令和元年１０月３１日判決言渡

平成３１年（ネ）第１００３４号　損害賠償請求控訴事件

（原審・東京地方裁判所平成２９年（ワ）第３１７０６号）

口頭弁論終結日　令和元年９月３日

判　　決

控　訴　人　　株式会社パッセルインテグレーション

訴訟代理人弁護士　中　村　隆　夫

加　藤　伸　樹

我　妻　崇　明

被　控　訴　人　　ソフトバンクロボティクス株式会社

訴訟代理人弁護士　鮫　島　正　洋

和　田　祐　造

森　下　梓

主　　文

１　本件控訴を棄却する。

２　控訴費用は控訴人の負担とする。

事 実 及 び 理 由

第１　控訴の趣旨

１　原判決を取り消す。

２　被控訴人は，控訴人に対し，３億４９１５万５０００円及びこれに対する平成２９年１０月１８日から支払済みまで年５分の割合による金員を支払え。

第２　事案の概要（略称は，特に断りのない限り，原判決に従う。）

１　事案の要旨

本件は，発明の名称を「情報管理方法，情報管理プログラム，及び情報管理装置」とする特許（特許第３７５４４３８号。請求項の数１５。以下「本件特許」という。）に係る特許権（以下「本件特許権」という。）を有していた控訴人が，被控訴人においてウェブサイト上で提供している別紙プログラム目録記載のプログラム（以下「被告プログラム」という。）が本件特許の特許請求の範囲の請求項１４に係る発明（以下「本件発明」という。）の技術的範囲に属し，被控訴人による被告プログラムのウェブサイト上での提供等が本件特許権の侵害に当たる旨主張して，被控訴人に対し，本件特許権侵害の不法行為に基づく損害賠償として３億４９１５万５０００円及びこれに対する不法行為の後の日である平成２９年１０月１８日（訴状送達の日の翌日）から支払済みまで民法所定の年５分の割合による遅延損害金の支払を求めた事案である。

原判決は，被告プログラムは本件発明の技術的範囲に属すると認めることはできないから，その余の点について判断するまでもなく，控訴人の請求は理由がないとして，これを棄却した。

控訴人は，原判決を不服として，本件控訴を提起した。

２　前提事実

以下のとおり訂正するほか，原判決の「事実及び理由」の第２の２記載のとおりであるから，これを引用する。

⑴　原判決２頁１６行目から２０行目までを次のとおり改める。

「　本件特許の特許請求の範囲の請求項１及び１４の記載は，次のとおりである。

【請求項１】

コンピュータが情報を管理する情報管理方法であって，

前記コンピュータに複数のノードそれぞれに対応付けて入力された管理すべき情報を，前記ノードを識別するノード識別情報に対応付けられた複

数のノードデータを含む文書ファイルとして前記コンピュータが記憶する情報記憶ステップと，

前記情報記憶ステップで記憶された前記文書ファイルの情報を前記コンピュータが表示する情報表示ステップと，

前記ノードデータに含まれるスクリプトを前記コンピュータが実行する情報評価ステップとを備え，

前記ノードデータは，ルートノードを除いて，当該ノードの親ノードを特定する親ノード識別情報を含んでおり，

前記スクリプトは，当該ノードデータに含まれる変数データである自ノード変数データと，当該ノードの直系上位ノードのノードデータに含まれる変数データである上位ノード変数データを利用した演算を行って，前記自ノード変数データの値を求める代入用スクリプトを含んでおり，

前記情報表示ステップは，前記親ノード識別情報を利用して，前記ノードの木構造を表示する木構造表示ステップと，前記表示された木構造のノードのうちの選択されたノードの前記自ノード変数データ，前記上位ノード変数データ及び前記スクリプトを表示するノードデータテーブル表示ステップを含み，

前記情報評価ステップは，前記代入用スクリプトの実行により，前記自ノード変数データの値を更新するステップを含む情報管理方法。

【請求項１４】

請求項１ないし１３のいずれか１項記載の情報管理方法における各ステップを，コンピュータに実行させるための情報管理プログラム。

⑷　本件発明の構成要件の分説

本件発明は，請求項１記載の情報管理方法における各ステップを発明特定事項に含むものであるところ，かかる本件発明を構成要件に分説すると，次のとおりである（以下，頭書の記号に従って，「構成要件Ａ」などとい

う。）。」

⑵　原判決４頁１３行目の「本件ｂｅｈａｖｉｏｒ．ｘａｒ」を「本件ｂｅｈａｖｉｏｒ．ｘａｒ１」と改め，同行末尾に行を改めて次のとおり加える。

「ウ　被告プログラムの内容は,原判決別紙３－１被告プログラム説明書(以下「別紙被告プログラム説明書」という。）記載のとおりである。」

⑶　原判決４頁１６行目の「甲４～６」を「甲４ないし７，３０ないし３３」と改める。

３　争点

⑴　被告プログラムの本件発明の技術的範囲の属否（争点１）

ア　構成要件ＡないしＤの充足性（争点１－１）

イ　構成要件Ｅの充足性（争点１－２）

ウ　構成要件Ｆの充足性（争点１－３）

エ　構成要件Ｇの充足性（争点１－４）

オ　構成要件Ｈ及びＩの充足性（争点１－５）

⑵　無効の抗弁の成否（争点２）

ア　本件特許出願の優先日前に頒布された刊行物である乙９（特開平６－１７５８５２号公報）に記載された発明（以下「乙９発明」という。）を主引用例とする本件発明の新規性又は進歩性の欠如（争点２－１）

イ　本件特許出願の優先日前に頒布された刊行物である乙１６（特開平１０－６９３７９号公報）に記載された発明（以下「乙１６発明」という。）を主引用例とする本件発明の新規性又は進歩性の欠如（争点２－２）

ウ　本件特許は特許法３６条６項１号及び同条４項１号に違反しているか（争点２－３）

エ　本件特許は特許法３６条６項２号に違反しているか（争点２－４）

⑶　損害の発生の有無及びその額（争点３）

４　争点に関する当事者の主張

⑴　争点１（被告プログラムの本件発明の技術的範囲の属否）について

ア　争点１－１（構成要件ＡないしＤの充足性）について

【控訴人の主張】

（ア）　本件発明の特許請求の範囲（請求項１記載の各ステップを発明特定事項に含む請求項１４。以下同じ。）の記載によれば，構成要件Ｂの「管理すべき情報」は，ノードを識別する「ノード識別情報」に対応付けられた「複数のノードデータ」を含む「文書ファイル」としてコンピュータに記憶されるものであり，この「ノードデータ」は，「スクリプト」，「親ノード識別情報」，「自ノード変数データ」，「上位ノード変数データ」及び「木構造」を表示するための情報を含むものを意味する。

また，「ノード」は，「木構造」を前提とし，「木構造」は，数学の一分野であるグラフ理論に由来するところ，グラフ理論では，「グラフ」とは「いくつかの点とそれらを結ぶ何本かの線からなる図ないし構造」，「木」とは「閉路を含まないグラフ」，「閉路」とは「頂点を順につないで輪にしたもの」とそれぞれ定義される（甲２３の１）。

しかるところ，控訴人は，本件特許出願の当時，本件発明により表示される図形には閉路を持つグラフが含まれることを前提としていたが，単に「グラフ」という用語でクレームすると，これが閉路を持つ「グラフ」を意味するものとされ，「木」だけで表示されるケースは，より基本的な概念しか用いていないとして，クレームを充足しないものとされるのを懸念したため，「木構造」という用語を選択したものであるから，本件発明の「木構造の表示」の用語は，閉路を含むグラフを表示することを排除するものではない。したがって，構成要件Ｂの「ノード」は，ノード（頂点）と２つのノードを結ぶエッジ（辺）により構成されるグラフにおける頂点を意味するものであり，このグラフには，閉路を持つものも，閉路を持たないものも含まれる。

さらに，本件特許出願の願書に添付した明細書（以下，図面を含めて「本件明細書」という。甲３）の記載（【０００９】）に鑑みれば，構成要件Ｂの「文書ファイル」とは，テキストエディタ等で読むことのできるテキスト形式のファイルであり，１つの文書データを意味する。

（イ） 別紙被告プログラム説明書の記載によれば，被告プログラムの構成は，原判決別紙４被告プログラムの構成（原告）（以下「別紙４」という。）記載のとおりである。

（ウ） 被告プログラムは，ボックスを接続する形で本件ロボットのソフトウェアを実装することができる開発環境であり，別紙被告プログラム説明書及び別紙４に記載のとおり，コンピュータによって情報を管理するものであるから，構成要件Ａ（「コンピュータが情報を管理する情報管理方法」）を充足する。

次に，被告プログラムにおいて表示されるフローダイアグラムは，閉路を含まず，結合線により結合されたボックスで構成されているから，これらのボックスは，構成要件Ｂの「ノード」に該当し，各ボックスを識別するボックス識別番号（「ｉｄ」の番号）は，構成要件Ｂの「ノード識別情報」に該当する。また，被告プログラムにおいてロボットの動作に関する情報がまとめられた「文書ファイル」であるｂｅｈａｖｉｏｒ．ｘａｒ内で，ボックスに対応付けて入力されたスクリプトは，構成要件Ｂの「管理すべき情報」及び「ノードデータ」に該当する。そうすると，被告プログラムは，「ノードデータを含む文書ファイル」としてコンピュータが記憶する「情報記憶ステップ」を備えているから，構成要件Ｂを充足する。

さらに，被告プログラムは，「文書ファイル」であるｂｅｈａｖｉｏｒ．ｘａｒを読み込んで表示し，ｂｅｈａｖｉｏｒ．ｘａｒに含まれるスクリプトを実行するステップを備えているから，構成要件Ｃ及びＤを

充足する。

【被控訴人の主張】

（ア） 本件明細書の記載（【０００７】，【０００９】），構成要件Ｈ が「自ノード変数データの値を更新するステップ」と規定していることに照らせば，構成要件Ｂの「管理すべき情報」は，変数データの値を意味するから，「情報管理方法」における情報及び当該情報を含む「文書ファイル」（構成要件Ｂ）は，それ自体，多人数で共有し，再利用する価値のある，変数データの値を意味する。

また，構成要件Ｂの「複数のノードそれぞれに対応付けて入力された管理すべき情報」との文言によれば，構成要件Ｂの「ノード」は，管理すべき情報を含むものであり，木構造を前提とした概念である。

次に，本件明細書の記載（【００２９】）によれば，構成要件Ｂの「ノード識別情報」は，ノードを識別する情報であって，ノード生成時に自動的に一意の番号が付与される情報を意味する。そして，本件特許の特許請求の範囲の請求項２及び９の記載並びに本件明細書の記載（【００４２】，【００４９】）によれば，ノードの結合関係を表す参照リードは，構成要件Ｅの「親ノード識別情報」に基づく階層リードと区別されているから，構成要件Ｂの「ノード識別情報」は，ノードの結合関係を表す情報を含まないものを意味する。

（イ） 被告プログラムの構成が別紙４記載のとおりであるとの控訴人の主張は，争う。

別紙被告プログラム説明書の記載及び証拠（乙３ないし７，１９，２８ないし３１）によれば，被告プログラムの構成は，原判決別紙５被告プログラムの構成（被告）記載のとおりである。

（ウ） 被告プログラムが管理するｂｅｈａｖｉｏｒ．ｘａｒに含まれるのはロボットを動作させるためのプログラムコードであって，人間がこの

意味を読み取り，多人数で共有して再利用するものではないから，被告プログラムには「情報」に該当するものが存在しない。そうすると，被告プログラムは，本件ロボットを動作させるためのソフトウェアを開発するためのソフトウェア開発環境であり，構成要件Ａの「情報管理方法」であるとはいえず，構成要件Ｂの「管理すべき情報」，「ノード」及び「文書ファイル」を備えていない。

また，「木構造」は，階層構造を備えるものであるが，被告プログラムのフローダイアグラムはプロセスを規定するものであって，階層構造を備えていないから，被告プログラムにおけるボックスは，構成要件Ｂの「ノード」に該当するとはいえない。したがって，被告プログラムは，「ノードデータ」及び「情報記憶ステップ」を備えていない。

さらに，被告プログラムには，同一のｉｄを有する複数のボックスデータが存在するから，ｉｄによってボックスを識別することは不可能である上，ｉｄは編集可能であって，固有の識別情報ではなく，ボックス生成時に自動的に一意の番号が付与されるものではないから，ｉｄは構成要件Ｂの「ノード識別情報」に該当するとはいえない。したがって，被告プログラムは，構成要件Ｂの「ノード識別情報」を備えていない。

以上によれば，被告プログラムは，構成要件Ａ及びＢを充足しない。

そして　上記のとおり，被告プログラムは，構成要件Ｂの「情報記憶ステップ」，「文書ファイル」及び「ノードデータ」を備えていないから，これらの存在を前提とする構成要件Ｃ及びＤも充足しない。

イ　争点１－２（構成要件Ｅの充足性）について

【控訴人の主張】

（ア）　本件発明の構成要件Ｅの「親ノード」とは，あるノードに対して，当該ノードが属する順序系列内において，当該ノードに直近して先行するノードを指し，構成要件Ｅの「ルートノード」とは，当該ノードが属

する順序系列内における一番最初のノードを指し，「ルートノード」に対する「親ノード」は存在しない。

次に，前記アの【控訴人の主張】のとおり，本件発明の「木構造の表示」は，閉路を含むグラフの表示を排除するものではなく，構成要件Ｂの「ノード」は，ノード（頂点）とエッジ（辺）により構成されるグラフにおける頂点を意味するものである。

本件明細書の記載（【００４２】，【００４９】，【００５４】）によれば，親子のノード間を接続するリードは「階層リード」と呼ばれ，「階層リード」は親子関係に基づいて表示されるから，「親」は，専ら木構造の表示におけるノード間のリードの接続に関するものを意味する。

そして，構成要件Ｇの文言及び本件明細書の記載（【００３８】，【００６０】）によれば，「木構造」は階層ごとに表示されるから，構成要件Ｅの「親ノード識別情報」は，階層ごとの木構造表示のために「親ノード」を識別する機能を有する情報を意味する。

（イ） 被告プログラムでは，各ボックスに入力コネクタと出力コネクタが含まれており，コネクタ同士を線でつなげることによって，ボックス間に結合関係が生じ（別紙被告プログラム説明書第２の３(2)），各ボックスは，コネクタを介して結合されると，出力コネクタのある結合元のボックスから，入力コネクタのある結合先のボックスへと，結合ボックス同士の処理順序が指定される（別紙被告プログラム説明書第２の４ア）。コネクタを介したボックスの結合により各ボックスの処理順序が指定されることは，①ボックスの結合関係を無視した順序で各ボックスのスクリプト処理が行われることがないこと，②「Ｐｅｐｐｅｒプログラミング　基本動作からアプリの企画・演出まで」と題する解説書（甲７の２９頁）において，ボックスの結合線に従って「最初のボックスから次のボックスへと，処理が順番に移って」いくことを「順序実行」と呼んで

いることからも明らかである。

したがって，被告プログラムにおいては，各ボックスをコネクタを介して結合することによって，ボックス間に「階層関係」が生じ，先に処理される（出力コネクタのある）「親ボックス」と，後に処理される（入力コネクタのある）「子ボックス」の親子関係が成立する。

（ウ）a　原判決別紙３－２（以下「別紙３－２」という。甲１７）の本件ｂｅｈａｖｉｏｒ．ｘａｒ１をフローダイアグラムに表示した場合の接続関係は，次のとおりである。

(a)　各ボックスのｉｄは，次のとおりである。ボックス自体とは異なる各「内壁コネクタ」（左側内壁コネクタ及び右側内壁コネクタ）（原判決別紙６（以下「別紙６」という。）の図１及び図２参照）の「ｉｄ」は「０」である。

| Set Language | Localized Text | Say Text | Say |
|---|---|---|---|
| id="2"<br>（１８行） | id="5"<br>（１８０行） | id="2"<br>（１２５行） | id="1"<br>（９２行） |

(b)　別紙３－２のｒｏｏｔボックスタグ内の結合情報（３０１行ないし３０３行）及びＳａｙボックスタグ内の結合情報（２９２行ないし２９４行）は，次のとおりである。各結合情報の「inputowner=」及び「outputowner=」の次に記載された数字部分の「"1"」，「"2"」及び「"5"」は，括弧内の行（例えば，「１８０行」）において定義されたボックスの「ｉｄ」の番号を表す（例えば，「inputowner="5"」及び「outputowner="5"」の「"5"」は，「ｉｄ５」のＬｏｃａｌｉｚｅｄ　Ｔｅｘｔボックスを意味する。）。

【ｒｏｏｔボックスタグ内の結合情報】

301: <Link inputowner="1" indexofinput="2" outputowner="2"
indexofoutput="3" />

302: <Link inputowner="0" indexofinput="4" outputowner="1"
indexofoutput="4" />
303: <Link inputowner="2" indexofinput="2" outputowner="0"
indexofoutput="2" />

ｒｏｏｔボックスのフローダイアグラムは，①「ｉｄ０」（ｒｏｏｔボックスの左側内壁コネクタ）から「ｉｄ２」（Ｓｅｔ　Ｌａｎｇｕａｇｅボックス）へ（３０３行），②「ｉｄ２」（Ｓｅｔ　Ｌａｎｇｕａｇｅボックス）から「ｉｄ１」（Ｓａｙボックス）へ（３０１行），③「ｉｄ１」（Ｓａｙボックス）から「ｉｄ０」（ｒｏｏｔボックスの右側内壁コネクタ）へ（３０２行）と結合していることを表している。

【Ｓａｙボックスタグ内の結合情報】

292: <Link inputowner="0" indexofinput="4" outputowner="2"
indexofoutput="4" />
293: <Link inputowner="5" indexofinput="2" outputowner="0"
indexofoutput="2" />
294: <Link inputowner="2" indexofinput="2" outputowner="5"
indexofoutput="3" />

Ｓａｙボックスのフローダイアグラムは，①「ｉｄ０」（Ｓａｙボックスの左側内壁コネクタ）から「ｉｄ５」（Ｌｏｃａｌｉｚｅｄ　Ｔｅｘｔボックス）へ（２９３行），②「ｉｄ５」（Ｓｅｔ　Ｌａｎｇｕａｇｅボックス）から「ｉｄ２」（Ｓａｙ　Ｔｅｘｔボックス）へ（２９４行），③「ｉｄ２」（Ｓａｙ　Ｔｅｘｔボックス）から「ｉｄ０」（Ｓａｙボックスの右側内壁コネクタ）へ（２９２行）と結合していることを表している。

そして，被告プログラムにおいては，あるボックスの入力コネクタ

に結合されている出力コネクタ及び当該出力コネクタを有しているボックスの情報がｂｅｈａｖｉｏｒ．ｘａｒという文書ファイル上に記載されるから（別紙被告プログラム説明書第２の３(2)），ボックス及びコネクタ同士の結合情報が，ボックスデータの一部としてｂｅｈａｖｉｏｒ．ｘａｒに集積され，その結合情報によってボックス間の親子関係を読み取り，あるボックスに対する親ボックスを特定することができる。

したがって，上記結合情報のうち，ボックスの親ボックスを特定する情報（例えば，３０１行の「inputowner=″1″」に対する「outputowner=″2″」）は，構成要件Ｅの「親ノード」を特定する「親ノード識別情報」に該当する。

ｂ　また，被告プログラムにおいては，他のボックスの出力コネクタとの結合情報を持たないボックスが存在し，これは，当該ボックスが属する順序系列内における一番最初のボックスとなる。

したがって，被告プログラムにおけるボックスのうち，当該ボックスの入力コネクタと他のボックスの出力コネクタとの結合情報を含まない特定のボックスが，構成要件Ｅの「ルートノード」に該当する。

ｃ　以上によれば，被告プログラムは，構成要件Ｅを充足する。

(エ)　（当審における控訴人の追加主張）

ａ　別紙７－１（甲２５の１）記載のｂｅｈａｖｉｏｒ．ｘａｒ（以下「本件ｂｅｈａｖｉｏｒ．ｘａｒ２」という。）をフローダイアグラムに表示した場合の接続関係は，次のとおりである。

各ボックスのｉｄは，次のとおりである。ボックス自体とは異なる「内壁コネクタ」（左側内壁コネクタ及び右側内壁コネクタ）（別紙７－２（甲２５の２）参照）の「ｉｄ」は「０」である。

| Set Language | Localized Text | Say Text |
|---|---|---|
| id="2"<br>（１８行） | id="5"<br>（１４７行） | id="3"<br>（９２行） |

別紙７－１のｒｏｏｔボックスタグ内の結合情報（２５９行ないし２６２行）は，次のとおりである。各結合情報の「inputowner=」及び「outputowner=」の次に記載された数字部分の「"2"」，「"3"」及び「"5"」は，ボックスの「ｉｄ」の番号を表す。

259: <Link inputowner="3" indexofinput="2" outputowner="5"
indexofoutput="3" />

260: <Link inputowner="5" indexofinput="2" outputowner="2"
indexofoutput="3" />

261: <Link inputowner="2" indexofinput="2" outputowner="0"
indexofoutput="2" />

262: <Link inputowner="0" indexofinput="4" outputowner="3"
indexofoutput="4" />

上記結合情報は，①「ｉｄ０」（ｒｏｏｔボックスの左側内壁コネクタ）から「ｉｄ２」（Ｓｅｔ　Ｌａｎｇｕａｇｅボックス）へ（２６１行），②「ｉｄ２」（Ｓｅｔ　Ｌａｎｇｕａｇｅボックス）から「ｉｄ５」(Ｌｏｃａｌｉｚｅｄ　Ｔｅｘｔボックス)へ(２６０行)，③「ｉｄ５」（Ｌｏｃａｌｉｚｅｄ　Ｔｅｘｔボックス）から「ｉｄ３」（Ｓａｙ　Ｔｅｘｔボックス）へ（２５９行），④「ｉｄ３」（Ｓａｙ　Ｔｅｘｔボックス）から「ｉｄ０」（ｒｏｏｔボックスの右側内壁コネクタ）へ（２６２行）と結合していることを表している。

したがって，前記(ウ)ａと同様に，上記結合情報のうち，ボックスの親ボックスを特定する情報（例えば，２５９行の「inputowner="3"」に対する「outputowner="5"」）は，構成要件Ｅの「親ノード」を特定

する「親ノード識別情報」に該当する。

ｂ　また，前記(ウ)ｂと同様に，被告プログラムにおけるボックスのうち，当該ボックスの入力コネクタと他のボックスの出力コネクタとの結合情報を含まない特定のボックスが構成要件Ｅの「ルートノード」に該当する。

ｃ　以上によれば，被告プロクラムは，構成要件Ｅを充足する。

【被控訴人の主張】

(ア)　構成要件Ｅは，「親ノード識別情報」は，「ノードデータ」に含まれると規定している。本件明細書の記載（【００３６】，【００３８】ないし【００４２】，図２ないし図４，図９，図１４）によれば，構成要件Ｅの「ノードデータ」は，ノードを示す特定の開始タグと，そのエンドタグとの間に挟まれた領域内のデータを意味するから，構成要件Ｅの「親ノード識別情報」は，ノードを示す特定の開始タグと，そのエンドタグとの間に挟まれた領域内のデータに含まれるものを意味する。

また，本件明細書の記載（【００１０】，【００１３】，【００４９】，図６，図８）によれば，構成要件Ｅの「親ノード」は，木構造を前提とした概念であるところ，「木構造」は，基本となるルートノードから複数の要素に枝分かれをした階層構造を意味するから，「親」は，１つの基本となる要素から複数の要素に枝分かれをした階層構造によって規定される親子関係における親を意味する。

(イ)　被告プログラムのフローダイアグラムは，ｒｏｏｔボックス，Ｓｅｔ　Ｌａｎｇｕａｇｅボックス，Ｓａｙボックス及びｒｏｏｔボックスの実行順序の処理の流れを定めたものであり，ｒｏｏｔボックスから出てｒｏｏｔボックスに戻る閉路を構成しており，階層構造は存在しないから，「木構造」であるとはいえず，親子及び上下という概念は存在しない。したがって，被告プログラムは，「木構造」を前提とした概念で

ある構成要件Ｅの「親ノード」及び「ルートノード」を備えていない。

また，控訴人が構成要件Ｅの「親ノード識別情報」であると主張するＬｉｎｋタグ内の情報は，ボックス間の結合関係を表すものである上，各ボックスのＢｏｘタグの開始タグとエンドタグとに挟まれた領域の外にあり（例えば，本件ｂｅｈａｖｉｏｒ．ｘａｒ１のＳｅｔ　ｌａｎｇｕａｇｅボックス（１８行ないし９１行）に係るＬｉｎｋタグは，このボックスの領域外である３０１行ないし３０３行に存在する。），「ノードデータ」に含まれているといえないから，構成要件Ｅの「親ノード識別情報」に当たらない。

以上によれば，被告プログラムは，「親ノード」，「ルートノード」及び「親ノード識別情報」を備えていないから，構成要件Ｅを充足しない。

（ウ）　当審における控訴人の追加主張は，争う。

ウ　争点１－３（構成要件Ｆの充足性）について

【控訴人の主張】

（ア）　構成要件Ｆの文言及び本件明細書の記載（【００３１】，【００３２】）によれば，構成要件Ｆの「（直系）上位ノード」とは，順序系列内において，当該ノードに先行するノードを意味し，「上位」は，専ら下位ノードによるデータの承継に関するものを意味する。また，本件明細書の記載（【０００７】）によれば，「上位ノード変数データ」が表示されることによって，選択したノードで用いられている上位ノード変数データを容易に把握することができ，管理すべき情報の更新を，簡単かつ効率的に行うことができるのであるから，構成要件Ｆの「上位ノード変数データ」は，当該ノードの直系上位ノードのノードデータに含まれる変数データを意味する。そして，本件明細書の記載（【００５６】，【００６６】）によれば，「変数データ」は，変数名及び変数の値の２

つの要素からなるものであり，変数名のみによって特定することもできるものである。

次に，構成要件Ｆの「代入用スクリプト」には，単純な代入処理だけでなく，数式を含んだ代入処理を行うスクリプトも含まれる。

（イ） 被告プログラムにおいて，Ｓｅｔ　Ｌａｎｇｕａｇｅボックス，Ｓａｙボックス及びＬｏｃａｌｉｚｅｄ　Ｔｅｘｔボックスは，順序系列内において，Ｓａｙ　Ｔｅｘｔボックスに先行するボックスであるから（別紙被告プログラム説明書の図１１及び図２２），「（直系）上位ノード」を有し，その変数名又は変数の値は，「上位ノード変数データ」に該当する。

次に，別紙４記載の被告プログラムの構成ｆにおいては，被告プログラムのＬｏｃａｌｉｚｅｄ　Ｔｅｘｔボックスは，直系上位ノードであるＳｅｔ　Ｌａｎｇｕａｇｅボックスにおいて入力された，上位ノード変数データである変数ｌａｎｇの値である「Ｅｎｇｌｉｓｈ」や「Ｊａｐａｎｅｓｅ」を利用して演算を行い，Ｌｏｃａｌｉｚｅｄ　Ｔｅｘｔボックスの自ノード変数データである「Ｈｅｌｌｏ」や「こんにちは」を決定しているから，被告プログラムは，構成要件Ｆの「代入用スクリプト」を備えている。

また，別紙４記載の被告プログラムの構成ｆ’においては，被告プログラムのＳａｙ　Ｔｅｘｔボックスは，「親からの変数を取得」機能を使用する場合，直系上位ノードであるＳａｙボックスにおいて入力された上位ノード変数データである「Ｓｐｅｅｄ（%）」や「Ｖｏｉｃｅ　Ｓｈａｐｉｎｇ（%）」の値を利用して演算を行うことにより，Ｓａｙ　Ｔｅｘｔボックスの自ノード変数であるｓｅｎｔｅｎｃｅに代入し，Ｓａｙ　Ｔｅｘｔボックスの自ノード変数データであるｓｅｎｔｅｎｃｅの値を更新しているから，被告プログラムは，構成要件Ｆの「代入用ス

クリプト」を備えている。

以上のとおり，被告プログラムは，「（直系）上位ノード」，「上位ノード変数データ」及び「代入用スクリプト」を備えているから，構成要件Ｆを充足する。

【被控訴人の主張】

（ア） 構成要件Ｆの文言によれば，「上位ノード変数データ」は，「直系上位ノードのノードデータに含まれる」ものであるところ，「ノードデータ」は，ノードを示す特定の開始タグと，そのエンドタグとの間に挟まれた領域内のデータであるから，「上位ノード変数データ」は，直系上位ノードのノードを示す特定の開始タグと，そのエンドタグとの間に挟まれた領域内のデータに含まれる変数データを意味する。

また，本件発明は「木構造」を前提とするものであるから，構成要件Ｆの「直系上位ノード」も「木構造」を前提とした概念であるところ，「直系」とは，一般に，人と人との間の血統が親子の関係で続いている系統を意味するから，「直系上位ノード」は，木構造を前提として，自ノードと親子関係にあり，自ノードよりもルートノードに近いノードを意味する。

さらに，構成要件Ｆは，「代入用スクリプト」は，「上位ノード変数データを利用した演算」を行って「自ノード変数データの値を求める」ものである旨を規定しているところ，「上位ノード変数データ」の変数名だけでは，変数名を利用した演算を行って，自ノード変数データの値を求めることは不可能であり，本件明細書の記載（【００３１】，【００３２】）にも照らせば，「上位ノード変数データ」は，変数名のみならず，変数の値を意味するか，少なくとも変数の値を含むものと解釈される。

次に，構成要件Ｄは，「前記ノードデータに含まれるスクリプト」と

規定し，構成要件Ｆは，この記載を受けて「前記スクリプトは…自ノード変数データの値を求める代入用スクリプトを含んでおり」と規定しているから，構成要件Ｆの「代入用スクリプト」は，自ノードのノードデータに含まれるものでなければならない。また，本件明細書記載の実施例（【００７２】，【００７３】）に照らせば，構成要件Ｆの「代入用スクリプト」は，親ノード変数データの値を自ノード変数データの値として代入するスクリプトを意味する。

（イ） 被告プログラムのフローダイアグラムにおける処理の流れは，閉路を構成し，親子，上下という概念は存在しないから，被告プログラムは，構成要件Ｆの「（直系）上位ノード」及び「上位ノード変数データ」を利用した演算を行って自ノード変数データの値を求める「代入用スクリプト」を備えていない。

控訴人の被告プログラムの構成ｆに関する主張についてみると，変数ｌａｎｇは，Ｓｅｔ　Ｌａｎｇｕａｇｅボックスに含まれる一時変数（ソースコードの実行後にはメモリから破棄される値を一時的に格納するもの）であり，自ノードであるＳａｙボックス内に記載されたスクリプトには利用されておらず，また，ｌａｎｇの値はＳｅｔ　Ｌａｎｇｕａｇｅボックスに記憶されていない。さらに，被告プログラムでは，Ｌｏｃａｌｉｚｅｄ　Ｔｅｘｔボックスに予め記載された値である「Ｈｅｌｌｏ」や「こんにちは」という値を，Ｓｅｔ　Ｌａｎｇｕａｇｅボックスの変数データを用いて選択しているにすぎず，自ボックス変数に上位ボックス変数の値を代入していないから，構成ｆは，構成要件Ｆの「代入用スクリプト」に当たらない。

次に，控訴人の被告プログラムの構成ｆ’に関する主張についてみると，被告プログラムのＳａｙ　Ｔｅｘｔボックスにおける「Ｓｐｅｅｄ（％）」及び「Ｖｏｉｃｅ　Ｓｈａｐｉｎｇ（％）」は，いずれも上位

ノード変数ではなく，Ｓａｙ　Ｔｅｘｔボックスの自ボックス変数である。また，Ｓａｙ　Ｔｅｘｔボックスが「親からの継承」機能によって「Ｓｐｅｅｄ（%）」及び「Ｖｏｉｃｅ　Ｓｈａｐｉｎｇ（%）」の値を継承する際，Ｓａｙボックスはその継承元のボックスであるところ，ＳａｙボックスとＳａｙ　Ｔｅｘｔボックスは結合情報で結合されていないから，ＳａｙボックスはＳａｙ　Ｔｅｘｔボックスの上位ノードに該当しない。

さらに，「Ｓｐｅｅｄ(%)」及び「Ｖｏｉｃｅ　Ｓｈａｐｉｎｇ(%)」は，直系上位ノードのノードデータに含まれるものではなく，上位ノードのノードデータ中に変数の値を含むものではない。

加えて，本件ｂｅｈａｖｉｏｒ．ｘａｒ１には，「親からの継承」機能を実行するスクリプトは存在しないから，被告プログラムは，自ノードのノードデータに含まれる代入用スクリプトに利用され，代入用スクリプトが親ノード変数データの値を自ノード変数データの値として代入するスクリプトという構成が存在しない。

以上のとおり，被告プログラムは，「（直系）上位ノード」，「上位ノード変数データ」及び「代入用スクリプト」を備えていないから，構成要件Ｆを充足しない。

エ　争点１－４（構成要件Ｇの充足性）について

【控訴人の主張】

（ア）　構成要件Ｇの「木構造表示ステップ」について

ａ　本件発明の「木構造の表示」の用語は，閉路を含むグラフを表示することを排除するものではなく，構成要件Ｂの「ノード」は，ノード（頂点）と２つのノードを結ぶエッジ（辺）により構成されるグラフにおける頂点を意味するものであり，このグラフには，閉路を持つものも，閉路を持たないものも含まれることは，前記アの【控訴人の主

張】の主張のとおりである。

被告プログラムは，Ｌｉｎｋタグ内のｉｎｐｕｔｏｗｎｅｒ（入力側・子）とｏｕｔｐｕｔｏｗｎｅｒ（出力側・親）により結合情報を認識し，これに従ってフローダイアグラム上でボックスとボックス間の結合線で表現することによって「木構造」を表示している。また，Ｓａｙボックスのフローダイアグラムの表示において，左側の内壁コネクタと右側の内壁コネクタは別の図形(点)として表示されており，ここに閉路は存在しないから，「親ノード識別情報」を利用して「ノード」の「木構造」を表示しているといえる。

したがって，被告プログラムは，構成要件Ｇの「木構造表示ステップ」を備えている。

ｂ　この点に関し，原判決は，本件発明の「木構造」とは，ノードを表示するラベルとラベル間を接続する結合線であるリードから構成される図として表現される表示に関する概念であって，基本となる要素，すなわちルートから複数の要素に枝分かれをした階層構造を意味し，閉路を含まないものと解される，被告プログラムのＳａｙボックスのフローダイアグラムにおけるボックスの接続関係は，Ｓａｙボックスから出発してＳａｙボックスに戻る閉路として表示されており，「木構造」であるとはいえないから，被告プログラムは，「木構造」を表示するものとはいえず，構成要件Ｇの「木構造表示ステップ」を備えていない旨判断した。

しかしながら，前記ａのとおり，本件発明の「木構造の表示」の用語は，閉路を含むグラフを表示することを排除するものではなく，また，Ｓａｙボックスのフローダイアグラムの表示において，左側の内壁コネクタと右側の内壁コネクタは別の図形（点）として表示されており，ここに閉路は存在しない。

したがって，被告プログラムは，「親ノード識別情報」を利用して「ノード」の「木構造」を表示しているといえるから，原判決の上記判断は誤りである。

（イ） 構成要件Ｇの「ノードデータテーブル表示ステップ」について

ａ 本件明細書の記載（【００４６】）によれば，「デザインテーブル２０は，ツリービューア１０に表示されたノードのうちの選択されたノードが有する情報を表示する領域であ」ることからすると，構成要件Ｇの「ノードデータテーブル表示ステップ」にいう「テーブル」とは，情報を表示する領域を意味する。

被告プログラムのフローダイアグラム画面上でボックスを選択してダブルクリックすると，当該ボックスに対応して管理されているスクリプトを「スクリプトエディタ」によって表示することができるから（別紙被告プログラム説明書の図１４），被告プログラムは，「スクリプト」を表示する「ノードデータテーブル表示ステップ」を備えている。

ｂ 被告プログラムのフローダイアグラム画面上でボックスを選択してクリックすると，右下に区分けされたインスペクタという名称のウィンドウ（以下「インスペクタ」という。）に，①当該ボックスが有している入力・出力コネクタの名称，②当該ボックスの変数名が表示される。インスペクタ上のアイコンをクリックすると，当該ボックスのコネクタや変数の追加，コネクタの名称やタイプ等の編集，削除をすることができる。このインスペクタに表示された入力コネクタの名称は，スクリプトエディタにより表示されるスクリプトにも含まれている。

このうち，上記②の当該ボックスの変数名は，自ノード変数データを表示するものである（例えば，別紙被告プログラム説明書の図１５

及び図１６記載の「変数：Ｖｏｉｃｅ　Ｓｈａｐｉｎｇ（%）」）。

したがって，被告プログラムは，「自ノード変数データ」を表示する「ノードデータテーブル表示ステップ」を備えている。

ｃ　次に，入力コネクタには，アプリを実行した際に，直近の親ボックスから引き渡される値が，当該コネクタに対応した引数へ書き込まれる。例えば，被告プログラムでは，Ｓａｙ　Ｔｅｘｔボックスの変数であるｐは，Ｓａｙ　Ｔｅｘｔボックスの入力コネクタ”ｏｎＩｎｐｕｔ＿ｏｎＳｔａｒｔ（）”という関数（別紙３－２の１５０行）の構成要素であり，直系上位ノードであるＬｏｃａｌｉｚｅｄ　Ｔｅｘｔボックスの出力コネクタｏｎＳｔｏｐｐｅｄ（Ｓｅｌｆ，ｓｅｎｔｅｎｃｅｓ[ｓＤｅｆａｕｌｔＬａｎｇ]）（別紙３－２の２１９行）から出力された値を受け取る，コネクタの内部領域である。

そして，入力コネクタとは，親ボックスから引き渡される値を記憶する変数が図形化されたものであり，入力コネクタの名称が構成要件Ｇにおける「上位ノード変数データ」に該当する。この入力コネクタの名称は，インスペクタに表示される。

したがって，インスペクタ及びスクリプトエディタにおける入力コネクタの名称に関する情報の表示は，上位ノード変数データを表示するものであるから，被告プログラムは，「上位ノード変数データ」を表示する「ノードデータテーブル表示ステップ」を備えている。

ｄ　被告プログラムの構成ｇ’に関し，被告プログラムのＳａｙ　Ｔｅｘｔボックスの「スクリプトエディタ」において「親からの変数を取得」機能を使う場合，上位ノードであるＳａｙボックスの変数から利用可能なものを一覧表示する機能がある。

したがって，被告プログラムは，「上位ノード変数データ」を表示する「ノードデータテーブル表示ステップ」を備えている。なお，本

件発明において，インスペクタとスクリプトエディタを同時に表示する必要はないが，被告プログラムにおいては，スクリプトエディタをインスペクタと同時に表示することも可能である。

ｅ　以上によれば，被告プログラムは，構成要件Ｇの「自ノード変数データ」，「上位ノード変数データ」及び「スクリプト」を表示する「ノードデータテーブル表示ステップ」を備えている。

ｆ　この点に関し原判決は，①構成要件Ｇの「ノードデータテーブル表示ステップ」にいう「ノードデータテーブル」とは，「ノードデータ」の一覧表であり，上位ノード変数データ，自ノード変数データ及び代入用スクリプトを同時に表示するものと解される，②本件明細書の【００３２】の記載から，「ノードデータテーブル」が表示する「ノード変数データ」は，変数の値を意味すると解されるとした上で，③被告プログラムの構成ｇについて被告プログラムのフローダイアグラム画面上のインスペクタに表示された入力コネクタの名称は変数の値ではないから，「上位ノード変数データ」に当たらず，また，被告プログラムの構成ｇ’についてインスペクタ上にＳａｙ　Ｔｅｘｔボックスの変数Ｓｐｅｅｄ（％）の値が表示されるが（別紙被告プログラム説明書の図１９），これはＳａｙ　Ｔｅｘｔボックスにおいて表示されるものであり，自ノード変数を表示しているものと認められ，「上位ノード変数データ」を表示しているとみることはできないから，被告プログラムは，一覧表として「自ノード変数データ」及び「上位ノード変数データ」を同時に表示しているということはできない，④さらに，「親からの継承」の機能に関して，本件ｂｅｈａｖｉｏｒ．ｘａｒ１内に，自ノード変数データ及び上位ノード変数データを利用した演算を行って自ノード変数データの値を求める「代入用スクリプト」があると認めるに足りる証拠はないとして，被告プログラムは，構成

要件Ｇの「自ノード変数データ，前記上位ノード変数データ及び前記スクリプトを表示するノードデータテーブル表示ステップ」を備えていない旨判断した。

しかしながら，本件明細書の記載（【００４６】，【００７９】，図６）によれば，本件明細書では，図６の「デザインテーブル」に示すように，代入用スクリプト表示領域，生成用スクリプト表示領域，操作ボタン表示領域の表示は表形式というよりは，個々の情報をまとめて表示する領域という意味で「テーブル」という用語が使われており，その領域で表示すべき情報を，画面上でひとまとめに配置するか，分割して配置するかは，設計事項にすぎない。加えて，本件発明の技術的思想に照らせば，「ノードデータテーブル表示ステップ」は，表示された木構造の個々のノードに対応付けられた詳細情報を簡単に表示することができること（【０００９】）により，文書ファイル（プログラム）の編集を容易にするためのものであるから，一覧表でなければならない等との制約を付けて解釈されるべきではない。

また，本件明細書の【００３２】における「変数の値（「変数データ」と記述する場合もある。）」との記載は，「変数データ」という用語を，文脈によって，変数の値を指す意味で用いることもあるという注意書きであると理解できること，「変数データ」は，変数名と変数の型を意味するというのが，プログラミングに関する通常の用語であること（甲２４），実質的にも，本件発明が「ノードデータテーブル表示ステップ」において上位ノード変数データを表示させる目的は，表示された木構造の個々のノードに対応付けられた詳細情報を簡単に表示することができる（【０００９】）ことにより，文書ファイル（プログラム）の編集を容易にする点にあり，変数名が分かれば，その目的を達成することができることからすると，本件発明の「変数データ」

は，本件明細書において文脈上変数の値を意味すべき場合を除き，変数名を指すと解すべきである。

さらに，別紙３の２の「１５３行」では，「self.getParameter(″Speed（%)″)」は，ＳａｙボックスのＳｐｅｅｄ（％）の値を代入し，これと自ノード変数データを組み合わせて発話に関する変数ｓｅｎｔｅｎｃｅを求めているから，「代入用スクリプト」に該当する。

したがって，原判決の上記判断は，その前提を欠くものであって，誤りである。

（ウ）　小括

前記(ア)及び(イ)によれば，被告プログラムにおける「情報表示ステップ」は，「木構造表示ステップ」と「ノードデータテーブル表示ステップ」を含むものといえるから，構成要件Ｇを充足する。

【被控訴人の主張】

（ア）　前記イの【被控訴人の主張】のとおり，「木構造」は，基本となるルートノードから複数の要素に枝分かれをした階層構造を意味するところ，被告プログラムのフローダイアグラムは，ｒｏｏｔボックス，Ｓｅｔ　Ｌａｕｇｕａｇｅボックス，Ｓａｙボックス，ｒｏｏｔボックスという順序の処理の流れを定めており，ｒｏｏｔボックスから出てｒｏｏｔボックスに戻る閉路を構成しているから，階層構造は存在せず，親子及び上下という概念は存在しない。

したがって，被告プログラムは，「木構造」を備えていないから，構成要件Ｇの「木構造表示ステップ」を備えていない。

（イ）ａ　「テーブル」は表を意味すること，本件明細書の記載（【００６５】，【００６６】，【００５７】，図６，図９，図１０，図１３）によれば，構成要件Ｇの「ノードデータテーブル表示ステップ」は，変数名と変数の値とを表形式で表示するステップを意味し，また，自

ノード変数データ，上位ノード変数データ並びに当該自ノード変数データ及び上位ノード変数データを用いた代入用スクリプトを，全て同時に表示するものを意味する。

b　この点に関し，控訴人は，控訴人主張の被告プログラムの構成ｇに関し，インスペクタに表示される入力コネクタの名称が「上位ノード変数データ」に該当する旨主張するが，入力コネクタの名称は，コネクタ名であって変数ではない上，変数の値を含まないから，構成要件Ｇの「上位ノード変数データ」に当たらない。

また，控訴人主張の被告プログラムの構成 g’は，Ｓａｙ　Ｔｅｘｔボックスのスクリプトエディタに表示される「ｐ」は，前記ウの【被控訴人の主張】のとおり，自ボックス関数の引数であって上位ノード変数ではなく，一時変数であり，ｐの値はＳａｙ　Ｔｅｘｔボックスに記憶されておらず，被告プログラムにおいてｐの値を表示することはできないから，構成要件Ｇの「上位ノード変数データ」に当たらない。

そして， 被告プログラムにおいて，「親からの変数を取得」機能によって表示されるのは，変数名にとどまり，変数の値は表示されないから，「親からの変数を取得」機能による表示は「上位ノード変数データ」に当たらない。

したがって，被告プログラムは，「上位ノード変数データ」を表示する「ノードデータテーブル表示ステップ」を備えていない。

c　以上によれば，被告プログラムは，構成要件Ｇの「自ノード変数データ」，「上位ノード変数データ」及び「スクリプト」を表示する「ノードデータテーブル表示ステップ」を備えていない。

（ウ）　前記(ア)及び(イ)によれば，被告プログラムは，「木構造表示ステップ」及び「ノードデータテーブル表示ステップ」をいずれも備えてい

ないから，構成要件Ｇを充足しない。

オ　争点１－５（構成要件Ｈ及びＩの充足性）について

【控訴人の主張】

（ア）　「更新」とは，文書ファイルを変更するか否かに関わらず，自ノード変数データの値を更新することを意味し，構成要件Ｈの「更新するステップ」は，このような意味での「更新」を行うステップを意味する。

被告プログラムでは，Ｓａｙ　Ｔｅｘｔボックスの自ノード変数であるｓｅｎｔｅｎｃｅの値が，「代入用スクリプト」の実行によって導かれるところ，ｓｅｎｔｅｎｃｅの値は，直系上位ノードであるＳａｙボックスの「Ｓｐｅｅｄ（％）」及び「Ｖｏｉｃｅ　Ｓｈａｐｉｎｇ（％）」の値の設定や，Ｓｅｔ　Ｌａｎｇｕａｇｅボックスで設定した言語，Ｌｏｃａｌｉｚｅｄ　Ｔｅｘｔボックスで言語に対応して入力されるあいさつ文に応じて更新されるから，被告プログラムは，構成要件Ｈの「更新するステップ」を備えており，構成要件Ｈを充足する。

（イ）　以上によれば，被告プログラムは，構成要件ＡないしＨを充足するから，これを前提とする構成要件Ｉを充足する。

したがって，被告プログラムは，本件発明の構成要件を全て充足するから，その技術的範囲に属する。

【被控訴人の主張】

（ア）　構成要件Ｂ，Ｃ及びＧの文言並びに本件明細書の記載（【００４３】，【００６４】，【００６７】，【００７３】，図１０）に照らせば，構成要件Ｈの「更新ステップ」においては，文書ファイルに含まれる自ノード変数データが変更される必要がある。

しかしながら，被告プログラムにおいては，スクリプトの実行によりｂｅｈａｖｉｏｒ．ｘａｒの値が変更されることはないから，被告プログラムは，「更新するステップ」を備えておらず，構成要件Ｈを充足し

ない。

（イ） 以上によれば，被告プログラムは，構成要件ＡないしＨをいずれも充足しないから，これを前提とする構成要件Ｉを充足しない。

したがって，被告プログラムは，本件発明の技術的範囲に属さない。

⑵ 争点２（無効の抗弁の成否）について

以下のとおり訂正するほか，原判決の「事実及び理由」の第２の４⑵記載のとおりであるから，これを引用する。

ア 原判決１８頁１７行目から１８行目までを「ア 争点２－１（乙９発明を主引用例とする本件発明の新規性又は進歩性の欠如）について」と改める。

イ 原判決１８頁２１行目の「出願時」を「本件特許出願の優先日当時」と改める。

ウ 原判決１９頁１２行目の「乙９発明の公報」を「乙９」と，同頁１３行目から１４行目にかけての「実質的相違点ではないから，新規性を欠く。」を「実質的相違点ではない。したがって，本件発明は，新規性を欠く。」と改める。

エ 原判決１９頁１８行目の「設計事項にすぎないから，」の次に「本件発明は，」を加える。

オ 原判決１９頁２３行目から２４行目にかけての「相違点について」を「相違点に係る本件発明の構成が」と，同行目の「よって」から２５行目末尾までを「したがって，被控訴人の主張は理由がない。」と改める。

カ 原判決１９頁末行から２０頁１行目までを「イ 争点２－２（乙１６発明を主引用例とする本件発明の新規性又は進歩性の欠如）について」と改める。

キ 原判決２０頁４行目の「出願時」を「本件特許出願の優先日当時」と，同頁１０行目の「相違点が既存の」を「相違点に係る本件発明の構成が」

と改める。

ク　原判決２１頁９行目及び１３行目の各「本件明細書等」を「本件明細書」と改める。

⑶　争点３（損害の発生の有無及びその額）について

以下のとおり訂正するほか，原判決の「事実及び理由」の第２の４⑶記載のとおりであるから，これを引用する。

ア　原判決２１頁１９行目，２０行目及び２３行目の各「売上」を「売上げ」と改める。

イ　原判決２１頁２３行目の「本件特許権」を「本件発明」と，同頁２４行目の「本件におけるライセンス料率」を「本件発明の実施に対して受けるべき金銭の額に相当する損害額を算定するに当たっての実施料率」と改める。

ウ　原判決２２頁１行目の「不法行為」を「本件特許侵害の不法行為」と，同頁２行目の「本件特許権のライセンス料（特許法１０２条３項）に相当する損害額」を「本件発明の実施に対して受けるべき金銭の額に相当する損害額（特許法１０２条３項）」と改める。

第３　当裁判所の判断

当裁判所も，控訴人の請求は理由がないものと判断する。その理由は，以下のとおりである。

１　本件明細書の記載事項

⑴　本件明細書（甲３）の「発明の詳細な説明」には，次のような記載がある（下記記載中に引用する図１ないし図４，図６，図１０は，別紙明細書図面のとおりである。）。

ア　【技術分野】

【０００１】

本発明は，コンピュータを用いて情報を管理する情報管理方法，情報管

理プログラム，及び情報管理装置に関する。

【背景技術】

【０００２】

コンピュータを用いて各種情報の管理を行う場合，それぞれの情報を記憶したファイル（文書ファイル，画像ファイル等）を，所定のフォルダに保管することによって行うのが一般的である。しかし，作成したフォルダの構造及びそれぞれのフォルダに保管するファイルの種類等は，任意であってフォルダの作成者に依存するため，作成者以外の者が必要な情報に適確にアクセスすることは，必ずしも簡単ではない。すなわち，多数の者が情報を共有化し，再利用できるように，情報管理を行うことは容易ではない。

【０００３】

特許文献１には，情報の共有化，再利用を効率よく実現することができる文書情報管理システムが記載されている。この文書情報管理システムは，案件（プロジェクト）毎にツリーを作成して表示し，作成した文書ファイルを，表示されたツリーの任意のノードに付随させて，サーバコンピュータに保管するものである。

【０００４】

また，異なる計算機やアプリケーションで共通に取扱うことができるデータ形式として，ＸＭＬ（Extensible Markup Language）等の構造化文書規格があるが，特許文献２には，このような構造化文書を木構造として捉えて処理する構造化文書処理システムが記載されている。

【０００５】

しかし，管理される各種情報の更新については，上記した管理システムにおいても，効率化が十分図られているとはいえない。すなわち，木構造のノードに含まれる情報は，相互に関連するものが多いが，上記した管理

システムにおいては，それぞれの文書の該当する部分を個別に更新する必要があり，十分効率的とはいえない。

【０００７】

本発明は，上記事情に鑑みなされたもので，管理すべき情報の更新を，簡単かつ効率的に行うことができる情報管理情報を提供することを目的とする。

イ　【課題を解決するための手段】

【０００８】

本発明の情報管理方法は，コンピュータが情報を管理する情報管理方法であって，前記コンピュータに複数のノードそれぞれに対応付けて入力された管理すべき情報を，前記ノードを識別するノード識別情報に対応付けられた複数のノードデータを含む文書ファイルとして前記コンピュータが記憶する情報記憶ステップと，前記情報記憶ステップで記憶された前記文書ファイルの情報を前記コンピュータが表示する情報表示ステップと，前記ノードデータに含まれるスクリプトを前記コンピュータが実行する情報評価ステップとを備え，前記ノードデータは，ルートノードを除いて，当該ノードの親ノードを特定する親ノード識別情報を含んでおり，前記スクリプトは，当該ノードデータに含まれる変数データである自ノード変数データと，当該ノードの直系上位ノードのノードデータに含まれる変数データである上位ノード変数データを利用した演算を行って，前記自ノード変数データの値を求める代入用スクリプトを含んでおり，前記情報表示ステップは，前記親ノード識別情報を利用して，前記ノードの木構造を表示する木構造表示ステップと，前記表示された木構造のノードのうちの選択されたノードの前記自ノード変数データ，前記上位ノード変数データ及び前記スクリプトを表示するノードデータテーブル表示ステップを含み，

前記情報評価ステップは，前記代入用スクリプトの実行により，前記自

ノード変数データの値を更新するステップを含むものである。

【０００９】

本発明によれば，利用者が入力したデータに含まれるスクリプトを利用して，ノードデータを更新することができるので，管理すべき情報の更新を，簡単かつ効率的に行うことができる。また，複数のノードデータを含む１つの文書データを用いて，個々の業務や案件に関する情報を管理しているので，多数の利用者による情報の共有化，再利用を，簡単かつ効率的に行うことができるとともに，文書データに基づいて，簡単にノードの木構造を表示させることができ，業務や案件全体の把握を簡単に行うことができる。さらに，表示された木構造の個々のノードに対応付けられた詳細情報を簡単に表示することができる。

【００１０】

本発明の情報管理方法は，前記木構造表示ステップが，前記ノードを示すノードラベルと，前記親ノードの前記ノードラベルとの間を接続する階層リードとの表示を含むものを含む。本発明によれば，ノードの階層関係を容易に識別することができる。

【００１３】

本発明の情報管理方法は，前記ルートノードの前記ノードデータが，前記ルートノードが形成するページを識別する自己ページ番号を含んでおり，前記ルートノードを除くノードの前記ノードデータが，当該ノードが所属する所属ページを識別する所属ページ番号を含むとともに，当該ノードが形成する自己ページを識別する自己ページ番号を含むことが可能であり，前記木構造表示ステップが，前記自己ページ番号及び前記所属ページ番号に基づいて，前記ルートノードを先頭とする木構造を表示するとともに，前記自己ページ番号を有する前記ノードを先頭とする木構造を，異なるページに表示可能であるものを含む。本発明によれば，大きな木構造

でも効率よく表示することができ，管理すべき情報の全体構成を容易に把握することができる。また，特定のノードから別の木構造を作成することができるので，別の観点の木構造を簡単に作成することができ，管理すべき情報の整理が容易になる。

【００１５】

本発明の情報管理方法は，前記ノードデータテーブル表示ステップが，当該ノードの直系下位ノードの前記スクリプトでも利用される公開変数と，当該ノードの前記スクリプトでのみ利用される限定変数を含むものを含む。本発明によれば，必要な変数データのみを下位ノードに継承させることができる。

【００２２】

本発明の情報管理プログラムは，上記した情報管理方法における各ステップを，コンピュータに実行させるための情報管理プログラムである。

【発明の効果】

【００２４】

以上の説明から明らかなように，本発明によれば，管理すべき情報の更新を，簡単かつ効率的に行うことができる。

ウ　【００２７】

コンピュータにインストールする情報管理プログラムは，管理すべき情報をノードに対応付けて入力する情報入力ステップ，情報入力ステップで入力されたデータを，各ノードを識別するノード識別情報に対応付けられた複数のノードデータを含む文書として記憶する情報記憶ステップ，情報記憶ステップで記憶された文書の情報を表示する情報表示ステップ，及びノードデータに含まれるスクリプトを実行する情報評価ステップを実行するためのプログラムを含んでいる。

【００２８】

図１に，ノードデータとして記憶される情報の一例を示す。記憶される情報は，ノード番号，ページ番号，親ノード番号，ノードラベル，ノード表示属性情報，変数情報，代入用スクリプト，生成用スクリプト，リンク情報を含む。

【００２９】

ノード番号は，ノードを識別する情報であり，ノード生成時に自動的に一意の番号が付与される。ページ番号は，文書に含まれるノードを複数の木構造として表示するためのもので，そのノードが所属するページを識別する所属ページ番号に，そのノードが別のページを形成する場合にそのページを識別する自己ページ番号を含む。したがって，両方のページ番号が記憶されているノードは，２つのページに属することになる。親ノード番号は，そのノードの親ノードを識別する番号であり，ノード生成時に親ノードを指定することにより，その指定された親ノードのノード番号が自動的に記憶される。

【００３０】

ノードラベルは，木構造表示時にそのノードを示す情報であり，ノード名称等任意の情報を記憶することができる。ノード表示属性情報は，ノード表示時の背景，文字の色，フォント等の文字属性，枠の形状，大きさ等の枠属性等を指定する情報である。ここで，ＪＰＧ画像等をノードに表示したい場合は,その画像ファイルの場所を指定するＵＲＬ等が記憶される。

【００３１】

変数情報は，各ノードが保持するデータであって，変数名に対応させて記憶される。記憶される変数は，下位ノードから参照される公開変数と，自ノード内でのみ使用する限定変数を含む。また，変数の値（「変数データ」と記述する場合もある。）は，固定値が設定されても，スクリプトの実行によって演算された値が設定されてもよい。また，ＵＲＬが設定され

てもよい。どのような値が設定されるかは任意である。

【００３２】

代入用スクリプトは,自ノードの変数の値を演算するためのものである。代入用スクリプトは，自ノードの変数の値である自ノード変数データと，そのノードの直系上位ノードの公開変数の値である上位ノード変数データを利用して記述することが可能である。

【００３３】

生成用スクリプトは，辞書に登録してある別のノードやノード群（木構造の複数ノード）を利用して，新規にそのノードの下位のノードを生成するものである。生成用スクリプトを条件文と併用することにより，代入用スクリプトの実行により求められた変数データの結果によって，子ノードや孫ノードを生成することができる。生成用スクリプトを利用することにより，例えば，部品管理を行う場合，親部品のサイズによって子部品が変わるケースの子部品のデータを簡単に生成することができる。

【００３４】

なお,代入用スクリプト及び生成用スクリプトに使用する言語としては,スクリプト言語として使用されている任意の言語を使用することができる。

【００３５】

リンク情報は，各ノードにリンクするファイルに関する情報である。スタンドアローン型のコンピュータで実施する場合，この情報はリンクファイルのインデックス情報である。また，クライアント- サーバ型のコンピュータで実施する場合，リンクファイルをサーバに転送後，インデックス情報を作成し，記憶する。リンク情報を記憶することにより，各ノードをフォルダとして利用することが可能となる。

【００３６】

ノードデータは，例えばタグ付き文書情報として記憶される。図２に，

ノードデータの一例を示す。図２のデータは，ルートノードのノードデータの例であり，ノード番号(nodeNo)が「3450」，自己ページ番号(ownPageNo)が「10」，ノードラベル(label)が「パッセル操作マニュアル」である。所属ページ番号を示す（belongPageNo）が「0」，親ノード番号を示す（parentNodeNo）が「0」であることで，ルートノードであることを示している。図２の「x=”100”」から「color=”0”」までは，ノードの表示位置等のノード表示属性情報である。

【００３７】

この形式では，変数情報が，「<DataDivision>」と「</DataDivision>」の間に挿入され，代入用スクリプトが，「<ProcesureDivision>」と「</ProcesureDivision>」の間に挿入され，生成用スクリプトが，「<GenerateDivision>」と「</GenerateDivision>」の間に，リンク情報が，「<LinkageDivision>」と「</LinkageDivision>」の間に挿入される。ただし，図２の例では，変数情報，代入用スクリプト，生成用スクリプトは，記憶されていない。

【００３８】

図３に，ノードデータの別の例を示す。図３のデータは，ルートノード以外のノードデータの例である。所属ページ番号が「3484」，親ノード番号が「3488」となっており，ルートノード以外のノードのノードデータであることが把握できる。また，自己ページ番号が「3526」となっていることから，別ページの木構造の先頭ノードであることも把握できる。

【００３９】

図４に，管理すべきデータを複数のノードデータを含む文書情報として記憶させたものの一例を示す。図４の文書は，ヘッダ部４０，ノードデータ部４１ａ～４１ｎ，ライン部４２，レポート部４３を備える。

【００４０】

ヘッダ部４０は，管理される案件等を指すプロジェクトのプロジェクト番号，名称（プロジェクト名）等を示す情報を含んでいる。図４の例では，プロジェクト名が「Manual ver2」，プロジェクト番号が「10」であることを示している。

【００４１】

ノードデータ部４１ａは，ルートノードのノードデータを示し，ノードデータ部４１ｂ～４１ｎは，ルートノード以外のノードのノードデータである。

【００４２】

ライン部４２は，ノード間を接続するリードを定義する情報が記憶される領域である。ノード間を接続するリードは，親子のノード間を接続する階層リードと，階層関係とは無関係に一時的に変数を参照する参照元ノードと参照先ノード間を接続する参照リードがあるが，ライン部４２は，参照リードの存在及び，位置，表示属性等を規定する。

エ　【００４３】

次に，記憶された文書情報の表示について説明する。図５に，文書情報の表示を行う場合の概略動作フローを示し，図４に示す文書の表示を行った場合の表示画面の例を図６に示す。

【００４４】

図６の表示画面は，ツリービューア１０とデザインテーブル２０を有する。ツリービューア１０は，ノードの木構造を表示する領域であり，情報管理時の各種操作を行うためのプルダウンメニュー，及びポップアップメニューを表示する領域も兼ねる。ノードの木構造の表示は，ラベルとリードの表示によって行い，図６の例では，ルートノードのラベル表示１１ａとルートノード以外のノードのラベル表示１１ｂ，１１ｃ，１１ｄと，それらの間を接続する階層リード１２ｂ，１２ｃ，１２ｄが表示されている。

【００４５】

ラベル表示１１ａ～１１ｄは，ノード表示属性情報に基づいて表示されるが，ルートノードのラベル表示１１ａには，ルートノードであることを示すマーク１３ａが付加される。また，ルートノード以外のノードで自己ページ番号を有するノードは，その旨を示すマーク１４ｂ，１４ｄが付加される。マーク１４ｂ，１４ｄが付加されていることで，別ページの木構造が存在することを簡単に認識することができる。

【００４６】

デザインテーブル２０は，ツリービューア１０に表示されたノードのうちの選択されたノードが有する情報を表示する領域であり，公開変数表示領域２１，限定変数表示領域２２，代入用スクリプト表示領域２３，生成用スクリプト表示領域２４，操作ボタン表示領域２１ａ，２２ａ，２０ａを有する。操作ボタン表示領域２１ａの各操作ボタンは，公開変数に対する各種操作を行うためのものであり，操作ボタン表示領域２２ａの各操作ボタンは，限定変数に対する各種操作を行うためのものである。また，操作ボタン表示領域２０ａの各操作ボタンは，デザインテーブル２０に関する各種操作を行うためのものである。

【００４７】

プロジェクト毎に作成された文書ファイルを選択し，開くと，図５に示す手順で表示処理が行われる。ステップＳ１０１では，文書データからルートノードを認識する。既述のように，ルートノードのノードデータは，所属ページ番号及び親ページ番号が「０」であるので，そのようなノードを探すことにより，ルートノードを認識することができる。なお，文書番号を一義的に割り当て，ルートノードの自己ページ番号を文書番号と一致させておくと，ルートノードの認識がさらに簡単になる。

【００４８】

ステップＳ１０２では，認識したルートノードの自己ノード番号を認識し，ステップＳ１０３では，認識したページ番号のページに属するノードを認識する。すなわち，ルートノードの自己ページ番号を所属ページ番号として有するノードを認識する。

【００４９】

次いで，ステップＳ１０３で認識したノードのラベル表示を行う（ステップＳ１０４）。ラベル表示は，そのノードのノードラベル及びノード表示属性情報に基づいて表示する。そして，表示したノードの親子関係に基づいて階層リードを表示し，さらに，文書のライン部４２の情報を参考にして，参照リードを表示する（ステップＳ１０５）。

【００５０】

この状態では，ツリービューア１０に木構造が表示された状態である。ステップＳ１０６では，ツリービューア１０に表示されたノードが選択されたかどうかを判定し，選択されている場合，デザインテーブル２０の表示を行い（ステップＳ１０７），そのノードの変数，スクリプト等を表示する（ステップＳ１０８）。図６は，ルートノードを選択した場合の例であり，ルートノードには，変数情報等が記憶されていないので，デザインテーブル２０に，データの表示はない。この状態で，別のノードを選択すると，そのノードの変数等が表示される。

【００５１】

記憶された変数の値が，ＵＲＬ又はファイルパスである場合は，その変数を選択した状態で領域２１ａ，２２ａの「実行」ボタンをおすことにより，そのＵＲＬ又はファイルパスの内容を表示する。

【００５２】

デザインテーブル２０は，ノードを選択することで表示されるが，各ノードのリンク情報，及びノードレポートの表示は，ノード選択後メニュー

を表示させて行う。リンク情報表示が指示されると，そのノードのノードデータのリンク情報のリストを別ウィンドウで表示する。リンク情報が記憶されていない場合は，リストが空欄である。表示されたリストの中のファイルが選択されると，そのファイルの内容に応じた情報の表示を行う。ノードレポート表示が指示されると，別ウィンドウに表示するとともに，レポート領域４３を参照し,該当ノードの情報が記憶されている場合には,その情報を表示した別のウィンドウに表示する。

【００５３】

ノードのリンク情報，及びノードレポートは，リンク情報のリスト，あるいはレポート表示ウィンドウを表示した状態で，追加，削除を行うことができる。

オ　【００５４】

図６に示した状態で，表示された木構造及びノードデータの編集が可能であり，編集操作に対応した表示を行う（ステップＳ１０９）。ツリービューア１０上では，ノードの追加，削除，表示位置移動，表示属性変更，ノードラベル変更等を，プルダウンメニュー，ポップアップメニューの設定により行う。例えば，表示位置の変更は，ラベル表示をドラッグすることによって行い，ノードラベル及び表示属性の変更は，変更用のウィンドウを表示させて行う。また，ノードの削除は，削除したいノードを選択した状態で，メニューを表示させて削除する。ノードの追加は，メニューで追加モードに設定後，追加したいノードの親となるノードを選択し，そのままドラッグすることにより，新規ノードを生成する。また，ノードの繋ぎ換えは，繋ぎ換えたいノードを選択し，メニューを表示させて「ノード繋ぎ換え」を選択し，変更したい繋ぎ先のノード（親ノードとしたいノード）を選択することによって行う。生成されたノードのノードラベル，表示属性情報は，修正用のウィンドウを表示させて設定する。

【００５５】

それぞれのノードに関するデータは，ノードデータとして１まとまりになっているので，これらの編集を行った場合でも，編集を行ったノードのノードデータに反映させるだけでよく，軽い処理負担で編集作業を行うことができる。

【００５６】

デザインテーブルの情報を追加，修正する場合は，領域２１ａ，２２ａ，２０ａの該当するボタンを押すことによって行う。変数の追加は，領域２１ａ又は２２ａ「追加」ボタンを押して，新規の変数を生成し，変数名，値，修飾情報を入力する。変数の入力は，別ウィンドウで入力フォームを表示させて行ってもよい。

【００５７】

代入用スクリプト及び生成用スクリプトは，公開変数領域２１および限定変数領域２２に表示された変数を利用して作成する。公開変数領域２１には，自ノードの公開変数だけでなく，直系上位のすべてのノードの公開変数が表示される。直系上位のノード以外のノードの変数を参照したい場合は，参照リードを生成して，参照先のノードと関連付けておく。代入用スクリプト及び生成用スクリプトは代入用スクリプト領域２３及び生成用スクリプト領域２４に直接入力してもよいし，別のウィンドウを開いて入力するようにしてもよい。

【００５８】

なお，デザインテーブル２０の編集内容は，領域２０ａの更新ボタンを押すことによって，文書情報に反映される。

【００５９】

続いて，ステップＳ１１０では，表示終了，すなわち文書クローズが指示されたかどうかを判定し，表示終了が支持されていない場合は，階層移

動指示がされているか否かを判定する（ステップＳ１１１）。階層移動は，マーク１４ｂ，１４ｄ等別の木構造の先頭ノードとなるノードを選択した状態で，ポップアップメニューから指示する。

【００６０】

階層移動が指示されていると判定した場合，ステップＳ１１２で別ページの木構造を表示する。木構造の表示は，表示すべきページ番号を認識後，ステップＳ１０３～ステップＳ１０５の処理と同様の処理を行う。木構造の先頭でないノードに対して階層移動が指示される（階層を降りる旨の指示）と，そのノードを先頭とするノードを表示する。その時のページ番号は，そのノードの自己ページ番号である。木構造の先頭ノードを選択して階層移動を指示すると，そのノードの所属ページ番号が示すページの木構造を表示する。

カ　【００６４】

図８から明らかなように，部分「fore」は，３つの「MW70 巾木(表)」と３つの「MW70 パネル(表)」から構成される。図９に，１つの部品「MW70 巾木(表)」に対応するノード（図８では，便宜的に右肩に「＊」を付してある。）のノードデータの一部を示し，図１０に，そのノードが選択された場合の公開変数表示領域２１の表示例を，図１１に，限定変数表示領域の表示例を示す。

【００６５】

公開変数表示領域に表示される公開変数は，自ノードの公開変数５１と，直系上位ノードの公開変数５２を含み，直系上位のノードの公開変数５２は，自ノードの公開変数５１と異なる色で表示される（図１０では，フォントを変えて示してある。）。また，公開変数には，固定値が入力される公開変数と，代入用スクリプトの実行によって計算される公開変数があり，修飾領域に「なし」あるいは「要計算」を表示することによりに区別され

る。

【００６６】

要計算の公開変数の値は，後述するように代入用スクリプトが実行されるまでは空欄であり，図９及び図１０は，代入用スクリプト実行前の状態を示している。なお，直系上位ノードに要計算の公開変数が含まれ，その公開変数の値が代入スクリプトの実行前で定まっていない場合，その公開変数は，下位ノードの公開変数領域２１に表示されない。すなわち，他のノードからの参照が一時停止される。

【００６７】

代入用スクリプト及び生成用スクリプトは，操作ボタン表示領域２０ａの「評価」ボタンを押すことにより実行される。図１２に，スクリプト実行時の概略動作フローを示す。スクリプトを実行したいターゲットのノード選択し，「評価」ボタンが押されると，評価条件設定用のダイアログボックスを表示する（ステップＳ３０１）。このダイアログボックスでは，評価対象スクリプトの種類と，評価階層が少なくとも可能となっている。すなわち，代入用スクリプトのみの実行，生成用スクリプトの実行，両スクリプトの実行の設定と，自ノードのみを評価するか直系下位ノードの指定階層まで評価するかを設定する。

【００７１】

ステップＳ３０７で，スクリプトを実行すべきノードが残っていると判断された場合は，ステップＳ３０３に戻り，下位のノードについて同様の判断を行い，スクリプトの実行を行う。

キ　【００７２】

次に，代入用スクリプト及び生成用スクリプトの具体例を，図８の「＊」を付したノードをターゲットノードとして説明する。図９に示すように，ターゲットノードは，要計算の公開変数として，「スライス数」と「色」

を有しており，代入用スクリプトとして，「スライス数＝同一面数；」と「色＝同一面数」を有している。評価前は，図１０に示すように，公開変数「スライス数」と「色」の値は空欄となっている。

【００７３】

この状態で，このノードを選択し，「評価ボタン」を押し，評価条件として代入用スクリプトの実行を設定すると，記憶された代入用スクリプトを実行する。したがって，公開変数「スライス数」の値は，上位ノードの公開変数である「同一面数」の値「１」となり，公開変数「色」の値は，同様に上位ノードの公開変数である「巾木色」の値「Ｆ－ ２０５」となる。代入用スクリプト実行後のデザインテーブルの公開変数表示領域２１の表示例を，図１３に示す。

⑵ 前記⑴の記載事項によれば，本件明細書には，本件発明に関し，次のとおりの開示があることが認められる。

ア コンピュータを用いて各種情報の管理を行う場合，それぞれの情報を記憶したファイルを，所定のフォルダに保管することによって行うのが一般的であるが，作成したフォルダの構造及びそれぞれのフォルダに保管するファイルの種類等は，任意であってフォルダの作成者に依存するため，作成者以外の者が必要な情報に適確にアクセスすることは簡単ではなく，多数の者が情報を共有化し，再利用できるように，情報管理を行うことは容易ではない（【０００２】）。

また，従来，案件（プロジェクト）毎にツリーを作成して表示し，作成した文書ファイルを表示されたツリーの任意のノードに付随させてサーバコンピュータに保管することができる文書情報管理システムや，異なる計算機やアプリケーションで共通に取扱うことができるＸＭＬ等の構造化文書を木構造として捉えて処理する構造化文書処理システムが存在したが，木構造のノードに含まれる情報は，相互に関連するものが多いにもかかわ

らず，上記各システムにおいては，それぞれの文書の該当する部分を個別に更新する必要があり，管理される各種情報の更新として，十分効率的とはいえなかった（【０００３】ないし【０００５】）。

イ　「本発明」は，上記事情に鑑み，管理すべき情報の更新を，簡単かつ効率的に行うことができる情報管理方法を提供することを目的とするものであり，その課題を解決する手段として，コンピュータが情報を管理する情報管理方法であって，前記コンピュータに複数のノードそれぞれに対応付けて入力された管理すべき情報を，前記ノードを識別するノード識別情報に対応付けられた複数のノードデータを含む文書ファイルとして前記コンピュータが記憶する情報記憶ステップと，前記情報記憶ステップで記憶された前記文書ファイルの情報を前記コンピュータが表示する情報表示ステップと，前記ノードデータに含まれるスクリプトを前記コンピュータが実行する情報評価ステップとを備え，前記ノードデータは，ルートノードを除いて,当該ノードの親ノードを特定する親ノード識別情報を含んでおり,前記スクリプトは，当該ノードデータに含まれる変数データである自ノード変数データと，当該ノードの直系上位ノードのノードデータに含まれる変数データである上位ノード変数データを利用した演算を行って，前記自ノード変数データの値を求める代入用スクリプトを含んでおり，前記情報表示ステップは，前記親ノード識別情報を利用して，前記ノードの木構造を表示する木構造表示ステップと，前記表示された木構造のノードのうちの選択されたノードの前記自ノード変数データ，前記上位ノード変数データ及び前記スクリプトを表示するノードデータテーブル表示ステップを含み，前記情報評価ステップは，前記代入用スクリプトの実行により，前記自ノード変数データの値を更新するステップを含むという構成を採用した（【０００７】，【０００８】）。

「本発明」によれば，利用者が入力したデータに含まれるスクリプトを

利用してノードデータを更新し，管理すべき情報の更新を簡単かつ効率的に行うことができ，また，複数のノードデータを含む１つの文書データを用いて個々の業務や案件に関する情報を管理しているので，多数の利用者による情報の共有化，再利用を，簡単かつ効率的に行うことができるとともに，文書データに基づいて，簡単にノードの木構造を表示させることができ，業務や案件全体の把握を簡単に行うことができる上，表示された木構造の個々のノードに対応付けられた詳細情報を簡単に表示することができるため，管理すべき情報の更新を，簡単かつ効率的に行うことができるという効果を奏する（【０００９】，【００２４】）。

２　争点１（被告プログラムの本件発明の技術的範囲の属否）について

本件の事案に鑑み，まず，争点１－２について判断し，次いで争点１－４について判断する。

⑴　争点１－２（構成要件Ｅの充足性）について

ア　被告プログラムについて

原判決３３頁１５行目の「１７，」及び同頁１７行目の「(ア)　」を削るほか，同頁１５行目から３４頁２行目までに記載のとおりであるから，これを引用する。

イ　構成要件Ｅの「ルートノード」の意義について

(ア)　本件発明の構成要件Ｅの「前記ノードデータは，ルートノードを除いて，当該ノードの親ノードを特定する親ノード識別情報を含んでおり」との記載によれば，「親ノード識別情報」は「ノードの親ノードを特定する」識別情報であり，「当該ノード」の「ノードデータ」に含まれていることを理解できる。

そして，本件発明の特許請求の範囲には，「ノードデータ」に関し，「前記コンピュータに複数のノードそれぞれに対応付けて入力された管理すべき情報を，前記ノードを識別するノード識別情報に対応付けられ

た複数のノードデータを含む文書ファイルとして前記コンピュータが記憶する情報記憶ステップ」（構成要件Ｂ），「親ノード識別情報」に関し，「前記情報表示ステップは，前記親ノード識別情報を利用して，前記ノードの木構造を表示する木構造表示ステップ」（構成要件Ｇ）との記載がある。これらの構成要件Ｂ及びＧの記載によれば，本件発明においては，「複数のノード」が「木構造」を有し，各「ノード」は当該ノードを識別する「ノード識別情報」を有すること，「親ノード識別情報」は，「木構造」における親子関係にあるノードの親ノードを識別する情報であることを理解できる。

そして，構成要件Ｅの記載から，本件発明には，「ルートノード」と「ルートノード」以外の「ノード」が存在し，「ルートノード」以外の「ノード」の「ノードデータ」は，「親ノード識別情報」を含んでいるが，「ルートノード」の「ノードデータ」は，「親ノード識別情報」を含んでいないことを理解できる。

（イ） 本件明細書には，「ルートノード」を定義した記載はないが，「ルートノード」に関し，「…前記木構造表示ステップが，前記自己ページ番号及び前記所属ページ番号に基づいて，前記ルートノードを先頭とする木構造を表示するとともに，…」（【００１３】），「ノードデータは，例えばタグ付き文書情報として記憶される。図２に，ノードデータの一例を示す。図２のデータは，ルートノードのノードデータの例であり，ノード番号（nodeNo）が「3450」，自己ページ番号（ownPageNo）が「10」，ノードラベル(label)が「パッセル操作マニュアル」である。所属ページ番号を示す（belongPageNo）が「0」，親ノード番号を示す（parentNodeNo）が「0」であることで，ルートノードであることを示している。」（【００３６】），「…図４の文書は，ヘッダ部４０，ノードデータ部４１ａ～４１ｎ，ライン部４２，レポート部４３を備え

る。」（【００３９】），「ノードデータ部４１ａは，ルートノードのノードデータを示し，ノードデータ部４１ｂ～４１ｎは，ルートノード以外のノードのノードデータである。」（【００４１】），「ライン部４２は，ノード間を接続するリードを定義する情報が記憶される領域である。ノード間を接続するリードは，親子のノード間を接続する階層リードと，階層関係とは無関係に一時的に変数を参照する参照元ノードと参照先ノード間を接続する参照リードがある…」（【００４２】），「図６の表示画面は，ツリービューア１０とデザインテーブル２０を有する。ツリービューア１０は，ノードの木構造を表示する領域であり，情報管理時の各種操作を行うためのプルダウンメニュー，及びポップアップメニューを表示する領域も兼ねる。ノードの木構造の表示は，ラベルとリードの表示によって行い，図６の例では，ルートノードのラベル表示１１ａとルートノード以外のノードのラベル表示１１ｂ，１１ｃ，１１ｄと，それらの間を接続する階層リード１２ｂ，１２ｃ，１２ｄが表示されている。」（【００４４】）との記載がある。

そして，図４には，ノードデータ部４１ａの「ｎｏｄｅＮｏ」が「３４５０」の「ノード」は，「ルートノード」であり，「ｐａｒｅｎｔＮｏｄｅＮｏ」が「０」であること，ノードデータ部４１ｂの「ｎｏｄｅＮｏ」が「３４５１」の「ノード」は，「ｐａｒｅｎｔＮｏｄｅＮｏ」が「３４５０」であることが示されている。また，図６には，ルートノードのラベル表示１１ａが，階層関係にあるノードの最初のノードとして表示されており，親ノードに当たるノードが存在しないことが示されている。

これらの記載から，本件明細書には，「ルートノード」は，階層関係にあるノードの「木構造」の先頭のノードであり，親ノードに当たるノードが存在しないこと，このため，「ルートノード」のノードデータの

親ノード番号（「ｐａｒｅｎｔＮｏｄｅＮｏ」）「０」は，親ノードを特定する識別情報には当たらないことが開示されていることが認められる。

（ウ） 以上の本件発明の特許請求の範囲の記載及び本件明細書の記載によれば，構成要件Ｅの「ルートノード」は，階層関係にあるノードの「木構造」の先頭のノードであって，そのノードデータに親ノードを特定する「親ノード識別情報」が含まないものであると解される。

ウ　控訴人の主張について

控訴人は，本件ｂｅｈａｖｉｏｒ．ｘａｒ１（別紙３－２）及び本件ｂｅｈａｖｉｏｒ．ｘａｒ２（別紙７－１）の記載に基づいて，被告プログラムにおけるボックスにおいては，当該ボックスが属する順序系列内における一番最初のボックスであって，当該ボックスの入力コネクタと他のボックスの出力コネクタとの結合情報を持たないボックスが存在し，このボックスは，構成要件Ｅの「ルートノード」に該当する旨主張するので，以下において判断する。

（ア） 本件ｂｅｈａｖｉｏｒ．ｘａｒ１について

ａ　被告プログラムにより，本件ロボットに「こんにちは」を意味する単語をしゃべらせる振る舞いを構築した際に作成されたｂｅｈａｖｉｏｒ．ｘａｒが本件ｂｅｈａｖｉｏｒ．ｘａｒ１（別紙３－２）であり，この内容をフローダイアグラムとして示したものが，別紙６の図１ないし図３である。

本件ｂｅｈａｖｉｏｒ．ｘａｒ１で記述されるＳａｙボックスは，別紙６の図１に示され，ＳａｙボックスのｏｎＳｔａｒｔコネクタとｏｎＳｔｏｐｐｅｄコネクタの間のフローダイアグラムは，別紙６の図２のとおり，①ＳａｙボックスのｏｎＳｔａｒｔコネクタ，②Ｌｏｃａｌｉｚｅｄ　Ｔｅｘｔボックス，③Ｓａｙ　Ｔｅｘｔボックス，

④ＳａｙボックスのｏｎＳｔｏｐｐｅｄコネクタの順にリードで接続されている。

本件ｂｅｈａｖｉｏｒ．ｘａｒ１においては，ｒｏｏｔボックス（３行ないし３０８行)の階層，ｒｏｏｔボックスに包含されるＳｅｔ　Ｌａｎｇｕａｇｅボックス（１８行ないし９１行）及びＳａｙボックス（９２行ないし３００行）の階層並びにＳａｙボックスに包含されるＳａｙ　Ｔｅｘｔボックス（１２５行ないし１７９行）及びＬｏｃａｌｉｚｅｄ　Ｔｅｘｔボックス（１８０行ないし２９１行）の階層が存在し，各ボックスを識別するために割り振られたｉｄは，ｒｏｏｔボックスが「－１」（3行），Ｓｅｔ　Ｌａｎｇｕａｇｅボックスが「2」（１８行），Ｓａｙボックスが「1」（９２行），Ｓａｙ　Ｔｅｘｔボックスが「2」（１２５行），Ｌｏｃａｌｉｚｅｄ　Ｔｅｘｔボックスが「5」（１８０行）である。

そして，上記各ボックス間の結合情報については，２９２行ないし２９４行及び３０１行ないし３０３行の「＜Ｌｉｎｋ　ｉｎｐｕｔｏｗｎｅｒ＝…／＞」という形式で記載されたＬｉｎｋタグ内において，「ｉｎｐｕｔｏｗｎｅｒ」が入力側のボックスのｉｄを，「ｏｕｔｐｕｔｏｗｎｅｒ」が出力側のボックスのｉｄをそれぞれ表している。ただし，２９２行及び２９３行のＬｉｎｋタグ内においては，Ｓａｙボックスのｉｄが「1」から「0」に，３０２行及び３０３行のＬｉｎｋタグ内においては，ｒｏｏｔボックスのｉｄが「－１」から「0」に，それぞれ置き換えられている。

「292: <Link inputowner="0" indexofinput="4" outputowner="2"
indexofoutput="4" />
293: <Link inputowner="5" indexofinput="2" outputowner="0"
indexofoutput="2" />

294: <Link inputowner="2" indexofinput="2" outputowner="5" indexofoutput="3" />」

「301: <Link inputowner="1" indexofinput="2" outputowner="2" indexofoutput="3" />

302: <Link inputowner="0" indexofinput="4" outputowner="1" indexofoutput="4" />

303: <Link inputowner="2" indexofinput="2" outputowner="0" indexofoutput="2" />」

b　前記ａの認定事実によれば，本件ｂｅｈａｖｉｏｒ．ｘａｒ１におけるボックス間の結合関係は，Ｌｉｎｋタグ内の「ｉｎｐｕｔｏｗｎｅｒ」及び「ｏｕｔｐｕｔｏｗｎｅｒ」のｉｄを用いて，本件発明の「親ノード」に相当する結合元のボックスを識別する情報（「ｏｕｔｐｕｔｏｗｎｅｒ」の番号）と本件発明の（子）「ノード」に相当する結合先のボックスを識別する情報（「ｉｎｐｕｔｏｗｎｅｒ」の番号）とにより示されていること，本件ｂｅｈａｖｉｏｒ．ｘａｒ１の３０２行においては，ｉｄが「０」のｒｏｏｔボックスの結合元である親としてｉｄが「１」のＳａｙボックスが記述されていることが認められる。

c　控訴人は,本件ｂｅｈａｖｉｏｒ.ｘａｒ１記載のボックスのうち，当該ボックスが属する順序系列内における一番最初のボックスであって，当該ボックスの入力コネクタと他のボックスの出力コネクタとの結合情報を持たないボックスが，構成要件Ｅの「ルートノード」に該当する旨主張するところ，具体的にどのボックスが「ルートノード」に該当するのか明示していないが，「ｒｏｏｔボックスタグ内の結合情報」を取り上げて主張を構築していることに照らすと，「ｒｏｏｔボックス」が「ルートノード」に該当する旨主張しているものと解さ

れる。

しかるところ，前記ｂ認定のとおり，本件ｂｅｈａｖｉｏｒ．ｘａｒ１には，ｒｏｏｔボックスの結合情報（Ｌｉｎｋタグ内の「ｉｎｐｕｔｏｗｎｅｒ」及び「ｏｕｔｐｕｔｏｗｎｅｒ」のｉｄ）として，結合元である親を示すＳａｙボックスのｉｄである「１」が記述されており，このｉｄ「１」は，ｒｏｏｔボックスの親ノードを特定する「親ノード識別情報」に該当するものと認められる。

そうすると，ｒｏｏｔボックスは，そのノードデータに「親ノード識別情報」が含まないものとはいえないから，構成要件Ｅの「ルートノード」に該当しない。

（イ） 別紙７－１の本件ｂｅｈａｖｉｏｒ．ｘａｒ２について

ａ 本件ｂｅｈａｖｉｏｒ．ｘａｒ２（別紙７－１）も，被告プログラムにより，本 件ロボットに「こんにちは」を意味する単語をしゃべらせる振る舞いを構築した際に作成されたｂｅｈａｖｉｏｒ．ｘａｒであるが，本件ｂｅｈａｖｉｏｒ．ｘａｒ１と異なり，フローダイアグラムボックスが存在しないものである。

本件ｂｅｈａｖｉｏｒ．ｘａｒ２の内容をフローダイアグラムとして示したものが，別紙７－２である。

本件ｂｅｈａｖｉｏｒ．ｘａｒ２においては，ｒｏｏｔボックス（３行ないし２６７行）の階層並びにｒｏｏｔボックスに包含されるＳｅｔ Ｌａｎｇｕａｇｅボックス（１８行ないし９１行），Ｓａｙ Ｔｅｘｔボックス(９２行ないし１４６行)及びＬｏｃａｌｉｚｅｄ Ｔｅｘｔボックス（１４７行ないし２５８行）の階層が存在し，各ボックスを識別するために割り振られたｉｄは，ｒｏｏｔボックスが「－１」（3行)，Ｓｅｔ Ｌａｎｇｕａｇｅボックスが「2」（１８行)，Ｓａｙ Ｔｅｘｔボックスが「3」（９２行），Ｌｏｃａｌｉｚｅｄ

Ｔｅｘｔボックスが「５」（１４７行）である。

そして，上記各ボックス間の結合情報については，２５９行ないし２６２行のＬｉｎｋタグ内において，次のとおり記述されている。前記(ア)ｂと同様に，Ｌｉｎｋタグ内において，「ｉｎｐｕｔｏｗｎｅｒ」が入力側のボックスのｉｄを，「ｏｕｔｐｕｔｏｗｎｅｒ」が出力側のボックスのｉｄをそれぞれ表している。ただし，２６１行及び２６２行のＬｉｎｋタグ内においては，ｒｏｏｔボックスのｉｄが「－１」から「０」に置き換えられている。

「259: <Link inputowner="3" indexofinput="2" outputowner="5"
indexofoutput="3" />
260: <Link inputowner="5" indexofinput="2" outputowner="2"
indexofoutput="3" />
261: <Link inputowner="2" indexofinput="2" outputowner="0"
indexofoutput="2" />
262: <Link inputowner="0" indexofinput="4" outputowner="3"
indexofoutput="4" />」

ｂ　前記ａの認定事実によれば，本件ｂｅｈａｖｉｏｒ．ｘａｒ２におけるボックス間の結合関係も，本件ｂｅｈａｖｉｏｒ．ｘａｒ１と同様に，「ｏｕｔｐｕｔｏｗｎｅｒ」の番号と「ｉｎｐｕｔｏｗｎｅｒ」の番号）により示されていること，本件ｂｅｈａｖｉｏｒ．ｘａｒ２の２６２行においては，ｉｄが「０」のｒｏｏｔボックスの結合元としてｉｄが「３」のＳａｙ　Ｔｅｘｔボックスが記述されていることが認められる。

ｃ　控訴人は，前記(ア)と同様に，本件ｂｅｈａｖｉｏｒ．ｘａｒ２記載のボックスのうち，具体的にどのボックスが「ルートノード」に該当するのか明示していないが，「ｒｏｏｔボックスタグ内の結合情報」

を取り上げて主張を構築していることに照らすと，「ｒｏｏｔボックス」が「ルートノード」に該当する旨主張しているものと解される。

しかるところ，前記ｂ認定のとおり，本件ｂｅｈａｖｉｏｒ．ｘａｒ２には，ｒｏｏｔボックスの結合情報（Ｌｉｎｋタグ内の「ｉｎｐｕｔｏｗｎｅｒ」及び「ｏｕｔｐｕｔｏｗｎｅｒ」のｉｄ）として，結合元である親を示すＳａｙ　Ｔｅｘｔボックスのｉｄである「３」が記述されており，このｉｄ「３」は，ｒｏｏｔボックスの親ノードを特定する「親ノード識別情報」に該当するものと認められる。

そうすると，ｒｏｏｔボックスは，そのノードデータに「親ノード識別情報」が含まないものとはいえないから，構成要件Ｅの「ルートノード」に該当しない。

（ウ）　以上によれば，被告プログラムは構成要件Ｅの「ルートノード」に該当するボックスを備えているとの控訴人の主張は，理由がない。

エ　小括

以上のとおり，被告プログラムは，構成要件Ｅの「ルートノード」に該当する構成を備えているものと認められないから，同構成要件を充足しない。

⑵　争点１－４（構成要件Ｇの充足性）について

ア　構成要件Ｇの「木構造表示ステップ」について

控訴人は，被告プログラムは，Ｌｉｎｋタグ内のｉｎｐｕｔｏｗｎｅｒ（入力側・子）とｏｕｔｐｕｔｏｗｎｅｒ（出力側・親）により結合情報を認識し，これに従ってフローダイアグラム上でボックスとボックス間の結合線で表現することによって「木構造」を表示しており，「親ノード識別情報」を利用して「ノード」の「木構造」を表示しているといえるから，被告プログラムは，構成要件Ｇの「前記親ノード識別情報を利用して，前記ノードの木構造を表示する木構造表示ステップ」を備えている旨主張す

る。

そこで検討するに，本件発明は，「複数のノード」が「木構造」を有していること，このノードの「木構造」の先頭のノードが構成要件Ｅの「ルートノード」であることは，前記(1)イ(ア)及び(ウ)認定のとおりである。

しかるところ，前記(1)ウ認定のとおり，被告プログラムは，構成要件Ｅの｢ルートノード｣に該当する構成を備えているものと認められないから，被告プログラムは，「複数のノード」が「木構造」を有しているものと認めることはできない。

そうすると，控訴人が挙げるフローダイアグラム上のボックスとボックス間の結合線の表現（表示）は，本件発明のノードの「木構造」の表示に当たるものと認めることはできないから，被告プログラムが構成要件Ｇの「前記親ノード識別情報を利用して，前記ノードの木構造を表示する木構造表示ステップ」を備えているとの控訴人の上記主張は理由がない。

イ　構成要件Ｇの「ノードデータテーブル表示ステップ」について

(ア)　構成要件Ｇの「前記上位ノード変数データ」の意義について

a　本件発明の構成要件Ｆの「前記スクリプトは，当該ノードデータに含まれる変数データである自ノード変数データと，当該ノードの直系上位ノードのノードデータに含まれる変数データである上位ノード変数データを利用した演算を行って，前記自ノード変数データの値を求める代入用スクリプトを含んでおり」との記載及び構成要件Ｇの「前記表示された木構造のノードのうちの選択されたノードの前記自ノード変数データ，前記上位ノード変数データ及び前記スクリプトを表示するノードデータテーブル表示ステップ」との記載から，本件発明の「上位ノード変数データ」は，「当該ノードの直系上位ノードのノードデータに含まれる変数データ」であり，構成要件Ｆの「前記自ノード変数データの値」を求める「代入用スクリプト」による演算に利用

される「変数データ」であることを理解できる。

次に，本件明細書には，「上位ノード変数データ」に関し，「変数情報は，各ノードが保持するデータであって，変数名に対応させて記憶される。記憶される変数は，下位ノードから参照される公開変数と，自ノード内でのみ使用する限定変数を含む。また，変数の値（「変数データ」と記述する場合もある。）は，固定値が設定されても，スクリプトの実行によって演算された値が設定されてもよい。また，ＵＲＬが設定されてもよい。どのような値が設定されるかは任意である。」（【００３１】），「代入用スクリプトは，自ノードの変数の値を演算するためのものである。代入用スクリプトは，自ノードの変数の値である自ノード変数データと，そのノードの直系上位ノードの公開変数の値である上位ノード変数データを利用して記述することが可能である。」（【００３２】），「公開変数表示領域に表示される公開変数は，自ノードの公開変数５１と，直系上位ノードの公開変数５２を含み，直系上位のノードの公開変数５２は，自ノードの公開変数５１と異なる色で表示される（図１０では，フォントを変えて示してある。）。また，公開変数には，固定値が入力される公開変数と，代入用スクリプトの実行によって計算される公開変数があり，修飾領域に「なし」あるいは「要計算」を表示することによりに区別される。」（【００６５】）との記載がある。

そして，図１０には，「直系上位ノードの公開変数の値である上位ノード変数データ」として，「５２」に「変数名」及びそれに対応する「値」が示されている（例えば，「変数名」の欄「パネル色」・「値」欄「ＫＷ－４００」）。

これらの記載によれば，本件明細書には，「上位ノード変数データ」にいう「変数データ」は，「変数の値」を含むデータであることの開

示があることが認められる。

以上の本件発明の特許請求の範囲の記載及び本件明細書の記載によれば，構成要件Ｇの「前記表示された木構造のノードのうちの選択されたノードの前記自ノード変数データ，前記上位ノード変数データ及び前記スクリプトを表示するノードデータテーブル表示ステップ」にいう「前記上位ノード変数データ」は，「当該ノードの直系上位ノードのノードデータ」に含まれる「変数の値」を含むデータであると解される。

ｂ　これに対し控訴人は，本件明細書の【００３２】における「変数の値（「変数データ」と記述する場合もある。）」との記載は，「変数データ」という用語を，文脈によって，変数の値を指す意味で用いることもあるという注意書きであると理解できること，「変数データ」は，変数名と変数の型を意味するというのが，プログラミングに関する通常の用語であること（甲２４），実質的にも，本件発明が「ノードデータテーブル表示ステップ」において上位ノード変数データを表示させる目的は，表示された木構造の個々のノードに対応付けられた詳細情報を簡単に表示することができる（【０００９】）ことにより，文書ファイル（プログラム）の編集を容易にする点にあり，変数名が分かれば，その目的を達成することができることからすると，本件発明の「上位ノード変数データ」は，本件明細書において文脈上変数の値を意味すべき場合を除き，変数名を指すと解すべきである旨主張する。

しかしながら，本件明細書には，「上位ノード変数データ」が変数名のみで構成される場合を含むことについての記載や示唆はない。

また，前記ａの本件明細書の記載に照らすと，【００３２】の「変数の値（「変数データ」と記述する場合もある。）」との記載は，「変

数データ」は「変数の値」を意味することを示した記載であると解するのが自然であり，これが変数の値を指す意味で用いることもあるという注意書きであるということはできない。

したがって，控訴人の上記主張は採用することができない。

（イ） 被告プログラムにおける「ノードデータテーブル表示ステップ」の有無について

ａ 控訴人は，入力コネクタは，親ボックスから引き渡される値を記憶する変数が図形化されたものであり，入力コネクタの名称が構成要件Ｇにおける「上位ノード変数データ」に該当すること，インスペクタ及びスクリプトエディタに表示される入力コネクタの名称に関する情報の表示は，上位ノード変数データを表示するものであることからすると，被告プログラムは，「上位ノード変数データ」を表示する「ノードデータテーブル表示ステップ」を備えている旨主張する。

しかしながら，前記(ア)ａ認定のとおり，構成要件Ｇの「前記上位ノード変数データ」は，「当該ノードの直系上位ノードのノードデータ」に含まれる「変数の値」を含むデータであると認められるところ，入力コネクタの名称は，「変数の値」であるとはいえないから，控訴人の上記主張は，その前提を欠くものであり，理由がない。

ｂ 控訴人は，被告プログラムの構成ｇ’に関し，被告プログラムのＳａｙ Ｔｅｘｔボックスの「スクリプトエディタ」において「親からの変数を取得」機能を使う場合，上位ノードであるＳａｙボックスの変数から利用可能なものを一覧表示する機能があるから，被告プログラムは，「上位ノード変数データ」を表示する「ノードデータテーブル表示ステップ」を備えている旨主張する。

しかしながら，控訴人の上記主張は，「スクリプトエディタ」において，どのような「上位ノード変数データ」が表示されるのかについ

て具体的に主張するものではないから，その主張自体理由がない。

c　以上によれば，被告プログラムは，「上位ノード変数データ」を表示する「ノードデータテーブル表示ステップ」を備えているものと認めることはできないから，構成要件Ｇの「前記表示された木構造のノードのうちの選択されたノードの前記自ノード変数データ，前記上位ノード変数データ及び前記スクリプトを表示するノードデータテーブル表示ステップ」を備えているものと認めることはできない。

ウ　まとめ

以上のとおり，被告プログラムは，構成要件Ｇの「木構造を表示する木構造表示ステップ」及び「ノードデータテーブル表示ステップ」を備えているものと認められないから，構成要件Ｇを充足しない。

(3)　小括

以上によれば，被告プログラムは，本件発明の構成要件Ｅ及びＧを充足しないから，その余の点について判断するまでもなく，本件発明の技術的範囲に属するものと認めることはできない。

したがって，その余の点について判断するまでもなく，控訴人の請求は理由がない。

3　結論

以上のとおり，控訴人の請求は理由がないから，控訴人の請求を棄却した原判決は相当である。

したがって，本件控訴は理由がないからこれを棄却することとし，主文のとおり判決する。

知的財産高等裁判所第４部

裁判長裁判官　　大　　鷹　　一　　郎

裁判官　　國　　分　　隆　　文

裁判官　　筈　　井　　卓　　矢

（別紙）

プ ロ グ ラ ム 目 録

名　　称　Ｃｈｏｒｅｇｒａｐｈｅ（コレグラフ）

バージョン　２．３及び２．４

対 象 ＯＳ　Ｍａｃ　ＯＳ，Ｗｉｎｄｏｗｓ及びＬｉｎｕｘ

（別紙７－１）

```
<?xml version="1.0" encoding="UTF-8" ?>
<ChoregrapheProject xmlns="http://www.aldebaran-robotics.com/schema/choregraphe/project.xsd" xar_version="3">
    <Box name="root" id="-1" localization="8" tooltip="Root box of Choregraphe&apos;s behavior. Highest level possible." x="0" y="0">
        <bitmap>media/images/box/root.png</bitmap>
        <script language="4">
            <content>
                <![CDATA[]]>
</content>
        </script>
        <Input name="onLoad" type="1" type_size="1" nature="0" inner="1" tooltip="Signal sent when diagram is loaded." id="1" />
        <Input name="onStart" type="1" type_size="1" nature="2" inner="0" tooltip="Box behavior starts when a signal is received on this input." id="2" />
        <Input name="onStop" type="1" type_size="1" nature="3" inner="0" tooltip="Box behavior stops when a signal is received on this input." id="3" />
        <Output name="onStopped" type="1" type_size="1" nature="1" inner="0" tooltip="ボックスBehaviorの終了時に信号を送る。" id="4" />
        <Timeline enable="0">
            <BehaviorLayer name="behavior_layer1">
                <BehaviorKeyframe name="keyframe1" index="1">
                    <Diagram scale="168.179">
                        <Box name="Set Language" id="2" localization="8" tooltip="Select the language you would like the robot to speak and understand. Any following call to&#x0A;ALSpeechRecognition (Speech Reco. box for instance) or ALTextToSpeech (Say box&#x0A;for instance) will use this language." x="122" y="225">
                            <bitmap>media/images/box/interaction/say.png</bitmap>
                            <script language="4">
                                <content>
                                    <![CDATA[class MyClass(GeneratedClass):

    def __init__(self):
        GeneratedClass.__init__(self, False)

    def onLoad(self):
        try:
            self.tts = ALProxy("ALTextToSpeech")
        except:
            self.logger.warn("ALTextToSpeech is not available, language setting cannot be applied to speech")
            self.tts = None

        try:
            self.asr = ALProxy("ALSpeechRecognition")
        except:
            self.logger.warn("ALSpeechRecognition is not available, language setting cannot be applied to recognition")
            self.asr = None

        try:
            self.dialog = ALProxy("ALDialog")
        except:
            self.logger.warn("ALDialog is not available, language setting cannot be applied to dialog")
            self.dialog = None

    def onInput_onSet(self):
        lang = self.getParameter("Language")
        try:
            if self.asr:
                self.asr.setLanguage( self.getParameter("Language") )
            if self.tts:
                self.tts.setLanguage( self.getParameter("Language") )
            if self.dialog:
                self.dialog.setLanguage( self.getParameter("Language") )
            if self.tts is None and self.asr is None and self.dialog is None:
                raise RuntimeError("Cannot set language: neither ALTextToSpeech nor ALSpeechRecognition nor ALDialog is available.")
            self.onReady()
        except:
            error = "Language " + lang + " cannot be set."
            self.logger.warn(error)
            self.onError(error)]]>
</content>
                                </script>
                                <Input name="onLoad" type="1" type_size="1" nature="0" inner="1" tooltip="Signal sent when diagram is loaded." id="1" />
                                <Input name="onSet" type="1" type_size="1" nature="1" inner="0" tooltip="The data is set when a signal is received on this input." id="2" />
                                <Output name="onReady" type="1" type_size="1" nature="2" inner="0" tooltip="Signal sent when the data has been set." id="3" />
                                <Output name="onError" type="3" type_size="1" nature="2" inner="0" tooltip="Error output:&#x0A;- triggered if the language asked cannot be set" id="4" />
                                <Parameter name="Language" inherits_from_parent="0" content_type="3" value="English" default_value="English" custom_choice="1" tooltip="Set the language the robot speaks and understands." id="5">
                                    <Choice value="Arabic" />
                                    <Choice value="Brazilian" />
                                    <Choice value="Chinese" />
                                    <Choice value="Czech" />
                                    <Choice value="Danish" />
                                    <Choice value="Dutch" />
                                    <Choice value="English" />
                                    <Choice value="Finnish" />
                                    <Choice value="French" />
                                    <Choice value="German" />
                                    <Choice value="Greek" />
                                    <Choice value="Italian" />
                                    <Choice value="Japanese" />
```

```
                                <Choice value="Korean" />
                                <Choice value="Norwegian" />
                                <Choice value="Polish" />
                                <Choice value="Portuguese" />
                                <Choice value="Russian" />
                                <Choice value="Spanish" />
                                <Choice value="Swedish" />
                                <Choice value="Turkish" />
                            </Parameter>
                            <Resource name="Speech" type="Lock" timeout="0" />
                        </Box>
                        <Box name="Say Text" id="3" localization="8" tooltip="Say the text received on its input." x="610" y="179">
                            <bitmap>media/images/box/interaction/say.png</bitmap>
                            <script language="4">
                                <content>
                                    <![CDATA[import time

class MyClass(GeneratedClass):
    def __init__(self):
        GeneratedClass.__init__(self, False)
        self.tts = ALProxy('ALTextToSpeech')
        self.ttsStop = ALProxy('ALTextToSpeech', True) #Create another proxy as wait is blocking if audioout is remote

    def onLoad(self):
        self.bIsRunning = False
        self.ids = []

    def onUnload(self):
        for id in self.ids:
            try:
                self.ttsStop.stop(id)
            except:
                pass
        while( self.bIsRunning ):
            time.sleep( 0.2 )

    def onInput_onStart(self, p):
        self.bIsRunning = True
        try:
            sentence = "\RSPD="+ str( self.getParameter("Speed (%)") ) + "\ "
            sentence += "\VCT="+ str( self.getParameter("Voice shaping (%)") ) + "\ "
            sentence += str(p)
            sentence += " \RST\ "
            id = self.tts.post.say(str(sentence))
            self.ids.append(id)
            self.tts.wait(id, 0)
        finally:
            try:
                self.ids.remove(id)
            except:
                pass
            if( self.ids == [] ):
                self.onStopped() # activate output of the box
                self.bIsRunning = False

    def onInput_onStop(self):
        self.onUnload()]]>
</content>
                            </script>
                            <Input name="onLoad" type="1" type_size="1" nature="0" inner="1" tooltip="Signal sent when Diagram is loaded." id="1" />
                            <Input name="onStart" type="3" type_size="1" nature="2" inner="0" tooltip="Box behavior starts when a signal is received on this Input." id="2" />
                            <Input name="onStop" type="1" type_size="1" nature="3" inner="0" tooltip="Box behavior stops when a signal is received on this Input." id="3" />
                            <Output name="onStopped" type="1" type_size="1" nature="1" inner="0" tooltip="Signal sent when Box behavior is finished." id="4" />
                            <Parameter name="Voice shaping (%)" inherits_from_parent="1" content_type="1" value="100" default_value="100" min="50" max="150" tooltip="Used to modify at runtime the voice feature (tone, speed). In a slighty&#x0A;different way than pitch and speed, it gives a kind of &quot;gender or age&#x0A;modification&quot; effect.&#x0A;&#x0A;For instance, a quite good male derivation of female voice can be&#x0A;obtained setting this parameter to 78%.&#x0A;&#x0A;Note: For a better effect, you can compensate this parameter with the&#x0A;speed parameter. For example, if you want to decrease by 20% the voice&#x0A;shaping, you will have to increase by 20% the speed to keep a constant&#x0A;average speed." id="5" />
                            <Parameter name="Speed (%)" inherits_from_parent="1" content_type="1" value="100" default_value="100" min="50" max="200" tooltip="Changes the speed of the voice.&#x0A;&#x0A;Note: For a better effect, you can compensate this parameter with the voice&#x0A;shaping parameter. For example, if you want to increase by 20% the speed, you&#x0A;will have to decrease by 20% the voice shaping to keep a constant average&#x0A;speed." id="6" />
                        </Box>
                        <Box name="Localized Text" id="5" localization="8" tooltip="Send on the output the text associated with the robot&apos;s current voice language.&#x0A;You can display and edit the text associated with any language by&#x0A;selecting the language in the combobox.&#x0A;&#x0A;!!Warning!! The text sent on the output is NOT the displayed one but the one&#x0A;associated with the robot&apos;s current voice language." plugin="localizationbox_plugin" x="282" y="342">
                            <bitmap>media/images/box/interaction/vocabulary.png</bitmap>
                            <script language="4">
                                <content>
                                    <![CDATA[# /!\ Generated content. Do not edit!
class MyClass(GeneratedClass):
    def __init__(self):
        try: # disable autoBind
            GeneratedClass.__init__(self, False)
```

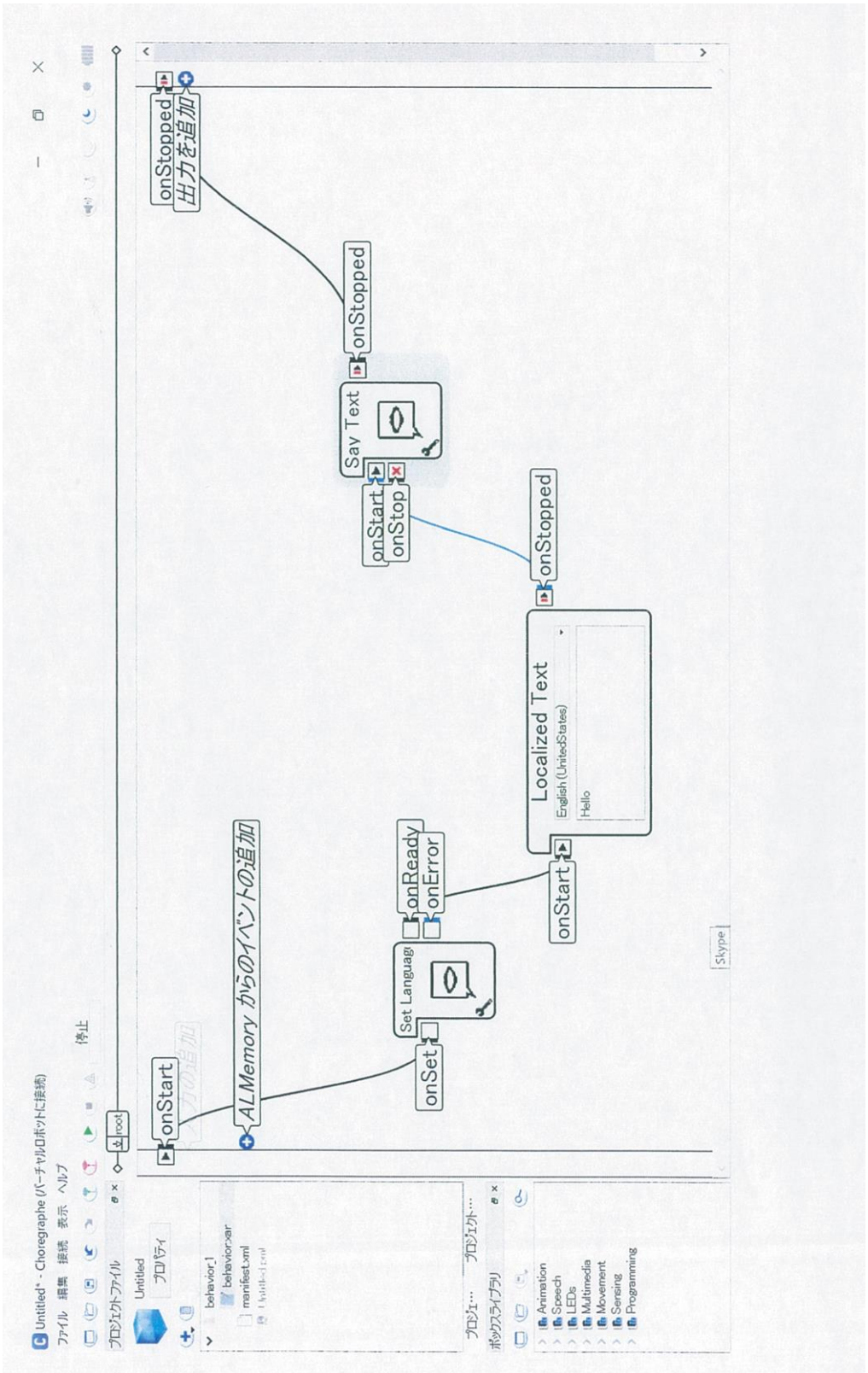
```
        except TypeError: # if NAOqi < 1.14
            GeneratedClass.__init__(self)

        self.tts = ALProxy("ALTextToSpeech")
        self.sentences = {
            "Arabic" : " مرحبا ",
            "Czech" : " Ahoj ",
            "Danish" : " Hej ",
            "German" : " Hallo ",
            "Greek" : " ",
            "English" : " Hello ",
            "Spanish" : " Hola ",
            "Finnish" : " Hei ",
            "French" : " Bonjour ",
            "Italian" : " Ciao ",
            "Japanese" : " こんにちは ",
            "Korean" : " 안녕하세요 ",
            "Dutch" : " Hallo ",
            "Norwegian" : " ",
            "Polish" : " Cześć ",
            "Brazilian" : " Olá ",
            "Portuguese" : " Olá ",
            "Russian" : " П р и в е т ",
            "Swedish" : " Hallå ",
            "Turkish" : " Merhaba ",
            "Chinese" : " 你好 "
        }

    def onInput_onStart(self):
        sDefaultLang = self.tts.getLanguage()
        self.onStopped(self.sentences[sDefaultLang])]]>
</content>
                            </script>
                            <pluginContent>
                                <arabic>
                                    <![CDATA[مرحبا]]>
</arabic>
                                <czech>
                                    <![CDATA[Ahoj]]>
</czech>
                                <danish>
                                    <![CDATA[Hej]]>
</danish>
                                <german>
                                    <![CDATA[Hallo]]>
</german>
                                <greek>
                                    <![CDATA[]]>
</greek>
                                <english>
                                    <![CDATA[Hello]]>
</english>
                                <spanish>
                                    <![CDATA[Hola]]>
</spanish>
                                <finnish>
                                    <![CDATA[Hei]]>
</finnish>
                                <french>
                                    <![CDATA[Bonjour]]>
</french>
                                <italian>
                                    <![CDATA[Ciao]]>
</italian>
                                <japanese>
                                    <![CDATA[こんにちは]]>
</japanese>
                                <korean>
                                    <![CDATA[안녕하세요]]>
</korean>
                                <dutch>
                                    <![CDATA[Hallo]]>
</dutch>
                                <norwegian>
                                    <![CDATA[]]>
</norwegian>
                                <polish>
                                    <![CDATA[Cześć]]>
</polish>
                                <brazilian>
                                    <![CDATA[Olá]]>
</brazilian>
                                <portuguese>
                                    <![CDATA[Olá]]>
</portuguese>
                                <russian>
                                    <![CDATA[П р и в е т]]>
</russian>
                                <swedish>
                                    <![CDATA[Hallå]]>
</swedish>
                                <turkish>
                                    <![CDATA[Merhaba]]>
</turkish>
                                <chinese>
```

```
                                    <![CDATA[你好]]>
</chinese>
                                <language>5</language>
                            </pluginContent>
                            <Input name="onLoad" type="1" type_size="1" nature="0" inner="1" tooltip="Signal sent when diagram is loaded." id="1" />
                            <Input name="onStart" type="1" type_size="1" nature="2" inner="0" tooltip="Data is sent on the output when this input is stimulated." id="2" />
                            <Output name="onStopped" type="3" type_size="1" nature="1" inner="0" tooltip="Data sent when asked." id="3" />
                        </Box>
                        <Link inputowner="3" indexofinput="2" outputowner="5" indexofoutput="3" />
                        <Link inputowner="5" indexofinput="2" outputowner="2" indexofoutput="3" />
                        <Link inputowner="2" indexofinput="2" outputowner="0" indexofoutput="2" />
                        <Link inputowner="0" indexofinput="4" outputowner="3" indexofoutput="4" />
                    </Diagram>
                </BehaviorKeyframe>
            </BehaviorLayer>
        </Timeline>
    </Box>
</ChoregrapheProject>
```

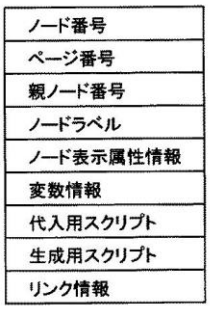
（別紙７－２）


Untitled* - Choregraphe (バーチャルロボットに接続)
ファイル 編集 接続 表示 ヘルプ
停止
プロジェクトファイル
Untitled
プロパティ
behavior_1
behavior.xar
manifest.xml
プロジェ...
プロジェクト...
ボックスライブラリ
Animation
Speech
LEDs
Multimedia
Movement
Sensing
Programming
root
onStart
ALMemory からのイベントの追加
onSet
Set Languag
onReady
onError
onStart
Localized Text
English (UnitedStates)
Hello
onStopped
onStart
onStop
Sav Text
onStopped
onStopped
出力を追加
Skype

（別紙）

明細書図面

【図１】




【図２】

```
<node nodeNo="3450" ownPageNo="10" belongingPageNo="0" parentNodeNo="0"
label="パッセル操作マニュアル" x="100" y="100" width="124" height="18"
ownX="345" ownY="63" ownWidth="130" ownHeight="28" sideNo="0"
decorate="ncolor=-1;fcolor=-16777216;llcolor=-16777216;dir=0;f=10,0;resizable;
frameshape=1;icon=40;" line="message=;fcap=0;tcap=3;f=10,0;lcolor=-16777216;
color=-4144960;lpos=0;" color="0"><![CDATA[
<DataDivision>
</DataDivision>
<ProcedureDivision></ProcedureDivision>
<GenerateDivision></GenerateDivision>
<LinkageDivision>
</LinkageDivision>
</Node>
]]></node>
```

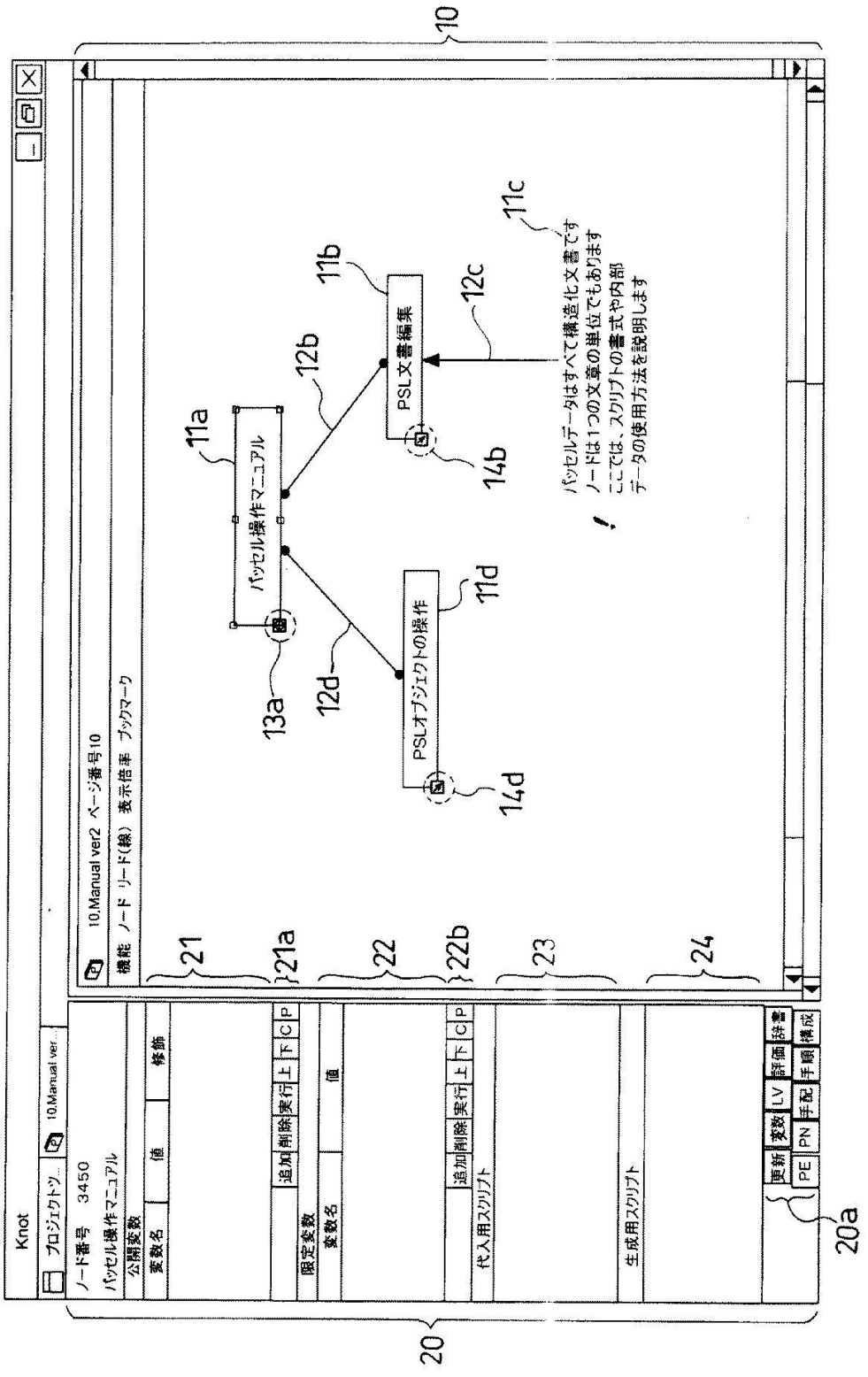
【図３】

```
<node nodeNo="3526" ownPageNo="3526" belongingPageNo="3484" parentNodeNo=
"3488"label="判断文" x="311" y="223" width="44" height="18" ownX="94" ownY="228"
 ownWidth="44" ownHeight="18" sideNo="0" decorate="ncolor=-3355393;fcolor=-16777216;
llcolor=-16777216;dir=0;f=10,0;frameshape=0;" line="message=;fcap=0;tcap=3;f=10,0;
lcolor=-16777216;color=-4144960;lpos=0;" color="0"><![CDATA[
<DataDivision>
</DataDivision>
<ProcedureDivision></ProcedureDivision>
<GenerateDivision></GenerateDivision>
<LinkageDivision>
</LinkageDivision>
</Node>
]]></node>
```

【図４】


<project projectNo="10" name="Manual ver2" createdate="2004/05/01" note="">
40
<nodearea>
<node nodeNo="3450" ownPageNo="10" belongingPageNo="0" parentNodeNo="0" label="パッセル操作マニュアル" ・・・・
・・・・</node>
41a
<node nodeNo="3451" ownPageNo="3451" belongingPageNo="10" parentNodeNo="3450" label="PSL文書編集" ・・・・
・・・・</node>
41b
<node nodeNo="3479" ownPageNo="0" belongingPageNo="10" parentNodeNo="3451" label="パッセルデータはすべて構造化文書です
ノードは1つの文章の単位でもあります
ここでは、スクリプトの書式や内部
データの使用方法を説明します" ・・・・
・・・・</node>
41c
<node nodeNo="3623" ownPageNo="3623" belongingPageNo="10" parentNodeNo="3450" label="PSLオブジェクトの操作"・・・・
・・・・</node>
41d
41e
41n
</nodearea>
<linearea>
<line lineNo="1" fromNodeNo="3502" toNodeNo="3503" PageNo="3489" decorate="fcap=0;tcap=3;f=10,0; ・・・・ Target="" Parameter="" Message="" Options="" />
・・・・
・・・・
・・・・
42
</linearea>
<reportarea>
<report reportNo="2" nodeNo="3833" title="hello" contents="aaa" createDate="2004/06/23" />
・・・・
43
</project>

【図６】


Knot
プロジェクトツ...
10.Manual ver...
ノード番号 3450
パッセル操作マニュアル
公開変数
変数名
値
修飾
追加
削除
実行
上
下
C
P
限定変数
変数名
値
追加
削除
実行
上
下
C
P
代入用スクリプト
生成用スクリプト
更新
変数
LV
評価
辞書
PE
PN
手配
手順
構成
20a
20
10
10.Manual ver2 ページ番号10
機能 ノード リード(線) 表示倍率 ブックマーク
21
21a
22
22b
23
24
パッセル操作マニュアル
PSL文書編集
PSLオブジェクトの操作
パッセルデータはすべて構造化文書です
ノードは１つの文章の単位でもあります
ここでは、スクリプトの書式や内部
データの使用方法を説明します
11a
11b
11c
11d
12b
12c
12d
13a
14b
14d

【図１０】

| | 変数名 | 値 | 修飾 |
|---|---|---|---|
| 51 | _x | 930.0 | なし |
| | _y | 0.0 | なし |
| | _w | 800.0 | なし |
| | _h | 75.0 | なし |
| | ・・・ | ・・・ | ・・・ |
| | スライス数 | | 要計算 |
| | 色 | | 要計算 |
| | 巾木材質 | スチール | なし |
| | ・・・ | | |
| 52 | _layername | fore | なし |
| | 面NO | 1 | なし |
| | 面ID | MW | なし |
| | 同一面数 | 1 | なし |
| | ・・・ | | |
| | パネル色 | KW-400 | なし |
| | 巾木色 | F-205 | なし |
| | ・・・ | | |
| | 登録年月 | 200310 | なし |
| | 製造番号 | 000000990070 | なし |
| | ・・・ | | |