてソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述され、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号とを組み合わせた符号が付加されているソースファイルを入力し、ソースファイルを解釈し、組み合わせた符号を識別してソースコード文と対応づけてコメントをメモリ上に格納し、使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントのみを抽出し、抽出されたコメントに基づいて前記ソースコード文に対する出力対象の自然言語のソフトウェア説明書を出力する。
【選択図】　図１



【特許請求の範囲】
【請求項１】
プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内においてソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述されており、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号とを組み合わせた符号が付加されているソースファイルを入力する入力手段と、
入力されたソースファイルを解釈し、組み合わせた符号を識別してソースコード文と対応づけてコメントをメモリ上に格納する格納手段と、
使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントのみを抽出する抽出手段と、
前記抽出されたコメントに基づいて前記ソースコード文に対する出力対象の自然言語のソフトウェア説明書を出力する出力手段と、
を備えることを特徴とするソフトウェア説明書生成システム。
【請求項２】
プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内においてソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述されており、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号と更に国もしくは地域を示す符号とを組み合わせた符号が付加されているソースファイルを入力する入力手段と、
入力されたソースファイルを解釈し、組み合わせた符号を識別してソースコード文と対応づけてコメントをメモリ上に格納する格納手段と、
使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントのみを抽出する抽出手段と、
前記抽出されたコメントに基づいて前記ソースコード文に対する出力対象の自然言語のソフトウェア説明書を出力する出力手段と、
を備えることを特徴とするソフトウェア説明書生成システム。
【請求項３】
請求項１に記載のソフトウェア説明書生成システムにおいて、更に、
１つの自然言語の文を他の自然言語の文に翻訳処理する翻訳処理手段を備え、
前記抽出手段が、使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントが存在しない場合に、所定の自然言語の種類に相当する符号がつけられたコメントを前記ソースファイルから抽出し、
前記出力手段は、出力対象の自然言語のコメントを、翻訳処理手段により所定の自然言語で記述されたコメントを機械翻訳し、使用者によって指定された出力対象の自然言語のソフトウェア説明書を出力する、
ことを特徴とするソフトウェア説明書生成システム。
【請求項４】
請求項３に記載のソフトウェア説明書生成システムにおいて、
機械翻訳の際に訳元として用いるコメントの自然言語の種類をデフォルトで示すための、主たる自然言語の種類を示す符号をソースファイル中に含む
ことを特徴とするソフトウェア説明書生成システム。
【請求項５】
請求項１に記載のソフトウェア説明書生成システムにおいて、
コメントに付ける符号には、コメントが更新を必要とする旨の符号を含み、
前記出力手段が、コメントが更新を必要とする旨の符号に基づいて、ソースファイルにおいて更新が必要な箇所もしくは更新が必要な言語についての情報を出力する
ことを特徴とするソフトウェア説明書生成システム。
【請求項６】
複数の自然言語のソフトウェア説明書を出力するソフトウェア説明書生成システムを構

成するためのプログラムであり、
コンピュータを、
プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内においてソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述されており、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号とを組み合わせた符号が付加されているソースファイルを入力する入力手段と、
入力されたソースファイルを解釈し、組み合わせた符号を識別してソースコード文と対応づけてコメントをメモリ上に格納する格納手段と、
使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントのみを抽出する抽出手段と、
前記抽出されたコメントに基づいて前記ソースコード文に対する出力対象の自然言語のソフトウェア説明書を出力する出力手段と、
して機能させるためのプログラム。
【請求項７】
複数の自然言語のソフトウェア説明書を出力するソフトウェア説明書生成システムを構成するためのプログラムであり、
コンピュータを、
プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内においてソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述されており、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号と更に国もしくは地域を示す符号とを組み合わせた符号が付加されているソースファイルを入力する入力手段と、
入力されたソースファイルを解釈し、組み合わせた符号を識別してソースコード文と対応づけてコメントをメモリ上に格納する格納手段と、
使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントのみを抽出する抽出手段と、
前記抽出されたコメントに基づいて前記ソースコード文に対する出力対象の自然言語のソフトウェア説明書を出力する出力手段と、
して機能させるためのプログラム。
【請求項８】
請求項６に記載のプログラムにおいて、
コンピュータを、更に、
１つの自然言語の文を他の自然言語の文に翻訳処理する翻訳処理手段として機能させるサブプログラムを備え、
前記抽出手段として機能するサブプログラムは、コンピュータを、使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントが存在しない場合に、所定の自然言語の種類に相当する符号がつけられたコメントを前記ソースファイルから抽出する手段として機能させ、
前記出力手段として機能するサブプログラムは、出力対象の自然言語のコメントを、翻訳処理手段により所定の自然言語で記述されたコメントを機械翻訳し、使用者によって指定された出力対象の自然言語のソフトウェア説明書を出力する手段として機能させる、
ことを特徴とするプログラム。
【請求項９】
請求項９に記載のプログラムにおいて、
機械翻訳の際に訳元として用いるコメントの自然言語の種類をデフォルトで示すための、主たる自然言語の種類を示す符号をソースファイル中に含む
ことを特徴とするプログラム。
【請求項１０】
請求項６に記載のプログラムにおいて、

コメントに付ける符号には、コメントが更新を必要とする旨の符号を含み、
前記出力手段として機能するサブプログラムは、コンピュータを、コメントが更新を必要とする旨の符号に基づいて、ソースファイルにおいて更新が必要な箇所もしくは更新が必要な言語についての情報を出力する手段として機能させる
ことを特徴とするプログラム。
【請求項１１】
プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内において、ソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述されており、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号とを組み合わせた符号が付加されていることを特徴とするソースファイルのデータ構造。
【請求項１２】
プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内において、ソースコード中のあるひとつの機能を説明するコメントが、機能を表す符号に続いて複数の自然言語で記述されており、前記コメントには、記述に用いられている自然言語の種類を示す符号が付加されていることを特徴とするソースファイルのデータ構造。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、複数の自然言語のソフトウェア説明書を出力することのできるソフトウェア説明書生成システムに関するものであり、更に、詳しくは、コメントが含まれているコンピュータソフトウェアのソースファイルからソフトウェアに関する説明書をテキスト処理により生成し、また、テキスト処理によりファイルを変換するソフトウェア説明書生成システムに関するものである。
【背景技術】
【０００２】
本発明の説明に先立ち、誤解を招きやすいと思われる言葉の定義を説明する。本発明においては、コンピュータ・システム上で幾つかの種類の「言語」を取り扱う。このため、本明細書において、「自然言語」は、日本語、英語、中国語、韓国語など、人類が普段使っている言語のことを意味する。また、「プログラミング言語」は、アセンブリ言語，Ｃ言語，Ｊａｖａ（登録商標）言語など、情報機器を動かすソフトウェアを記述するための言語一般を意味する。
【０００３】
今日、ソフトウェア製品を多くの国や地域に出荷することはソフトウェア企業にとって重要な戦略である。一般にソフトウェアを使用する際には、何らかの説明書を読んで理解を深める。その際、読者にとっての第一自然言語（すなわち母語）で書かれた説明書を読むことにより、最も効率的にソフトウェアの使用法を習得することが出来る。すなわち、ソフトウェアの使いやすさは、ソフトウェアに関連する説明書の読みやすさに大きく依存する。その為、ソフトウェアに関連する説明書、より一般的にはソフトウェアに関連するドキュメントをそれぞれの国や地域の現地語で提供することが、各出荷地域においてのソフトウェアの価値を高めることにつながる。
【０００４】
一方、ソフトウェア開発の国際化もまた急速に進んでいる。プログラミング言語それ自体は自然言語からはほぼ独立であり、世界のソフトウェア開発者が共有する知識体系であるため、開発の国際化は自然の流れであるといえる。しかし、ソフトウェアが複雑化した今日において、ソフトウェアのソースコードだけでは、そのソフトウェアに関する理解は非常に困難である。そのため、ソフトウェア開発に関する自然言語で書かれたドキュメント（例：○×ソフトウェア説明書、○×プログラム内部仕様書、など）をソースコードと

併せて、開発者間で共有もしくは配布するのが一般的である。ドキュメントによって、そのソフトウェアの仕様についての理解を促進することが可能となる。そこで問題となるのは、どの自然言語でその開発ドキュメントが書かれるかである。

【０００５】

一般に、開発に関するドキュメントは、開発地の第一自然言語、もしくは事実上の国際的標準自然言語となっている英語で記述されることが多い。しかし、十分な語学能力と十分なソフトウェア開発能力を同時に併せ持つ人材は稀である。その為、読者（すなわちソフトウェア開発者）にとっての母語で記述されたソフトウェア開発用ドキュメントを読むことで開発対象のソフトウェアに対する理解を深めることが、開発期間の大幅な短縮につながる。そのため、ソフトウェア製品（もしくはソフトウェアコンポーネント製品）は世界的な共通自然言語である英語でのドキュメントのみならず、現地の自然言語で書かれたドキュメントを備えた上で出荷されることが望ましい。その結果、世界の各地域におけるソフトウェア製品の価値を高めることにつながる。

【０００６】

そのため、ソフトウェアの開発側としてはいくつかの自然言語で書かれたドキュメントを作成する必要がある。しかし、ソフトウェア開発にかかわるドキュメント作成は非常に時間と労力のかかるものである。特に多数の自然言語で発行する必要がある場合には、発行の度に翻訳工程、および各自然言語で書かれたドキュメント間の整合性の確認工程が必要となり、ソフトウェア製品の生産性向上の障害となっていた。

【０００７】

これは、ソフトウェア開発ドキュメントの作成工程を、ソフトウェア自身と関連させて考えてみると理解が容易である。一般に、ソフトウェアのソースコードと、ソフトウェア開発ドキュメント（内部仕様書）を別個に作成・維持することは、コストのかかる工程である。何故なら、内部仕様書はソフトウェアのソースコードと非常に密接に関連するドキュメントであるため、ソースコードの変更時に、ソースコードとドキュメントの内容が一致するように保つことが困難である。

【０００８】

そのため、ソフトウェアのソースコード中に注釈（コメント）をあわせて書き込むことにより、動作するソフトウェアとドキュメントを一体として作り上げる開発スタイルが、ソフトウェア製品の生産性向上に有用であるとの認識が広まっている。例えば、Ｋｎｕｔｈは、文献「文芸的プログラミング（Ｌｉｔｅｒａｔｅ　Ｐｒｏｇｒａｍｍｉｎｇ）」において、文章の中にソフトウェアのソースコードを織り込んで、自然言語での説明文とともに書かれたプログラミング言語でのソフトウェアを示し、その有用性を示している。このような開発スタイルにおいて、ソースコード中のコメントは、コメント抽出・文書整形ソフトウェアにより自動的に抽出・整形されて、完成品のドキュメントとして即座に開発時もしくは運用時に利用可能である。

【０００９】

ソフトウェアおよびドキュメント双方の品質維持とコスト削減の観点から見て、ドキュメント自動生成システムの採用は非常に有効であるといえる。更には、即座に利用できることから、開発効率の向上に有用である。また、実行されるソフトウェアとドキュメントとの間の整合性の確保を行いやすい利点がある。

【００１０】

サンマイクロシステムズ社のＪａｖａｄｏｃは、その一例である。Ｊａｖａ（登録商標）のプログラムソースコード中に、クラスやメソッドといったソフトウェアの構成単位に対して、説明をコメント（注釈）として書き込み、ドキュメントをＨＴＭＬ文書やＰＤＦ文書として出力を行う。説明を書き込む際には、その説明の意味を示す符号をあわせて書き込むことにより、出力するドキュメント中のしかるべき場所にその説明が現れるような制御が可能である。

【００１１】

このようなドキュメントの自動生成は、旧来、コメントが単一の自然言語で書かれたソ

ースファイルにおいて行われてきた。何故なら、コンピュータの成立の歴史や今日の国際的な標準自然言語が英語であるという背景から、ソフトウェアは英語の文字セットを用いたプログラミング言語で記述することが一般的であり、そのソースファイルに付されるコメントも英語でかかれることが多いからである。

【００１２】

ソフトウェアソースコードからのドキュメント自動生成システムとして、Ｊａｖａｄｏｃと同様のドキュメント生成を行うオープンな実装のうち著名なものとしては、ＤｏｘｙｇｅｎやＫＤＯＣ、ＤＯＣ＋＋が挙げられる。しかし、ここに挙げたツールは、単一の自然言語で書かれたソフトウェア説明書を生成することを目的としたものである。

【００１３】

複数の種類の外国語で書かれた電子ドキュメント用のコンテンツフィルタの技術としては、特許文献１が公知である。このコンテンツフィルタの技術は、主に時事的な内容のニュース記事を話題分野別に分類することを目的としている。

【特許文献１】米国特許第６，５４２，８８８号明細書

【発明の開示】

【発明が解決しようとする課題】

【００１４】

従来、複数の自然言語のソフトウェア説明書を作成する技術として、特許文献１に記載されているように、複数の種類の外国語で書かれた電子ドキュメント用のコンテンツフィルタの技術がある。しかし、この技術は、主に時事的な内容のニュース記事を話題分野別に分類するのが目的であり、複数の自然言語で書かれた文章の分類・抽出についての方法を明示的・具体的に示しているものではない。このため、ソフトウェアのソースファイルに適用して、複数の自然言語のソフトウェア説明書を作成するシステムとすることの示唆は与えられない。

【００１５】

また、既存の技術であるテキストプリプロセッシング（事前処理）システムを用いる方法も考えられる。具体的に、文書プリプロセッシングシステムとは、例えば、Ｃ言語用のプリプロセッサである。想定される方法としては、例えば、生成元のソースファイル中にプリプロセッサ用の命令を埋め込んでおき、ドキュメント自動生成システムに入力する前にプリプロセスを行うことで、生成対象の自然言語以外で書かれたコメントを除去し、対象自然言語で記述されたソフトウェア説明書を作成するような構成とする。Ｃ言語プリプロセッサ用の命令の例としては、＃ｉｆｄｅｆ、＃ｅｎｄｉｆなどがあり、これらを駆使すれば目的を達成することはできそうである。しかし、記述が煩雑であることと、本来ソフトウェアのソースコードの記述用に使用されるものであることから、特に言語を識別する記号の管理において混乱を生じやすく、複数の種類の自然言語のコメントを識別する目的には不適切である。

【００１６】

現在のところ、複数の自然言語で書かれたソフトウェア説明書の生成に関して、生産性の向上に有用な技術は確立されていない。したがって、本発明の目的は、複数の自然言語で書かれたソフトウェア説明書を生成することのできる複数自然言語のソフトウェア説明書生成システムを提供することにある。

【課題を解決するための手段】

【００１７】

上記のような目的を達成するため、本発明による複数自然言語のソフトウェア説明書生成システムは、第１の態様として、プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内においてソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述されており、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号とを組み合わせた符号が付加されているソースファイルを入力する入力手段と、入力されたソースファイルを解釈し、組み合わせた符号を識別してソースコード文と対応づけてコメントをメモリ

上に格納する格納手段と、使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントのみを抽出する抽出手段と、前記抽出されたコメントに基づいて前記ソースコード文に対する出力対象の自然言語のソフトウェア説明書を出力する出力手段とを備えることを特徴とするものである。

【００１８】

また、第２の態様として、本発明による複数自然言語のソフトウェア説明書生成システムは、プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内においてソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述されており、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号と更に国もしくは地域を示す符号とを組み合わせた符号が付加されているソースファイルを入力する入力手段と、入力されたソースファイルを解釈し、組み合わせた符号を識別してソースコード文と対応づけてコメントをメモリ上に格納する格納手段と、使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントのみを抽出する抽出手段と、前記抽出されたコメントに基づいて前記ソースコード文に対する出力対象の自然言語のソフトウェア説明書を出力する出力手段とを備えることを特徴とするものである。

【００１９】

この場合に、ソフトウェア説明書生成システムは、更に、１つの自然言語の文を他の自然言語の文に翻訳処理する翻訳処理手段を備え、前記抽出手段が、使用者によって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントが存在しない場合に、所定の自然言語の種類に相当する符号がつけられたコメントを前記ソースファイルから抽出し、前記出力手段は、出力対象の自然言語のコメントを、翻訳処理手段により所定の自然言語で記述されたコメントを機械翻訳し、使用者によって指定された出力対象の自然言語のソフトウェア説明書を出力するように構成される。

【００２０】

また、この場合に、機械翻訳の際に訳元として用いるコメントの自然言語の種類をデフォルトで示すための、主たる自然言語の種類を示す符号をソースファイル中に含むように構成されてもよい。

【００２１】

また、本発明によるソフトウェア説明書生成システムにおいては、コメントに付ける符号には、コメントが更新を必要とする旨の符号を含み、前記出力手段が、コメントが更新を必要とする旨の符号に基づいて、ソースファイルにおいて更新が必要な箇所もしくは更新が必要な言語についての情報を出力するように構成されても良い。本発明の別の態様では、それぞれの処理要素（手段）がプログラムとして実現される。このプログラムが情報処理装置にインストールされることにより、本発明のソフトウェア説明書生成システムとして機能する。この場合において、システムで用いられるソースファイル構造を構成するためのデータ構造に特徴を有するものとなっている。その１つのソースファイルのデータ構造は、プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内において、ソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述されており、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号とを組み合わせた符号が付加されているデータ構造であり、他の１つのソースファイルのデータ構造は、プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内において、ソースコード中のあるひとつの機能を説明するコメントが、機能を表す符号に続いて複数の自然言語で記述されており、前記コメントには、記述に用いられている自然言語の種類を示す符号が付加されているデータ構造である。

【発明の効果】

【００２２】

上記のような構成による本発明のソフトウェア説明書生成システムによれば、複数の自然言語で書かれたコメントを、ソースコードと共にソースファイル中に含むことで、ソフ

トウェア開発者・各国語の編集者・各国語の翻訳者がそれぞれ誤った編集を行うことを防止できる。と共に、他国語で記述されたコメントなど翻訳を行う必要のある場所を翻訳者に表示することが可能となり、編集を効率よく行うことができる。この結果として、次のような種々の課題が解決できることとなる。

【００２３】

（課題１）：これまでソフトウェア開発者は、プログラミング言語で書かれたソースコードにコメントを付与したソースファイルを作成し、そのソースファイルを入力してソフトウェア説明書を生成するＪａｖａｄｏｃのようなツールを用いてソフトウェア説明書を生成してきた。しかし、それらのツールは単一の自然言語についてのものであった。このため、複数の自然言語で書かれたソフトウェア説明書への要求に対しては、個別に各自然言語毎のファイルそれぞれに翻訳および整合性の確認を経るプロセスが必要であって、複数の自然言語で書かれたコメントを、後に処理しやすいようにソースファイル内に保持することは行われていなかった。これに対して、本発明のソフトウェア説明書生成システムを用いることにより、ソースファイルのコメントを複数の自然言語で記述し、複数の自然言語で書かれたソフトウェア説明書を生成するシステムが実現される。

【００２４】

（課題２）：自然言語と使用する国・地域が一対一に対応しない場合においても、使用者の要求に応じて適切なソフトウェア説明書を提供する必要があるが、そのような要求に応えるシステムは未だ実現されていなかった。このような課題についても本発明によるソフトウェア説明書生成システムを用いることにより解決される。

【００２５】

（課題３）：プログラミング言語で書かれたソースコードと、これらに対して自然言語で書かれたコメントからなるソースファイルにおいて、翻訳が必要な箇所を適切に判断する方法は現在のところ明らかにはなっておらず、人手による翻訳が必須となっている。そのため、ソフトウェア製品の製作プロセスにおいては、ソフトウェア説明書を作成するための多大な時間とコストが必要となっていた。このような課題についても本発明によるソフトウェア説明書生成システムを用いることにより解決される。

【００２６】

（課題４）：ソースファイル中のコメントに関する機械翻訳が適用可能であった場合においても、翻訳元のコメントを適切に選択する事は困難であり、ソフトウェア開発者の意図を反映させるための何らかの明示的な選択手段が必要である。本発明によるソフトウェア説明書生成システムを用いることにより、このような課題についても解決される。

【００２７】

（課題５）：複数の自然言語でコメントで記述されるソースファイルの記述方法および処理方法のシステムがあったとしても、ソフトウェアの仕様変更・ソースコードの実装内容にあわせて、それぞれの自然言語で書かれたコメントの内容を適切に変更・管理していくことは、非常に困難な作業である。例えば、ある自然言語で記述されたコメントを変更した場合に、その他の自然言語で記述されたコメントとの不一致が起こるが、どのコメントを最新の情報に修正するべきであるかという情報を管理することは困難である。これまで、複数の自然言語でコメントが書かれたソースファイルを適切に変更・管理していくためのシステムは実現されていない。本発明によるソフトウェア説明書生成システムを用いることにより、このような課題についても解決される。

【発明を実施するための最良の形態】

【００２８】

以下、本発明の実施する場合の形態について具体的に実施例に基づき説明する。実施例の説明において、ソフトウェア説明書生成システムは、コンピュータ（もしくはより広く言うと情報機器）上で実行されるソフトウェア（プログラム）がインストールされることにより、それぞれの処理手段として機能するシステム要素が実現されてシステムが動作する。ここでのファイルとは、コンピュータの記憶装置に格納される電子ファイルを想定している。なお、このソフトウェア説明書生成システムは、その機能を同じくしたまま大部

分をハードウェアで構成した形のスタンドアローン型のソフトウェア説明書生成装置として構成されても良い。

【００２９】

ソフトウェア説明書作成システムに入力するソースファイルには、プログラム開発において記述されるプログラミング言語で書かれたソースコード文に加えて、そのソースコード文に対応して、これに付与されたコメントを含むソースファイルが入力される。それらのコメントは２種類以上必要とされるだけの種類の自然言語により、必要な説明が記述される。この場合に、次のようにして、ソースコードの機能を説明するコメントの意味および自然言語の種類を示す符号が付けられる。

【００３０】

通常のソフトウェア説明書作成システムに入力するソースファイルであれば、コメントにはコメントの意味を示す符号が付与される。通常は、入力されたソースファイルを構文解析するなどして、意味を示す符号により識別してコメントを格納手段に格納する。しかし、本発明の対象とするソースファイルにおいては、各々の自然言語で書かれた各々のコメントには複数の自然言語で記述されるため、それらのコメントの自然言語の種類を表す符号とコメントの意味を示す符号とを組み合わせた符号がつけられ、これらの符号が併せて記述されている。そのため、コメントを格納する際には、自然言語の種類を表す符号とコメントの意味を示す符号とを組み合わせた符号により識別されて格納される。

【００３１】

このソフトウェア説明書生成システムにおいては、上述したようにして、組み合わせた符号により識別されて格納手段に格納されたコメントから、ユーザによって指定された出力対象の自然言語の種類に相当する符号がつけられたコメントのみを抽出する。これにより、ソースコード文および実行可能ソフトウェアに対応するソフトウェア説明書を、指定された出力対象の自然言語で出力することができるシステム構成とされる。

【００３２】

また、このソフトウェア説明書生成システムにおいては、入力するソースファイル中の各々の自然言語で書かれた各々のコメントに対しては、その自然言語の種類を表す符号とコメントの意味を示す符号に加えて、更に、国もしくは地域を示す符号とを組み合わせた符号がつけられており、その符号を含むコメントを抽出してソフトウェア説明書を出力するように構成される。

【００３３】

また、ここでのソフトウェア説明書生成システムにおいては、指定された自然言語のコメントを、別の自然言語で記述されたコメントを元に機械翻訳により生成して、ソフトウェア説明書を出力するようなシステム構成とされるが、入力するソースファイルに主たる自然言語の種類を示す符号を含むことと、指定された自然言語のコメントを、そこで示された主たる自然言語を元にして機械翻訳することにより生成して、ソフトウェア説明書を出力するようにも構成される。

【００３４】

更に、入力するソースファイル中のソースコード文に付与された、コメントが更新を必要とする旨の符号と組み合わせた符号を含むことと、その符号を含むコメントを解釈して更新が必要な箇所の情報を出力するようにも構成される。

【００３５】

既存のソフトウェア説明書作成システムに入力可能なソースファイルを、変換により生成するシステムとすることも有用である。すなわち、目的とする自然言語で記述されたソフトウェア説明書を、複数言語で書かれたソースファイルから直接生成するのではなく、まずは、目的とする自然言語で記述されたコメントとソースコードのみを抽出することによって、目的とする自然言語のみでコメントされたソースファイルを出力する。その後、出力されたソースファイルは、既存の説明書生成システムに入力することで、最終的には目的とする自然言語で記述されたソフトウェア説明書が生成されるように構成されても有用なシステムとなる。

【００３６】
図中に現れるソースコードの例においては、Ｊａｖａ（登録商標）言語による記述が例示されているが、しかし、本発明によるソフトウェア説明書作成システムは、Ｊａｖａ（登録商標）言語で記述されたソフトウェアソースコードにおいてのみ適用されるものでなく、Ｊａｖａ（登録商標）言語に限らず、あらゆるプログラミング言語に対しても適応可能である。
【００３７】
図１は、複数の自然言語で書かれたソフトウェア説明書を、単一のプログラムソースファイルから生成するシステムの構成を示した図である。このシステムにおけるソースファイル１０１は、第１自然言語から第ｎ自然言語までの種類の言語で記述されたコメントおよびプログラミング言語で書かれたソフトウェアプログラムを含む処理対象ソースファイル（ｎは２以上の自然数）である。
【００３８】
本発明の実施例に係るソフトウェア説明書作成システムは、図１に示されるように、システム要素として、入力処理部１１、コメント格納処理部１２、コメント抽出処理部１３、出力処理部１４、および翻訳処理部１５を備えている。説明書生成システム１０２は、コメント抽出処理部１３、出力処理部１４、翻訳処理部１５、および図示されない制御処理部から構成されている。
【００３９】
入力処理部１１は、プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイルを入力する。すなわち、入力処理部１１により入力されるソースファイル１０１のデータ構造は、プログラミング言語で書かれたソースコード文および前記ソースコード文に対して付与されたコメントを含むソースファイル内において、ソースコード中のあるひとつの機能を説明するコメントが複数の自然言語で記述されて、それぞれの自然言語の記述には機能を表す符号と自然言語の種類を表す符号とを組み合わせた符号が付加されているデータ構造である。コメント格納処理部１２が、入力されたコメントおよびソースコード文を対応づけてソースファイルを構文解析するなどして格納する。この場合に、例えば、２種類以上の自然言語で書かれるコメントについては、各々の自然言語で書かれた各々のコメントに対して当該自然言語の種類を表す符号とコメントの意味を示す符号とを組み合わせた符号（例えば@ｕ．ｊａなど）により識別して格納する。コメント抽出処理部１３は、使用者によって指定された出力対象の自然言語の種類に相当する符号（例えばｊａなど）がつけられたコメントのみを前記コメント格納処理部１２から抽出する。出力処理部１４は、前記抽出されたコメントに基づいて前記ソースコード文に対する出力対象の自然言語のソフトウェア説明書を出力する。また、ここでの翻訳処理部１５は、後述するように、１つの自然言語の文を他の自然言語の文に機械翻訳処理する。
【００４０】
入力処理部１１により利用者から入力されたソースファイル、またはコメント格納処理部１２に格納されているコメントを含む入力ソースファイル１０１は、説明書生成システム１０２に入力される。使用者は、説明書生成システム１０２に対して出力したいソフトウェア説明書の自然言語の種類を指定する。そして、説明書生成システム１０２は、コメント抽出処理部１３により、指定された出力対象の自然言語の種類に相当する符号がつけられたコメントのみを抽出することによって、出力処理部１４により、ソースコード文および実行可能ソフトウェアに対応するソフトウェア説明書を、前記の出力対象の自然言語で出力する。
【００４１】
使用者が説明書生成システム１０２に対して、第１自然言語を指定した際は、第１自然言語で書かれた説明書１０３が、第２自然言語を指定した際は、第２自然言語で書かれた説明書１０４が、また、第ｎ自然言語を指定した際は、第ｎ自然言語で書かれた説明書１０５が出力される。なお、この場合に指定する自然言語の種類は一つだけでも、複数を同

時に指定しても構わない。

【００４２】

このように、このシステムは、ソースコード中に書かれた複数種類の自然言語からなるコメントから、対象自然言語で書かれたコメントを抽出し、ソースファイルから自動的に対象自然言語のドキュメントを生成する。

【００４３】

図２は、処理対象となる入力ソースファイルの詳細を例示する図である。図２に示すように、ソースファイル２００には、プログラムのソースコードと共に各自然言語で書かれたコメントが、単一のソースファイル上に保持される。その際、ソースファイル２００においては、単一の電子ファイル上に多自然言語のテキストを格納する文字コード方式を用いる。この実施例においては、Ｕｎｉｃｏｄｅ文字セット（ＩＳＯ　１０６４６）を用いた例を示している。また、文字符号化方式としてはＵＴＦ－８（ＲＦＣ　３２６９）を用いる。しかし、多自然言語で書かれた文章を表現可能な文字セットおよび文字符号化方式を用いる限り、本発明に適用可能である。

【００４４】

図２においては、書かれたコメントがどの自然言語用のコメントであるかを識別する符号を、ソースファイル中に記述する方法の一例が示されている。Ｊａｖａ（登録商標）などのプログラミング言語においては、「／＊」と「＊／」で囲まれた領域が、本来のプログラムソースコードではなく、コメントとして処理される。この例においてコメントは「／＊＊」で始まるが、これはドキュメントコメントと呼ばれ、通常のコメントとは区別されて取り扱われる。ドキュメントコメントは、Ｊａｖａｄｏｃ等のソフトウェア文書生成システムによって解釈されて出力される。ソフトウェア文書はこのドキュメントコメントの内容に基づいて出力される。なお、この実施例における入力ソースコードの例としては、Ｊａｖａｄｏｃに倣ったコメント形式を用いているが、他のコメント形式・プログラムソースコード記述形式、例えば、ＸＭＬを用いたコメント・プログラムソースコードの記述であっても、本発明に適用可能である。他の記述形式の例については、図１０に示されている。

【００４５】

図１０に示される記述形式の例は、ＸＭＬファイル形式およびＣＳＶファイル形式の一例である。図１０の上部に示すソースファイル１１０１は、ＸＭＬファイル形式においてソースコードおよび複数の自然言語で記述されたコメントを、単一のファイル中に保持している。また、ソースファイル１１０２では、ＣＳＶ形式（Ｃｏｍｍａ　Ｓｅｐａｒａｔｅｄ　Ｖａｌｕｅｓ）ファイルにより同様の内容を保持している。この他にも、多数の類似する形式によって、同様に適用できる。例えば、ソースファイル１１０１のＸＭＬ形式の適用について説明すると、＜ｕ．ｅｎ＞という符号は、自然言語を示す記号である「ｅｎ」をＸＭＬタグの属性として扱い、＜ｕ　ｌａｎｇ＝”ｅｎ”＞とする記述による符号とするようにしてもよい。このような符号によっても、一つの組み合わせ符号の実施例として同様の効果を示すものとなる。

【００４６】

参照を図２に戻すと、図２に示すソースファイル２００において、Ｈｅｌｌｏクラスの定義文２０２に先立って示されているドキュメントコメント２０１には、このクラスの概要説明が書かれている。ここで符号として「@ｕ．ｊａ」タグが示されているが、これは符号のタグに続くテキストが、日本語で書かれたこのクラスの概要説明であることを示している。また、別の種類の符号の「@ｕ．ｋｏ」タグが示されているが、これはこのタグに続くテキストが、韓国語で書かれたこのクラスの概要説明であることを示している。同様に別の種類の符号である「@ｕ．ｚｈ」のタグは、同様にこのタグに続くテキストが、中国語で書かれたこのクラスの概要説明であることを示している。

【００４７】

ここでの符号「@ｕ」は、コメントの概要説明を意味するタグとして定義したものであり、コメントの意味を示す符号である。そして、これに続く「．ｊａ」「．ｋｏ」「．ｚ

ｈ」の符号は、使用する自然言語の種類を示すための符号である。これらの符号は組み合わされて、その組み合わせ符号「＠ｕ．ｊａ」が、「コメントの意味を示す符号」および「自然言語の種類を示すための符号」を組み合わせたものとなっており、コメントの意味と自然言語の種類を同時に表現する符号としている。

【００４８】

本発明においては、このような組み合わせ符号を用いることにより、複数の種類の自然言語の説明書の編集作業および生成を効率化することができるものとなっている。プログラム開発を行うプログラマがソースコードの編集中においても、これらの符号を用いることにより、異なる種類の、また複数の種類の自然言語で書かれたコメントを提示することができる。

【００４９】

次に、これらの符号の別の使用法の一例として、メソッドｓａｙの定義２０５に対して付与されたドキュメントコメントについて説明する。前半部分２０３は、クラスに付与されたコメントと同様であり、メソッドの概要を説明している。ここでの符号の「＠ｐａｒａｍ」タグは、Ｊａｖａｄｏｃにおいては、メソッドの引数についての説明を行うコメントに先立って付与されるタグである。これは、この符号についても「コメントの意味を示す符号」となる。この場合においては、複数の種類の自然言語に対応するため「＠ｐａｒａｍ」タグに対して、自然言語の種類を示す符号を組み合わせて、その組み合わせ符号を用いる。つまり、コメント２０４における組み合わせ符号である「＠ｐａｒａｍ．ｊａ（日本語）」タグ、「＠ｐａｒａｍ．ｋｏ（韓国語）」タグ、および「＠ｐａｒａｍ．ｚｈ（中国語）」タグにより、複数の種類の自然言語に対応した組み合わせ符号とする。これらのタグにより説明書生成システムは、それぞれのコメントが、メソッドの引数についての説明であって、更に、それぞれの自然言語用のコメントであると識別することができる。

【００５０】

なお、自然言語名の国際規格としてはＩＳＯ６３９が規定されており、アルファベット２文字で示すためにはＩＳＯ６３９－１が、３文字で示すためにはＩＳＯ６３９－２が規定されている。この実施例においては、ＩＳＯ６３９－１を使用しているが、適切な自然言語名を規定すればＩＳＯ標準に限らずに、本発明のソフトウェア説明書生成システムに用いることができる。以上の表記法の採用により、本発明におけるソフトウェア説明書生成システムは、生成対象の自然言語に関連するコメントだけを抽出することができる。

【００５１】

図２のソースファイルを入力とし、出力対象言語を英語（ｅｎ）とした場合の説明書出力例を、図１２に示している。説明書の生成には既存の生成システムであるＪａｖａｄｏｃを用いた。また、図２のソースファイルを入力とし、出力対象言語を日本語（ｊａ）とした場合の説明書の出力例を図１３に示している。それぞれ、ソースファイル中のコメントから対応する言語の説明が抽出されて出力されたものとなっている。

【００５２】

図３は、複数自然言語でコメントされ更に対象の国もしくは地域を指定された入力ソースファイルの例を示す図である。図３に例示される入力ソースファイルは、複数の種類の自然言語でコメントされ、更に、対象の国もしくは地域を指定されたコメントを含む入力ソースファイルである。ここでのコメントに付加する符号として、図３に示すように、自然言語コードに加えて更に国コードが更に組み合わせられて使用される。これにより、処理の対象となる自然言語と国とを共に指定することが可能になる。

【００５３】

図３に示すソースファイル３００において、Ｈｅｌｌｏクラスの定義３０２に付与されたドキュメントコメント３０１においては、組み合わせ符号の「＠ｕ．ｄｅ－ｂｅ」タグおよび「＠ｕ．ｆｒ－ｂｅ」タグが用いられているが、この「＠ｕ．ｄｅ－ｂｅ」タグは、メソッド引数の説明を行う目的であって、ベルギー王国においてドイツ語を第一自然言語とする人に向けたコメントであることを示している。また、「＠ｕ．ｆｒ－ｂｅ」は対

象とする自然言語の種類が、ドイツ語ではなくてフランス語であることを示している。

【００５４】

以上の例により、自然言語の種類を示す符号（言語コード）に加えて国の種類を示す符号（国コード）を、更に、組み合わせて使用することにより、対象となる自然言語と国とを共に指定する符号とすることができる。このため、ソースコードと対応づけるコメントに付加する符号によりキメ細やかなドキュメントの書き分けが可能となる。

【００５５】

図４は、入力ソースファイルからソフトウェア説明書を生成する処理を説明するフローチャートである。これまでに説明したような入力ソースファイルがあった場合に、説明書生成処理がどのように行われるかを示す図となっている。説明書生成システム１０２は、初めにソースファイルの読み込み（ステップ４０１）を行う。その読み込み処理の後に（若しくは並行して）、説明書のシステム内部モデルを生成する（ステップ４０２）。ここでいう説明書のシステム内部モデルとは、この処理を行うコンピュータのメモリ上に説明書の構造を表現したものである。なお、この過程は多くの説明書生成システムに共通のものであり、詳細な説明は省略するが、本発明において特徴的な点は、説明書のシステム内部モデルから指定された自然言語（国・地域）を示す符号がつけられたコメントを抽出し、抽出したコメントのみを用いて、説明書を生成する（ステップ４０３）ことである。これにより、各国・地域向けのドキュメントを使用者の指定に応じて出力することが可能になる。

【００５６】

図５は、ソースファイルからソフトウェア説明書を生成する処理の別の例を説明するフローチャートである。この場合の説明書生成処理の処理手順においては、ソースファイルを読み込み（ステップ５０１）、説明書のシステム内部モデルを生成する（ステップ５０２）時点で、既に必要な情報（ドキュメントコメント）のみを取り込む。この時点において、使用者によって指定されていない自然言語に関するドキュメントコメントはフィルタアウトされる。そのため、説明書のシステム内部モデルから出力する説明書を生成する処理（ステップ５０３）は、従来の説明書生成システムと全く同様の処理プロセスを用いることが可能となっている。

【００５７】

図６は、ソースファイルからソフトウェア説明書を生成する処理の更に別の例を説明するフローチャートである。この処理の例では、機械翻訳を含む説明書生成の処理手順を示している。説明書生成システムは、初めにソースファイルの読み込み（ステップ６０１）を行う。その読み込み処理の後に説明書のシステム内部モデルを作成する（ステップ６０２）。次に、作成したシステム内部モデルにおいて指定された自然言語（国・地域）に対応するコメントがない場合、もしくは古い場合は機械翻訳によりコメントを作成する（ステップ６０３）。そして、説明書のシステム内部モデルから指定された自然言語（国・地域）を示す符号が付けられたコメントのみを用いて説明書を生成する（ステップ６０４）。これにより、各国・地域向けのドキュメントを使用者の指定に応じて出力することができる。図６に示される処理手順は、図４の手順のステップ４０２がステップ６０２に、ステップ４０３がステップ６０４にそれぞれ対応する。ステップ６０２とステップ６０４の間に、ステップ６０３の処理プロセスが挿入された処理手順となっている。ステップ６０３の処理においては、説明書のシステム内部モデルにおいて、指定された自然言語（国・地域）に対応するコメントがない若しくは古い場合に、機械翻訳により生成する処理を行う。

【００５８】

このような処理を行うことによって、人手による翻訳を介さずに必要な自然言語の説明書が生成可能になるため、非常に大きい利点となる。ここでの機械翻訳とは、別の言い方では電子翻訳・コンピュータ翻訳などに相当するものを想定しており、人間の翻訳者ではない翻訳機械による処理により翻訳作業を行わせるものである。機械翻訳における訳文の正当性・妥当性に関する信頼性が十分に高くない場合には、機械翻訳による訳文について

は、その文が機械翻訳によるものである旨を示す符号を併せて付加することが好ましい。この符号により、後に機械翻訳文の見直しが可能になる。

【００５９】

また、機械翻訳の処理を行うシステム要素（機械翻訳処理モジュール）は、システム内に含有する必要は必ずしもない。内部データ表現だけで処理するのではなく、外部ファイルやクリップボードを用いて、システム外部のソフトウェア・サービスを利用する方法であっても良い。また、ＯＬＥ（Ｏｂｊｅｃｔ Ｌｉｎｋｉｎｇ ａｎｄ Ｅｍｂｅｄｄｉｎｇ）の枠組みを用いるなどして機械翻訳の機能を追加する方法によりシステムが構成されても良い。

【００６０】

本発明によるソフトウェア説明書作成システムを用いて、複数の種類の自然言語で書かれたコメントを含むソースコードを用いたプログラム開発を行う際の重要な点は、ソースファイルを作成する際に、実際にはどのようにして、複数の種類の自然言語で書かれたコメントを含むソースファイルを作成していくか、ということである。

【００６１】

例えば、本発明によるソフトウェア説明書作成システムを用いない一般的な開発モデルにおいては、ソフトウェア設計者により設計されたソフトウェアは、ソフトウェア実装者によってソースファイルの形に具現化される。その際、ソースコードに付与されるコメントの自然言語は、該当プロジェクトにおいて規定される第１自然言語となる。当初のコメントは、その第１自然言語で記述されるのが通常である。

【００６２】

しかし、本発明によるソフトウェア説明書作成システムを用いると、この場合には、ソースファイル毎に主とする自然言語を設定できることと、ソースコード中に含まれる翻訳作業を完全分業化できることから、必ずしもプロジェクト単位で自然言語の統一を図る必要が無い。そのため、プロジェクトのメンバーには最も理解が容易であり、作業が効率的に進む自然言語を選択することにより、作業が効率よく進められる。

【００６３】

また、前述した翻訳作業において、翻訳の元となるオリジナル自然言語としてどの種類の言語を使うか、という点についても重要である。決定方法としては、１：使用者が指定する、２：使用者の作業環境の使用自然言語をオリジナル自然言語として指定する、などが考えられる。

【００６４】

しかし、ソースファイルもしくはクラスやメソッドという単位で開発者が違うという状況は多くの場合に起こりうる。これに対しては、そのため、ソースファイルに、主たる自然言語の種類を示す符号を含ませることで、説明書生成システムに対して情報を与える方式の採用が望ましい。

【００６５】

図７は主たる自然言語の種類を定めた符号を用いた機械翻訳を含む説明書生成の処理手順を示す図である。この処理では、ステップ７０１において、主とした自然言語を示す符号を含んだソースファイルを読み込む。次のステップ７０２において、内部モデルの生成を行い、ステップ７０３においては、内部モデルにおいて、指定された自然言語（国・地域）に対応するコメントがない若しくは古い場合は、機械翻訳によりコメントを生成する過程である。その際の訳元の自然言語としては、主たる自然言語を用いる。次のステップ７０４の処理においては、これまでと同様に説明書の生成を行う。

【００６６】

このようなソースファイルの例として、主たる自然言語を示す符号を含む入力ソースファイルの一例を、図１４に示している。ソースファイル１５００の冒頭部分１５０３において、コメント中に@ｍａｉｎｌａｎｇ．ｊａが含まれている。これは主たる自然言語として、日本語を用いるという意味である。例えばこのソースファイル１５００のコメントとして存在しないドイツ語のソフトウェア説明書を出力したい場合を考えてみる。システ

ムの使用者が訳元の言語を特に指定しなかった場合、仮にこの符号がない場合を考えると、訳元として複数の自然言語のコメント部分１５０１の日本語・韓国語・中国語のうち、どれを用いたらよいか不明である。この場合、冒頭部分１５０３に@ｍａｉｎｌａｎｇ．ｊａがあることにより、訳元のコメントとしては、デフォルトとして@ｕ．ｊａで示される日本語コメントを選択すれば良いことが明らかとなる。結果として、説明書生成システムは、そこで示された主たる自然言語を元にして機械翻訳することにより適切なコメントが得られ、ソフトウェア説明書を出力する事が可能になる。これにより、翻訳割り当て作業の簡易化を図ることができる。

【００６７】

また、本発明によるソフトウェア説明書生成システムにおいては、入力するソースファイルにおいて、各自然言語で書かれたコメント（コメント）間での意味の一貫性を保つことが重要である。

【００６８】

図８は、各自然言語で書かれたコメントについて、更新が必要な箇所を示す符号を含む入力ソースファイルの例を示す図である。図８に示すソースファイルの例は、全体がＨｅｌｌｏクラスの定義を示しており、その中のメソッドｓａｙの定義９０２に対して、コメント９０１が付与されている。ソフトウェアの仕様変更に伴いメソッド定義９０２に変更が生じた際、もしくはコメント９０１に修正が加わる、という状況は、ソフトウェア開発の現場においては頻繁に起こり得る。図８に示すソースファイルの一例は、図２に示されたソースファイルにおいてｓａｙメソッドの引数の名称がｒｅｐｅａｔからｌｉｎｅｓに変更になった状況を示している。以下に、ソースコード編集の順序を示す。

（１）：「ｓａｙ（ｉｎｔ　ｒｅｐｅａｔ）」を「ｓａｙ（ｉｎｔ　ｌｉｎｅｓ）」と変更した。

（２）：それに伴い、コメント中の「@ｐａｒａｍ　ｒｅｐｅａｔ　Ｒｅｐｅａｔｉｎｇ　ｎｕｍｂｅｒ　ｏｆ　ｔｈｅ　ｇｒｅｅｔｉｎｇ」を、「@ｐａｒａｍ　ｌｉｎｅｓ　Ｎｕｍｂｅｒ　ｏｆ　ｌｉｎｅｓ　ｆｏｒ　ｔｈｅ　ｇｒｅｅｔｉｎｇ」と、英語で記述されたコメントの内容を変更した。

（３）：英語のコメントの内容を変更したが、編集者は英語以外の自然言語が読み書きできない。そのため、他の自然言語については、「@ｐａｒａｍ．ｊａ」などのタグの後ろに、「／ｕｐｄａｔｅ」という符号を付加し、「@ｐａｒａｍ．ｊａ／ｕｐｄａｔｅ」とした。

【００６９】

以上の編集作業を行うことにより、英語以外の他の自然言語について、少なくとも再度翻訳の必要があるかどうかの判断が可能となる。このような方法の採用によって、各自然言語で書かれたコメント間での意味の一貫性を保つことが容易になる。

【００７０】

図９は、本発明によるソフトウェア説明書生成システムにおいて更新が必要な箇所が確認可能な説明書生成の処理手順を示す図である。この処理手順においては、これまでと同様に、ステップ１００１においてソースファイルを読み込み、次のステップ１００２においてシステム内部モデルの生成を行う。そして、次のステップ１００３において、図４の手順に加えて、システム内部モデルにおいて、指定された自然言語（国・地域）に、「更新が必要な旨の符号」があった場合に記録を行う過程が追加される。ステップ１００４においては、これまでと同様に説明書の生成を行うが、次のステップ１００５において、更新が必要なコメントの箇所に関する情報、もしくは更新が必要な自然言語がどれであるかを、全ステップで記録された情報を元に出力する。本発明によるソフトウェア説明書生成システムにおいては、使用者は、この情報を元に翻訳作業を完全に分業化することが可能になる。

【００７１】

なお、本発明のソフトウェア説明書生成システムにおいて、ここでの本質は、更新が必要な旨を示す符号を付与することにあるため、本発明の適用範囲が、この実施例で説明し

た例に限定されるわけではない。

【００７２】

また、本発明によるソフトウェア説明書生成システムは、以上で説明したそれぞれの手段により直接、目的とする自然言語で記述されたソフトウェア説明書を生成するシステム・装置・方法として実施しても良いが、必ずしも直接生成する実施形態を取る必要はない。複数の種類の自然言語を用いて記述されたコメントから、目的とする自然言語で記述されたコメントのみを抽出することによって、単一自然言語でコメントされたソースファイルを出力する電子ファイルの変換を行う実現形態も、本発明のソフトウェア説明書生成システムの有効な実装である。

【００７３】

その概要を図１１に示す。図１１において、参照符号１２０１は、これまで説明したソフトウェア説明書生成システム・装置・方法におけるものと同様のソースファイルである。電子ファイル変換システム１２０２に、その複数自然言語でコメントされたソースファイル１２０１が入力され、使用者が第１自然言語を選択した場合は、コメントが第１自然言語で書かれたソースファイル１２０３を出力し、また、使用者が第２自然言語を選択した場合は、コメントが第２自然言語で書かれたソースファイル１２０４を出力する。同様に第ｎ自然言語で書かれたソースファイル１２０５を出力する。

【００７４】

出力されたソースファイル（１２０３、１２０４、…、１２０５）は、それぞれ既存のソフトウェア説明書生成システム（１２１３、１２１４、…、１２１５）に入力することで、目標とする各自然言語で書かれたソフトウェア説明書（１２２３、１２２４、…、１２２５）を生成することができ、目的が達成される。既存のソフトウェア説明書生成システムを利用できるということは、本発明を適用して新たにシステムを開発することなしに様々な表現形態でのソフトウェア説明書が生成可能であるため、非常に有用である。

【００７５】

本発明において入力対象となるソースファイルには、自然言語の種類を表す符号とコメントの意味を示す符号とを組み合わせた符号が数多く含まれることになる。従来のテキストエディタ上でこの種のファイルを編集作業する場合には、各自然言語でコメントを記述する作業に加えて、上記の組み合わせ符号を記述する作業の量が増大するため、編集作業の効率が低下するおそれがある。そのため、符号を挿入する機構を、既存のテキストエディタのようなソフトウェア開発用の編集システムに設けることにより、この問題を回避することが可能である。

【００７６】

さらに、多くの自然言語を用いてコメントが記述されたソースファイルをテキストエディタ上で編集作業することを考えてみると、取り扱う自然言語の種類が増加するにつれてコメントとソースコードの画面上での位置が離れてしまい、編集作業が困難になる。また、編集作業中に、編集者が自分では理解できない自然言語のコメントの一部を誤って削除もしくは変更してしまって、そのことに気づかないという危険性がある。そのため、必要とされる自然言語のみを編集画面上に提示する機構を、既存のテキストエディタのようなソフトウェア開発用の編集システムに設けることにより、この問題を回避することが可能である。

【００７７】

具体的には、多くの自然言語でコメントされたソースコードを編集する際には、本発明における入力ソースファイルに用いる形式で記述を行い、そのソースファイルを編集対象として用いる。テキストエディタは、自然言語の種類を表す符号とコメントの意味を示す符号とを組み合わせた符号を利用してコメントを各自然言語毎に識別可能であり、必要な自然言語で記述されたコメントのみを編集者に対して提示することが可能になる。その結果、多くの自然言語でコメントが記述されたソースコードの編集効率の向上を図り、品質の向上を行うことが出来る。

【００７８】

図１５は、ソースファイル構造の別の例を示す図である。本発明によるソフトウェア説明書生成システムにおいて利用されるソースファイル構造としては、図１５に示すような形態のソースファイル構造が利用できる。図１５に示すソースファイル構造のソースファイル１６００においては、クラスＨｅｌｌｏのメソッドｓａｙの説明としてコメント１６０３が、引数ｐａｒａｍに対する説明としてコメント１６０４が付加されている。これらのコメント１６０３およびコメント１６０４のデータ構造は、それぞれ、機能を説明するコメント内容が機能を表す符号「＠ｕ」および「＠ｐａｒａｍ」に続いて複数の自然言語で記述されており、更にその記述に用いられている自然言語の種類を示す符号「＠ｊａ」、「＠ｋｏ」、「＠ｚｈ」が付加されているものとなっている。

【００７９】

本発明によるソフトウェア説明書生成システムは、図１５で示されるソースファイル構造のソースファイルを入力として用いることも有用である。すなわち、ソースコード中のあるひとつの機能を説明するコメントが、機能を表す符号に続いて複数の自然言語で記述されており、前記コメントには、記述に用いられている自然言語の種類を示す符号が付加されているので、複数の自然言語によるコメントを追加して記述する場合に有効に利用できる。このようなデータ構造を入力ファイルとして用いる実現形態も、本発明のソフトウェア説明書生成システムの有効な実装となる。

【産業上の利用可能性】

【００８０】

このように本発明のソフトウェア説明書生成システムによれば、各自然言語で書かれたコメントおよびソースコードを単一のファイル中に含むことで、各国・地域向けのソフトウェア説明書を、単一のソースファイルから生成することができる。単一のソースファイルに、すべての自然言語のバージョンを保持するということは、情報源の分散を防ぐことを意味し、一致性を保つ効果がある。ひいては各自然言語バージョン間での不一致を削減することにつながり、説明書の作成時間の短縮、および説明書の品質向上を同時に達成することが可能となる。

【００８１】

また、ソフトウェア説明書をそれぞれの自然言語別に生成することに加えて、国・地域に応じて適切なソフトウェア説明書を生成することができ、更にきめ細かな顧客への対応が可能になる。更に、明示的に自然言語の種類をソースコードコメントに付与することによって、従来は全く不可能であったソースコードコメントの機械翻訳による生成が可能になる。これは、ソフトウェア製品を全世界的に展開する際の、コストおよび時間の大幅な節約につながる。

【００８２】

また、機械翻訳を行う際に問題となる適切な訳元文章の選択に関しても、明示的な符号の使用によって、ソフトウェアプロジェクトの意図を反映させることが可能であり、生成されるソフトウェア説明書の品質向上に貢献する。スペルチェックもしくは文法チェックは、従来、単一の自然言語に対して行われるものであったが、複数の自然言語が混在するファイルに対してスペルチェックおよび文法チェックを可能にすることで、コメントの質を向上することができる。また、複数の自然言語によりコメントが記述された際に問題となるコメントの不整合性についても、更新が必要となる符号を用いることにより効率的に修正箇所を判断することが可能になり、更に、コメントが複数の自然言語で記述されたソースファイルから、単一の自然言語で記述されたソースファイルへ変換するシステムを用いることで、既存のソフトウェア説明書生成システムの有効利用が可能になる。

【図面の簡単な説明】

【００８３】

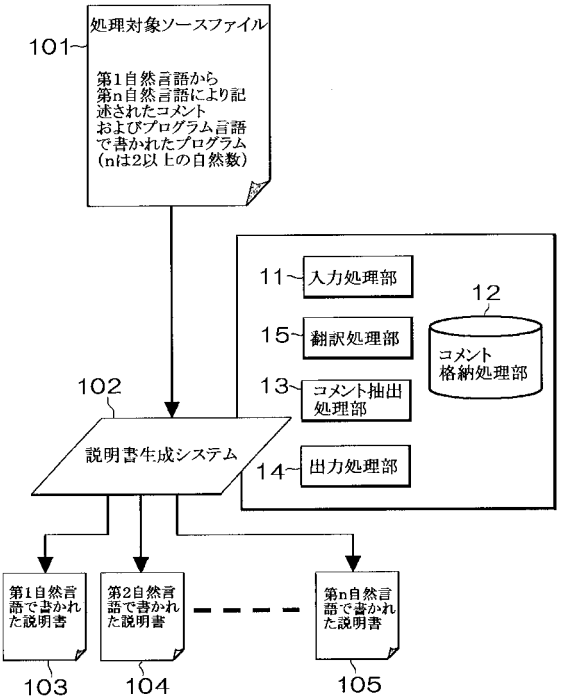
【図１】本発明における説明書生成システムを示す概略図である。

【図２】複数自然言語でコメントされた入力ソースファイルを示す図である。

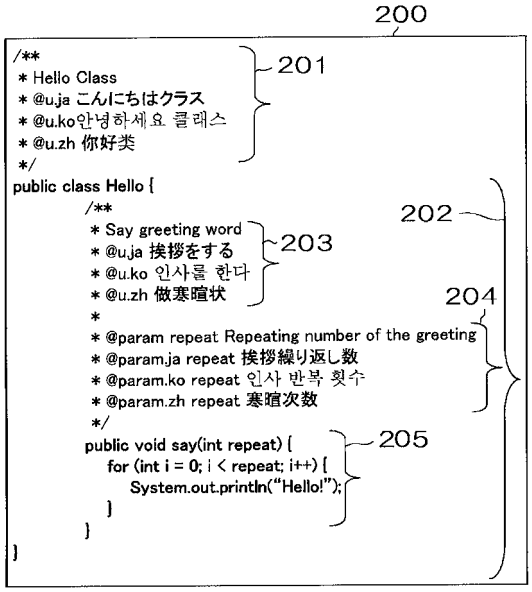
【図３】複数自然言語でコメントされ更に対象の国もしくは地域を指定された入力ソースファイルの例を示す図である。

【図４】説明書生成の処理手順を示す概略図である。
【図５】説明書生成のもう一つの処理手順を示す概略図である。
【図６】機械翻訳を含む説明書生成の処理手順を示す図である。
【図７】主たる自然言語の種類を定めた符号を用いた機械翻訳を含む説明書生成の処理手順を示す図である。
【図８】更新が必要な箇所を示す符号を含む入力ソースファイルを示す図である。
【図９】更新が必要な箇所を確認可能な説明書生成の処理手順を示す図である。
【図１０】コメント形式の例（ＸＭＬおよびＣＳＶ）を示す図である。
【図１１】電子ファイル変換システムの概要を示す図である。
【図１２】英語で書かれた説明書の出力例を示す図である。
【図１３】日本語で書かれた説明書の出力例を示す図である。
【図１４】主たる自然言語を示す符号を含む入力ソースファイルの例を示す図である。
【図１５】ソースファイル構造の別の例を示す図である。
【符号の説明】
【００８４】
１１　入力処理部
１２　コメント格納処理部
１３　コメント抽出処理部
１４　出力処理部
１５　翻訳処理部
１０１　処理対象ソースファイル
１０２　説明書生成システム
１０３　第１自然言語で書かれた説明書
１０４　第２自然言語で書かれた説明書
１０５　第ｎ自然言語で書かれた説明書

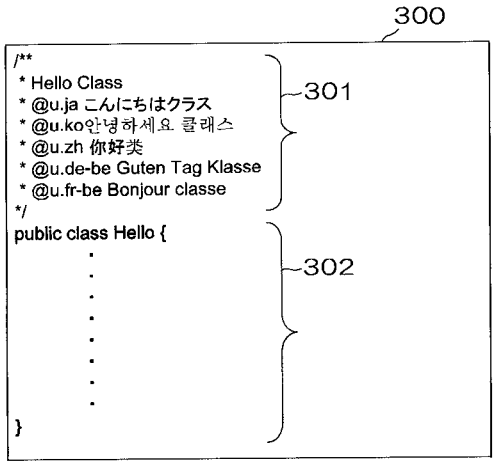
【図１】


101
処理対象ソースファイル
第1自然言語から第n自然言語により記述されたコメントおよびプログラム言語で書かれたプログラム（nは2以上の自然数）
11
入力処理部
12
コメント格納処理部
15
翻訳処理部
13
コメント抽出処理部
102
説明書生成システム
14
出力処理部
第1自然言語で書かれた説明書
第2自然言語で書かれた説明書
第n自然言語で書かれた説明書
103
104
105

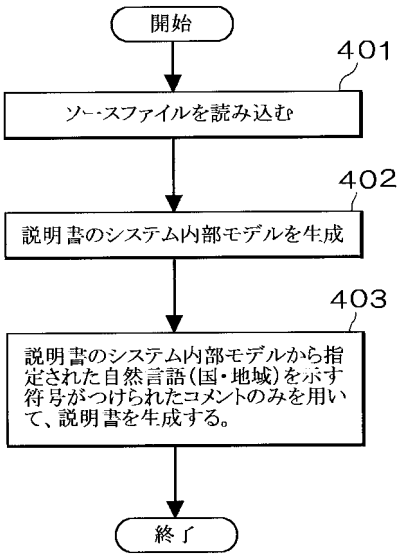
【図２】


200
/**
* Hello Class
* @u.ja こんにちはクラス
* @u.ko안녕하세요 클래스
* @u.zh 你好类
*/
201
public class Hello {
/**
* Say greeting word
* @u.ja 挨拶をする
* @u.ko 인사를 한다
* @u.zh 做寒暄状
203
202
204
*
* @param repeat Repeating number of the greeting
* @param.ja repeat 挨拶繰り返し数
* @param.ko repeat 인사 반복 횟수
* @param.zh repeat 寒暄次数
*/
public void say(int repeat) {
for (int i = 0; i < repeat; i++) {
System.out.println("Hello!");
}
205
}
}

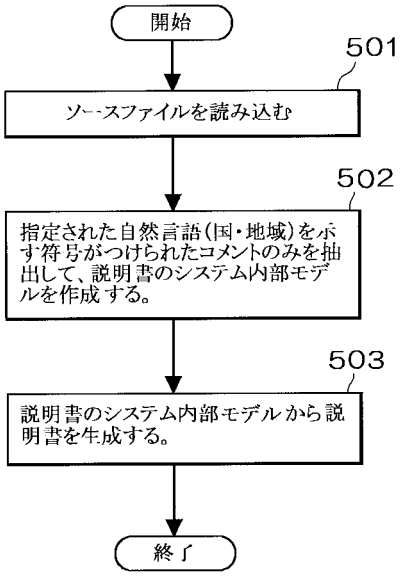
【図３】


300
/**
* Hello Class
* @u.ja こんにちはクラス
* @u.ko안녕하세요 클래스
* @u.zh 你好类
* @u.de-be Guten Tag Klasse
* @u.fr-be Bonjour classe
*/
301
public class Hello {
302
}

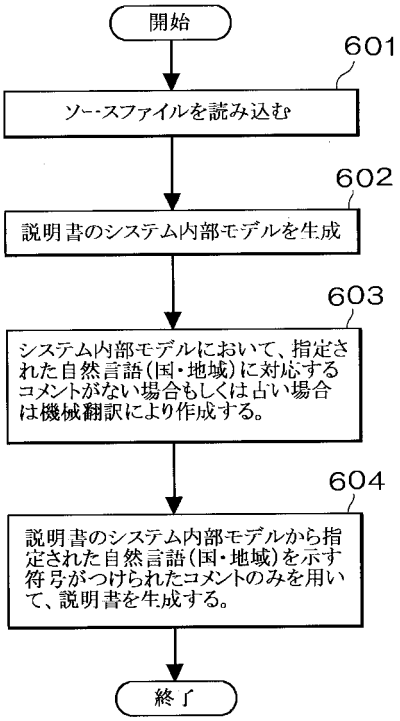
【図４】


開始
401
ソースファイルを読み込む
402
説明書のシステム内部モデルを生成
403
説明書のシステム内部モデルから指定された自然言語（国・地域）を示す符号がつけられたコメントのみを用いて、説明書を生成する。
終了

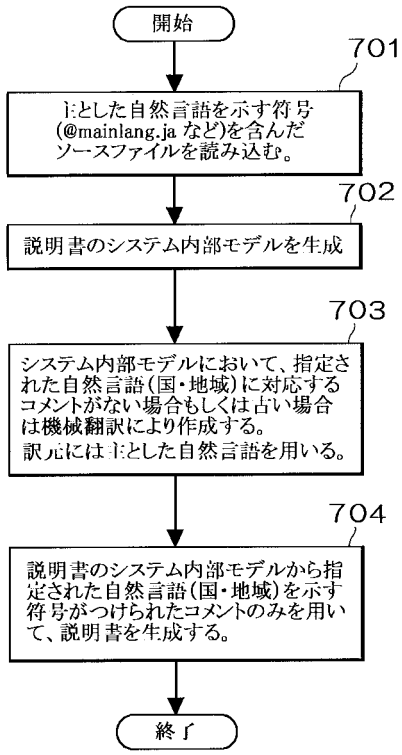
【図５】


開始
501
ソースファイルを読み込む
502
指定された自然言語(国・地域)を示す符号がつけられたコメントのみを抽出して、説明書のシステム内部モデルを作成する。
503
説明書のシステム内部モデルから説明書を生成する。
終了

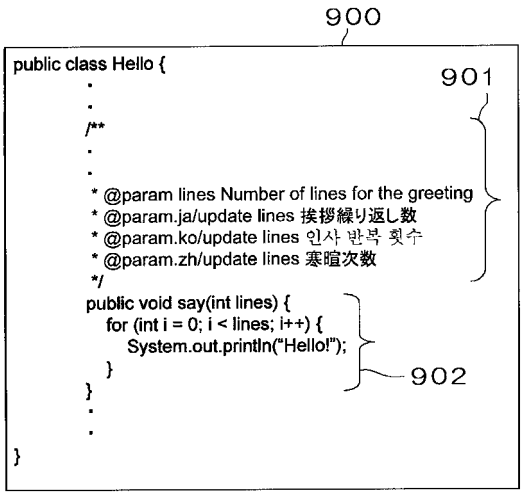
【図６】


開始
601
ソースファイルを読み込む
602
説明書のシステム内部モデルを生成
603
システム内部モデルにおいて、指定された自然言語(国・地域)に対応するコメントがない場合もしくは古い場合は機械翻訳により作成する。
604
説明書のシステム内部モデルから指定された自然言語(国・地域)を示す符号がつけられたコメントのみを用いて、説明書を生成する。
終了

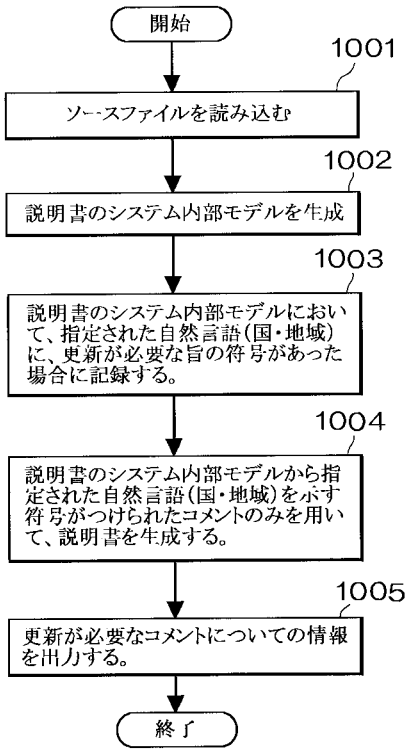
【図７】


開始
701
主とした自然言語を示す符号(@mainlang.ja など)を含んだソースファイルを読み込む。
702
説明書のシステム内部モデルを生成
703
システム内部モデルにおいて、指定された自然言語(国・地域)に対応するコメントがない場合もしくは古い場合は機械翻訳により作成する。
訳元には主とした自然言語を用いる。
704
説明書のシステム内部モデルから指定された自然言語(国・地域)を示す符号がつけられたコメントのみを用いて、説明書を生成する。
終了

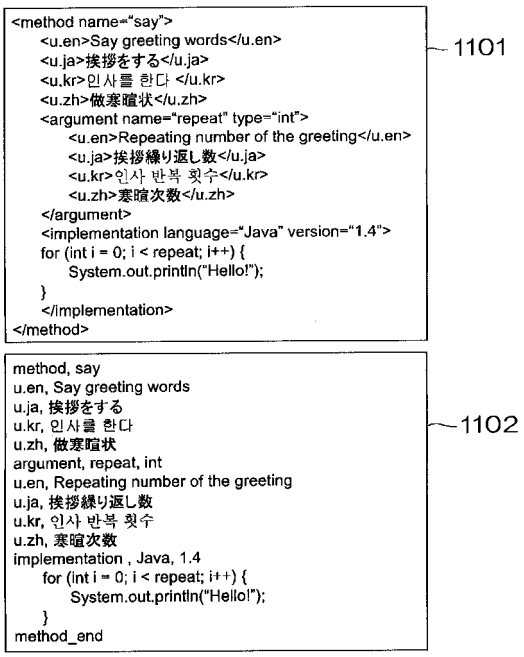
【図８】


900
public class Hello {
901
/**
* @param lines Number of lines for the greeting
* @param.ja/update lines 挨拶繰り返し数
* @param.ko/update lines 인사 반복 횟수
* @param.zh/update lines 寒暄次数
*/
public void say(int lines) {
for (int i = 0; i < lines; i++) {
System.out.println("Hello!");
}
902
}
}

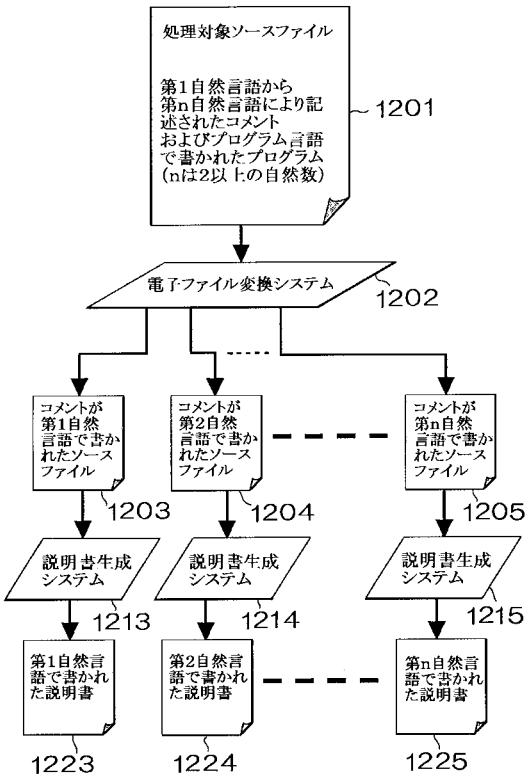
【図９】



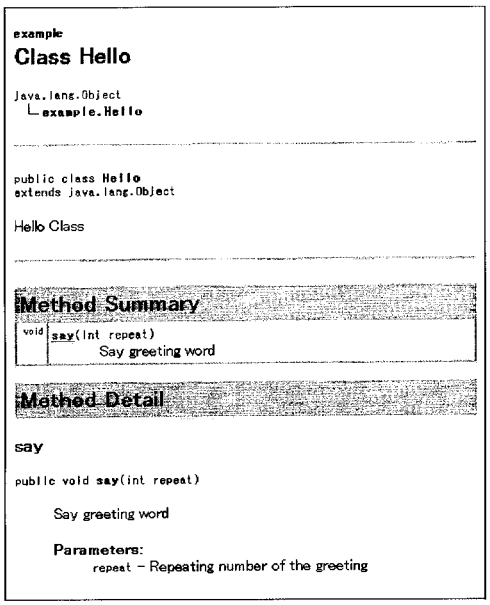
【図１０】



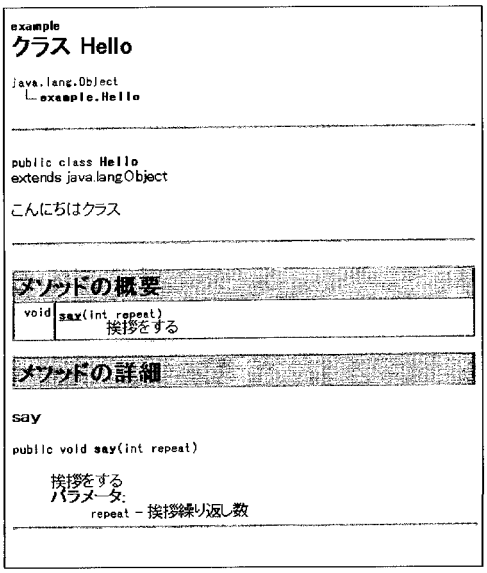
【図１１】



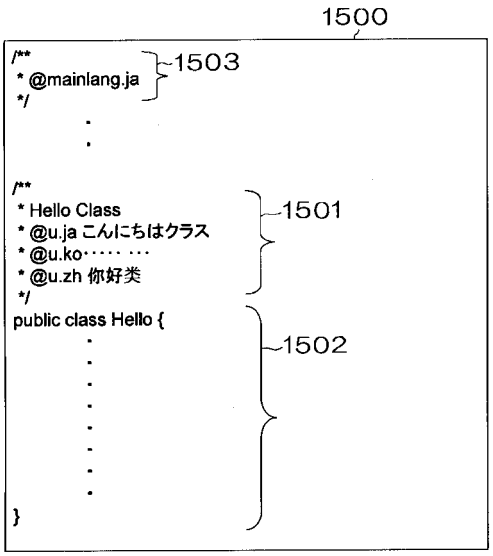
【図１２】



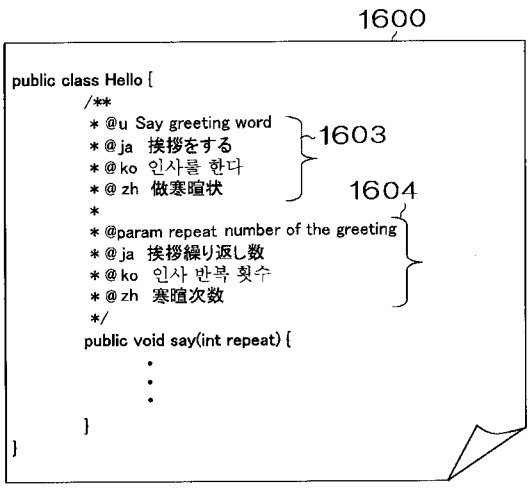
【図１３】


example
クラス Hello
java.lang.Object
└ example.Hello
public class Hello
extends java.lang.Object
こんにちはクラス
メソッドの概要
void
say(int repeat)
挨拶をする
メソッドの詳細
say
public void say(int repeat)
挨拶をする
パラメータ:
repeat - 挨拶繰り返し数

【図１４】


1500
/**
* @mainlang.ja
*/
1503
/**
* Hello Class
* @u.ja こんにちはクラス
* @u.ko…… …
* @u.zh 你好类
*/
1501
public class Hello {
1502
}

【図１５】


1600
public class Hello {
/**
* @u Say greeting word
* @ja 挨拶をする
* @ko 인사를 한다
* @zh 做寒暄状
1603
*
* @param repeat number of the greeting
* @ja 挨拶繰り返し数
* @ko 인사 반복 횟수
* @zh 寒暄次数
1604
*/
public void say(int repeat) {
}
}