(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3630457号
(P3630457)
(45)発行日　平成17年3月16日(2005.3.16)　(24)登録日　平成16年12月24日(2004.12.24)

(51)Int.Cl.[7]	F I		
G06F 9/44	G06F	9/06	620B
G06F 1/00	G06F	1/00	370A
G06F 9/00	G06F	9/00	320B
G06F 9/48	G06F	15/00	310Q
G06F 15/00	G06F	9/46	457

請求項の数 5　(全 30 頁)

(21)出願番号　特願平6-312426
(22)出願日　平成6年11月22日(1994.11.22)
(65)公開番号　特開平7-287651
(43)公開日　平成7年10月31日(1995.10.31)
審査請求日　平成13年10月17日(2001.10.17)
(31)優先権主張番号　156,592
(32)優先日　平成5年11月22日(1993.11.22)
(33)優先権主張国　米国(US)

(73)特許権者　398038580
ヒューレット・パッカード・カンパニー
HEWLETT-PACKARD COMPANY
アメリカ合衆国カリフォルニア州パロアルト　ハノーバー・ストリート　3000
(74)代理人　100081721
弁理士　岡田　次生
(72)発明者　ジョセフ・ジェイ・カーラント
アメリカ合衆国ミシガン州フォウラーヴィル、グレゴリー・ロード　7805

最終頁に続く

(54)【発明の名称】コンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法

(57)【特許請求の範囲】

【請求項１】

電子デジタル・コンピュータと、前記のコンピュータを動作させるための、事象デイスパッチング機構を介して互いに通信するための手段を有する複数のソフトウエア・ツールからなる、コンピュータ・ソフトウエアを含むメモリからなるコンピュータ・システム内においてソフトウエア・ツールの実行を自動化および制御する方法であって、

ルーチン編集プログラムが、人間の言語による抽象表現を用いて少なくとも１つのツール対話ルーチンを作成するステップであって、前記のツール対話ルーチンはユーザの定義する規則の集合であり、その規則の集合により、ユーザの指定する事象が発生すると、前記複数のソフトウエア・ツールの少なくとも一つの、少なくとも１つのユーザによって指定される動作が開始されるものであり、ツール機能のリストを表示し、当該リストの中から少なくとも1つのユーザの指定したツール機能を、「when」事象として選択するステップと、当該リストの中から少なくとも1つのユーザの指定したツール機能を、「then」事象として選択するステップとを含むステップ、

エンジンが、前記の少なくとも１つのツール対話ルーチンの前記の人間の言語による抽象表現に関連する生のメッセージを、一連の機械の実行可能な動作からなるルーチン・ファイルに変換するステップ、

ルーチン管理プログラムが、少なくとも１つのツール対話ルーチンをイネーブルするステップ、

メッセージ結合子が、前記のユーザにより指定される事象の発生をモニタするステップ、

前記のユーザにより指定される事象が発生すると、前記メッセージ結合子が、前記複数のソフトウエア・ツールの前記少なくとも1つに、前記の少なくとも１つのユーザの指定する動作を開始するように指示するステップ、
からなるコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法。

【請求項２】

コンピュータ・システム内において複数のソフトウエア・ツールの実行を自動化および制御する装置であって、
中央処理装置と前記のコンピュータを動作させるための、複数のソフトウエア・ツールからなるコンピュータ・ソフトウエアを格納した記憶装置を含む電子デジタル・コンピュータ、
前記の複数のソフトウエア・ツールの間で事象デイスパッチングするための事象サーバ、
ツール機能のリストを表示し、当該リストの中から少なくとも1つのユーザの指定したツール機能を、「when」事象として選択し、当該リストの中から少なくとも1つのユーザの指定したツール機能を、「then」事象として選択することによって、人間の言語による抽象表現を用いて、ユーザの定義した規則の集合からなるユーザの指定するルーチンを作成および編集し、これによって少なくとも１つのユーザの指定する事象が発生すると、少なくとも１つの対応するユーザの指定する動作が開始されるルーチン編集プログラム、
前記のルーチン編集プログラムによって作成される任意のルーチンを管理するためのルーチン管理プログラム、
前記の人間の言語による抽象表現に関連する生メッセージを、一連の機械の実行可能な動作からなるルーチン・ファイルに変換するエンジン、
ユーザの指定する事象の発生をモニタし、前記のユーザの指定する事象に対応する任意のユーザの指定する動作に対する要求メッセージを前記の事象サーバに通信するメッセージ結合子からなる、ソフトウエア・ツールの実行自動化および制御装置。

【請求項３】

電子デジタル・コンピュータと、前記のコンピュータを動作させるための、事象デイスパッチング機構を介して互いに通信するための手段を有する複数のソフトウエア・ツールからなる、コンピュータ・ソフトウエアを含むメモリからなるコンピュータ・システム内においてソフトウエア・ツールの実行を自動化および制御する方法であって、
ルーチン編集プログラムが人間の言語による抽象表現を用いて、ルーチン編集プログラムによって提供される少なくとも１つのツール対話ルーチンを作成するステップであって、
前記のツール対話ルーチンはユーザの定義する規則の集合であり、その規則の集合により、ユーザの指定する事象が発生すると、少なくとも１つのユーザによって指定される動作が開始されるものであり、ツール機能のリストを表示し、当該リストの中から少なくとも1つのユーザの指定したツール機能を、「when」事象として選択するステップと、当該リストの中から少なくとも1つのユーザの指定したツール機能を、「then」事象として選択するステップとを含むステップ、
エンジンが前記の少なくとも１つのツール対話ルーチンの前記の人間の言語による抽象表現に関連する生のメッセージを、一連の機械の実行可能な動作からなるルーチン・ファイルに変換するステップ、
ユーザがルーチン管理プログラムを用いて少なくとも１つのツール対話ルーチンをイネーブルした後に、メッセージ結合子が前記の少なくとも１つのツール対話ルーチンのユーザによって指定される事象を検出するステップ、
前記のユーザにより指定される事象が発生すると、前記のエンジンが前記のユーザにより指定される事象に関連する前記の少なくとも１つのユーザの指定する動作を開始するステップからなるコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法。

【請求項４】

少なくとも１つのツール対話ルーチンを作成するステップが、ツール対話ルーチンにユー

ザの定義するパターン照合を適用するステップをさらに含む請求項１または３に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法。
【請求項５】
ルーチン編集プログラムが、ツール対話ルーチンにユーザの定義するパターン照合を適用する請求項２に記載のソフトウエア・ツールの実行自動化および制御装置。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】
本発明は、広義にはコンピュータシステムの分野に関し、詳細にはソフトウエアツール間の通信と連動を向上させるプログラム通信装置によりソフトウエア・ツールの実行自動化および制御する方法関する。より詳細には、本発明はソフトウエアツールの実行を自動化および制御し、その動作を変更するためのコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法に関する。
【０００２】
【従来の技術】
コンピュータ・システムは通常ユーザによる特定のタスクの実行を助けるように設計された多数の異なるツールあるいはプログラムを含む。コンピュータ・システムに用いられるより一般的なツールにはたとえばプログラム編集ツール、構築ツール、スペル・チェック・ツール、デバッグ・ツール、バージョン管理ツール等がある。従来のほとんどのコンピュータ・システム環境においては、ツールの連動は最も簡単なツールの相互関係であっても、ユーザがそれらを組み合わせることによって達成される。かかるコンピュータ・システムはその基本的形態においてユーザがウィンド型のインターフェースを介して特定のツールの機能を利用できるものである。かかるコンピュータ・システムのツール・セット１００のブロック図を図１５に示す。図１５のシステムからツールＡ　１１０、ツールＢ　１２０、ツールＣ　１３０はコンピュータ・システム１００上で他のツールと通信することができないことがわかる。
【０００３】
【発明が解決しようとする課題】
図１６は、それぞれのツールの機能へのプログラムにしたがったアクセスに関するツール・セットのブロック図を示す。基本的には、図１６には各ツール１１０、１２０、１３０がメッセージ（１１２、１２２、１３２）を送出し、またメッセージ（１１４、１２４、１３４）を受け取ることのできるシステムを示す。このシステムの問題点は、それぞれのツールはメッセージを送受する能力を有するが、個々のツールの間に通信が行なわれないことである。
【０００４】
図１７は異なるツールの機能の間の関係が固定されたツール・セットのブロック図である。図１７に示すもののようなシステムの実施に関するより詳細な情報が「Ｍｅｔｈｏｄ　ａｎｄ　Ａｐｐａｒａｔｕｓ　ｆｏｒ　Ｃｏｍｐｕｔｅｒ　Ｐｒｏｇｒａｍ　Ｅｎｃａｐｓｕｌａｔｉｏｎ　（コンピュータ・プログラム・カプセル封止用方法および装置）」と題する米国特許第５，０７５，８４７号に開示されている。図１７のシステムは個々のツールがユーザの介入を必要とせずに他のツールに動作を要求する手段を設けることによってユーザの生産性を向上させるという点で図１６のシステムに対して改良されたものである。図１７のシステムはあらかじめプログラムされた事象駆動されるシステムである。たとえば、ツールＡ　１１０が編集ツールであり、ツールＢ　１２０がスペル・チェックツールであると仮定すると、このシステムは編集ツールがファイルを保管するときスペル・チェック・ツールが必ずそのファイルをチェックするようにあらかじめプログラムすることができる。かかるシステムは一般的には満足すべきものであるが、ソフトウエア・プログラマにしか変更できない非常に特殊なあらかじめプログラムされた対話を有する。
【０００５】
したがって、一般のユーザはシステムを所望のツール対話にカスタマイズすることができ

ない。一般のユーザはあらかじめプログラムされたツールおよび事象の相互関係になれる、あるいはツール使用をマニュアルで呼び出す必要がある。ユーザは必要に応じてツールをマニュアルで呼び出すことができるが、ツール対話システムを独自のニーズに合うようにカスタマイズすることができることが望ましく、これによってユーザはソフトウエア・ツールの調整や管理よりもアプリケーション・タスクに集中することができる。しかし、一般のユーザのほとんどはソフトウエアをプログラムすることができないため、ツールや事象の相互関係を自ら作成することはできない。したがって、一般のユーザが自己のソフトウエア・ツールの動きやソフトウエア・ツール間の相互関係を、そのソフトウエア・ツールのコード変更、または１行のコード書き込みを行うことなく簡単に変更できるシステムが必要とされている。

【０００６】

【課題を解決するための手段】

本発明によれば、上記の目的および利点またその他の目的および利点は、ユーザ・インターフェース、オペレーティング・システム、および所定のタスクを実行し、事象サーバを介してやり取りされる事象によって互いに通信する１つあるいはそれ以上の統合ソフトウエアを含むコンピュータ・ソフトウエア・システム内のツールあるいはツール・セットの実行を自動化および制御し、またその動作を変更する方法において達成される。このシステムはｗｈｅｎ（するときに）／ｔｈｅｎ （次に）システムであり、指定された事象が検出されるとき（「ｗｈｅｎ」）、（「ｔｈｅｎ」）対応する動作が開始される。

【０００７】

この方法は機能の接続にどのツールが使用可能であるかを判定するステップとかかるツールをユーザが選択できるように提示するステップからなる。ユーザがあるツールを選択すると、選択されたツールに固有の機能がユーザに提示され、これによってユーザはさらに選択されたツールの特定の機能あるいは事象を「ｗｈｅｎ」事象として選択することができる。ユーザが「ｗｈｅｎ」事象となる特定の事象を選択した後、ユーザに対してツールのリストが提示され、このリストから「ｔｈｅｎ」動作のための動作ツールが選択される。ユーザが「ｔｈｅｎ」動作のためのツールを選択すると、選択されたツールに固有の機能が提示され、ユーザはさらに選択された「ｔｈｅｎ」動作の特定の機能を選択することができる。また、ユーザは１つの「ｗｈｅｎ」事象に続く２つ以上の「ｔｈｅｎ」動作を選択することができる。

【０００８】

ユーザがルーチンを作成した後、ユーザはこれに名前を付け、保管し、それをイネーブルするあるいは使用不能にすることができる。ｗｈｅｎ／ｔｈｅｎ ツール・セットの多数のルーチンを作成してエンド・ユーザの個々のニーズを満たすことができる。それぞれのルーチンは動作のファイルである。ルーチンが作成された後、ユーザのニーズに応じてこれらのルーチンを表示、変更、複製、削除、名称変更、イネーブルあるいは使用不能化することができる。複数動作ルーチンは複雑なタスクを自動化して、オペレーティング・システムあるいは個々のツールの１つの事象が長い自動ツール対話処理を開始できるようにする。異なるルーチンがイネーブルされると、メッセージ結合子ツールがそれぞれのルーチンの「ｗｈｅｎ」事象を登録し、それらのメッセージのいずれかが検出されるまで待つ。メッセージ結合子ツールはある「ｗｈｅｎ」事象のメッセージを検出すると、検出された「ｗｈｅｎ」事象メッセージに関連するすべての「ｔｈｅｎ」事象動作メッセージを送出する。

【０００９】

本発明によれば、ユーザは任意のツールを図形を用いて選択し、そのツールから検出したい事象を選択し、その事象を用いて他の任意のソフトウエア・ツールの機能を呼び出すことができる。この機能の結合はソフトウエア・プログラミングの知識を必要とせずユーザが行なうことができる。さらに、ユーザは経過時間等の非ツール「ｗｈｅｎ」事象を選択して「ｔｈｅｎ」動作を起動することができる。このツール・ルーチンのタスクのカスタム化／自動化は、従来プログラムの書き込み、試験および文書化に必要としたような人数

や数か月単位の期間を必要とせずわずか数分で行なうことができる、また、本発明によって１つのタスクを実行する際に要した何時間にもおよぶ複数のユーザ／ツール対話を省くことができる。

【００１０】

【実施例】

本発明のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法は、メッセージ・サーバおよびメッセージ結合子ツールを含むすべてのソフトウエア・アプリケーション・ツールを実行する単一のコンピュータ、あるいは１つあるいはそれ以上のソフトウエア・ツールがネットワーク化された異なるローカル・コンピュータあるいは遠隔のコンピュータ上で動作する分散型計算環境において実施することができる。

【００１１】

以下、本明細書で使用される用語を次のように定義する。

抽象：分かりやすいように、コンピュータ文に代わって用いられる人間の言語（英語、日本語その他）文。

並行：並行して（同時に）実行すべき他の動作のリストを含む動作。並行動作はそれに含まれるすべての動作が完了したとき完了したものとみなされる。

不能化：特定のルーチンをただちに不能とする動作であり、その特定のルーチンのすべてのスレッドがただちに破壊されることを意味する。

エンジン：メッセージ・サーバと実際のｗｈｅｎ／ｔｈｅｎ　メッセージ対話を実施するメッセージ結合子の部分。

イネーブル：特定のルーチンをただちにイネーブルする動作であり、その特定のルーチンのあるスレッドが開始されることを意味する。

事象：ツールによって送られ、集中自称サーバによって他のツールに再分散される情報のパケット。

事象サーバ：事象を再分散するプログラムあるいは処理。ソフトウエア・ツール間の通信を可能にし、事象の授受を介して支援されるプログラムあるいは処理。たとえば、「Ｈｅｗｌｅｔｔ－Ｐａｃｋａｒｄ　Ｂｒｏａｄｃａｓｔ　Ｍｅｓｓａｇｅ　Ｓｅｒｖｅｒ（ヒューレット・パッカード放送メッセージ・サーバ）」{Ｈｗｅｌｅｔｔ－Ｐａｃｋａｒｄ　Ｃｏｍｐａｎｙ（ヒューレット・パッカード・カンパニ）の登録商標）、「Ｓｕｎ　ＴｏｏｌＴａｌｋ　Ｍｅｓｓａｇｅ　Ｓｅｒｖｅｒ」{Ｓｕｎ　Ｍｉｃｒｏｓｙｓｔｅｍｓ　Ｃｏｒｐｏｒａｔｉｏｎ（サン・マイクロシステム・コーポーレーション）の登録商標}、「Ｘ　Ｗｉｎｄｏｗｓ　Ｓｙｓｔｅｍ（Ｘ　ウインドウズ・システム）」{ＭＩＴ　（エム・アイ・テイ）の登録商標}、「Ｍｉｃｒｏｓｏｆｔ　Ｗｉｎｄｏｗｓ　（マイクロソフト・ウインドズ）」{Ｍｉｃｒｏｓｏｆｔ　Ｃｏｒｐｏｒａｔｉｏｎ　（マイクロソフト・コーポーレーション）の登録商標} 等がある。

実行管理プログラム：適当なツールがまだ始動されていない場合、要求をサービスするためのツールを始動させるプログラムあるいは処理。

ログファイル：対象となるすべての「ｗｈｅｎ」事象と「ｔｈｅｎ」事象の履歴を含むファイル。

メッセージ：事象の一種

メッセージ結合子：ユーザが他のツールからの事象を自動的に検出し、他のツールからのユーザによって指定された機能を要求することによってかかる事象に応答することを可能にするツール。

動作：あるルーチンにおける１つのステップ。動作としては、「ＷａｉｔＦｏｒＲｅｑｕｅｓｔ」（要求待ち）、「ＷａｉｔＦｏｒＮｏｔｉｆｙ　（通知待ち）」、「ＳｅｎｄＲｅｑｕｅｓｔ　」（要求送信）、「ＳｅｎｄＮｏｔｉｆｙ（通知送信）」、「Ｅｎａｂｌｅ（イネーブル）」、「Ｄｉｓａｂｌｅ　（不能化）」および「Ｃｏｎｃｕｒｒｅｎｔ（並行）」等がある。

パターン：サーチ・パターンを定義する１つの正則表現あるいは一連の正則表現。たとえば、＊．ｔｘｔ　はサブストリング．ｔｘｔで終わるすべてのストリングを突き合わせる

パターンである。
パターン・ケース：「ＷａｉｔＦｏｒＮｏｔｉｆｙ　」、「ＷａｉｔＦｏｒＲｅｑｕｅｓｔおよびＳｅｎｄＲｅｑｕｅｓｔ　」の各動作によって受け取られ、受け付けられる事象に突き合わせられるパターンのリスト。それぞれのパターンはその特定のパターンが事象と一致するとき実行される動作のリストを有する。
パターン突き合わせ：コード化表現をすべての項目の表現と比較することによって、あるオブジェクトに最も近い所定の項目の集合のうちの１つを同定すること。
生メッセージ：ソフトウエア・ツールとの間でやり取りされる特定のコンピュータの意味文。
ルーチン：ユーザが順に実行されることを希望する動作のリスト。任意の時点において、ルーチンはイネーブルされる（実行される）か不能とされる（実行されない）。あるルーチンがイネーブルされると、そのルーチンの１つあるいはそれ以上のスレッドが実行される。ルーチンが不能とされると、そのルーチンのいかなるスレッドも実行されない。
「　ＲｏｕｔｉｎｅＬｉｓｔ」：動作リストの変わりに他のルーチンの名前のリストを含む特殊なルーチン。ルーチンリストをイネーブルすると、リストされたルーチンがイネーブルされる。ルーチンリストが不能とされると、リストされたルーチンが不能になる。ルーチンリスト自体は実行すべき動作を有しないため、イネーブル状態にも不能状態にもならない。
「　ＳｅｎｄＮｏｔｉｆｙ　」：指定された通知事象を事象サーバに送る動作。
「　ＳｅｎｄＲｅｑｕｅｓｔ」：指定された要求事象を事象サーバに送り、任意にこの要求事象がその環境内の他のツールによってサービスされるとき生成される関連する通知事象を待つ動作。次に、この通知動作は「ＷａｉｔＦｏｒＮｏｔｉｆｙ　」と同様にパターンケースに対して突き合わされる。
ストリング：一連の文字｛たとえば、英語の文、「ｍｙｄａｔａ．ｔｘｔ（マイ・データ・テキスト）」、「ｔｈｉｓ　ｉｓ　ａ　ｓｔｒｉｎｇ（これはストリングである）」その他｝。
「ｔｈｅｎ」動作：「ｗｈｅｎ」事象に対するユーザの指定する応答であり、他のツールの機能の実行の要求であることが多い。
スレッド：あるルーチンの独立的に実行されるコピー。
スレッド・ツリー解釈プログラム：あるスレッドのツリー構造によって記述される動作を実行するコード。
スレッド・ツリー構造：そのスレッドによって実行されているルーチンを表わすデータ構造。
スレッド変数構造：あるスレッドに用いられる変数を含むデータ構造。
ツール：所定の機能あるいは機能群を実行するためのソフトウエア・プログラム。
「　ＷａｉｔＦｏｒＮｏｔｉｆｙ」：ルーチン・スレッドの実行を「ＷａｉｔＦｏｒＮｏｔｉｆｙ　」動作において指定されたものに一致する通知事象が事象サーバから受け取られるまで停止するルーチン中の１つの動作。それぞれの「ＷａｉｆＦｏｒＮｏｔｉｆｙ　」動作はある事象に対する２つ以上の一致を指定するためのパターン・ケースのリストを提供することができる。次に、それぞれのパターンはそのパターン・ケースが発生するとき常に実行される動作のそれ自身のリストと関係付けることができる。
「　ＷａｉｔＦｏｒＲｅｑｕｅｓｔ　」：通知事象のかわりに要求事象をトリガすることができることを除いて「ＷａｉｔＦｏｒＮｏｔｉｆｙ　」と同じ動作。この動作は使用中の特定の事象サーバが要求事象の概念を持ち、「ｅａｖｅｓｄｒｏｐｐｉｎｇ　（盗聴）」を支援する場合にのみ有効である。
「ｗｈｅｎ」事象：メッセージ結合子が検出される、あるいは待機している事象の記述。
【００１２】
図１は、本発明によるツールの機能間のフレキシブルなユーザの定義する相互関係を有するツールＡ　、Ｂ　、Ｃ　、…Ｘ　からなるツール・セットのブロック図である。要素２５０はメッセージ結合子ツールからなり、事象サーバは好適には、通常他のツールで生じ

るなんらかの事象によって事象がシステム内で発生するトリガ機構を含むたとえば、ツールＡ　２１０が他のツール、たとえば、ツールＣ　２３０内の特定の機能を呼び出す。このトリガ機構はユーザがプログラムすることができ、ここでｗｈｅｎ／ｔｈｅｎ　ルーチンと呼ぶ因果関係を作成する。このトリガ機構を用いてユーザの介入を必要とすることなく指定された事象が発生したとき自動的に動作をトリガすることができる。因果関係はユーザがプログラムすることができるが、ユーザは１行のコードも書かないということに注意しなければならない。さらに、個々のツールのコードは変更されない。

【００１３】

異なる事象が対話を行なって事象駆動通信とソフトウエアの実行が行なわれる。第１のツールによって生成された通知メッセージがメッセージ結合子ツール２５０と事象サーバを介して第２のツールに送られると、第２のツールの事象駆動トリガリングが達成される。たとえば、テキスト編集ツールからの首尾良い完了通知が事象サーバに送られると、これは事象サーバによってメッセージ結合子に送られる。メッセージ結合子はこれによって構築ツールの呼び出しを要求することができる。事象は次に構築ツールが開始した事象サーバに送ることができ、その結果メッセージ結合子はユーザの電子メールの表示を要求する。このようにして、適当な通知、要求、事象および／またはメッセージによって通信される事象の発生に基づいてソフトウエア・ツールが呼び出され、それによってユーザがリアルタイムで介入することなくソフトウエア・ツール間の閉じた対話が可能になる。

【００１４】

図２は事象サーバ５０が複数のソフトウエア・ツール５２～５７の間の中央事象分配機構として機能する計算環境７０の概念ブロック図である。この環境のツールはたとえば構築ツール５２、編集ツール５３、デバッグツール５４、スタティック・アナライザ・ツール５５、メッセージ結合子ツール５６およびスペル・チェック・ツール５７（以下、これらのツールを総称する場合には、ソフトウエア・ツールという）を含む。計算環境７０は事象サーバ５０とソフトウエア・ツール５２～５７を実行可能な単一のコンピュータを用いて実施することができる。あるいは、計算環境７０は事象サーバ５０とソフトウエア・ツール５２～５７が異なるコンピュータ上で実行される分散型計算環境において実施することができる。先行技術の最良の実施態様において、複数のソフトウエアツールがツールをフレーム・ワークに入れるのに用いられる「Ｈｅｗｌｅｔｔ－Ｐａｃｋａｒｄ　」の「Ｅｎｃａｐｓｕｌａｔｏｒ（カプセル封止器）」を有する「Ｈｅｗｌｅｔｔ－Ｐａｃｋａｒｄ　ＳｏｆｔＢｅｎｃｈ　（ヒューレット・パッカード・ソフトウエアベンチ）」フレーム・ワークにおいて動作し、また事象サーバとして用いられる「Ｈｅｗｌｅｔｔ－Ｐａｃｋａｒｄ　」の「ＳｏｆｔＢｅｎｃｈ　Ｍｅｓｓａｇｅ　Ｓｅｒｖｅｒ（ソフトベンチ・メッセージ・サービス）」において動作し、この場合用いられる事象はメッセージと呼ばれる。「Ｈｅｗｌｅｔｔ－Ｐａｃｋａｒｄ　」の「Ｓｏｆｔｂｅｎｃｈ　」製品を動作させるコンピュータであればどれでも本発明に用いることができる。このメッセージ・サーバに関するより詳細な情報については、「Ｍｅｔｈｏｄ　ａｎｄ　Ａｐｐａｒａｔｕｓ　ｆｏｒ　Ｃｏｎｔｒｏｌｌｉｎｇ　Ｅｘｅｃｕｔｉｏｎ　ｏｆ　Ｔｏｏｌｓ　ｉｎ　ａ　Ｃｏｍｐｕｔｅ－Ａｉｄｅｄ　Ｓｏｆｔｗａｒｅ　Ｅｎｇｉｎｅｅｒｉｎｇ　Ｓｙｓｔｅｍ（コンピュータ支援ソフトウエア工学システムのツールの実行を制御するための方法と装置）」と題する上述の米国特許５，２１２，７９２号を参照されたい。

【００１５】

１つのソフトウエア・ツールが他のソフトウエア・ツールと通信するためには、それぞれのソフトウエア・ツールと事象サーバ５０の間にまず双方向の通信路すなわちソケットを確立しなければならない。ソケットは双方向のポイント・ツー・ポイント通信路である。図２に示すように、構築ツール５２と事象サーバ５０の間にソケット７２が確立される。同様にして、ソケット・ツール、すなわち、ソフトウエア・ツール５３、５４、５５、５６、５７と事象サーバ５０の間にソケット７３、７４、７５、７６、７７が確立される。

【００１６】

他のツールとの通信はすべてメッセージ結合子ツール５６と事象サーバ５０を介して発生

する。メッセージ結合子ツール５６は後で図３を参照して説明するルーチン管理プログラム・ウインド３１０、ルーチン編集プログラム・ウインド３２０およびエンジン３３０からなる。メッセージ結合子ツール５６はユーザによって定義される「ｗｈｅｎ－ｔｈｅｎ」ルーチンを含む規則のリストを有する。メッセージ結合子ツール５６は所定の「ｗｈｅｎ」事象のうちの任意のものを検出し、ある「ｗｈｅｎ」事象を検出すると、メッセージ結合子ツール５６は検出された「ｗｈｅｎ」事象に関連するすべての「ｔｈｅｎ」事象に関して事象サーバ５０に対して要求および／または通知を送る。次に、事象サーバ５０はかかる要求を適当なソフトウエア・ツールに送る、最良の実施態様において、要求されたソフトウエア・ツールがまだ始動されていない場合、実行管理プログラム５１がこのソフトウエア・ツールを始動させる。したがって、メッセージ結合子ツール５６、実行管理プログラム５１および事象サーバ５０を介したソフトウエア・ツール間の通信はそれぞれのソフトウエア・ツール内で発生する事象によって駆動される。

【００１７】

上述した事象駆動される通信の一例としては、ユーザがテキスト編集ツール５３にファイルを保管することによって始動するものがある。テキスト編集ツール５３は事象サーバ５０にファイルが保管されたことを示す通知メッセージを送る。メッセージ結合子ツール５６は事象サーバ５０にかかつ通知をすべてメッセージ結合子ツール５６に送るように通知済みである。事象サーバ５０は編集ツールの通知をメッセージ結合子ツール５６に送る。メッセージ結合子ツール５６はテキスト編集ツール５３がファイルを保管するたびにスペル・チェック・メツール５７を実行しなければならないことを知っている。したがって、メッセージ結合子ツール５６は事象サーバ５０に事象をスペル・チェック・ツール５７に送るように要求する。実行管理プログラム５１はまずスペル・チェック・ツール５７がすでに実行中であり、事象を受け取る準備ができていることを確認する。スペル・チェック・ツール５７が稼働していない場合、実行管理プログラム５１はスペル・チェック・ツール５７を開始させる。事象サーバ５０はスペル・チェック・ツール５７に要求を送ってファイルのチェックを行なう。スペル・チェック・ツール５７はこの要求を実行し、それが完了すると事象サーバ５０に通知を送る。スペル・チェック・ツール５７からのこの事象通知は次に他のツール内の１つあるいはそれ以上の他の事象駆動される動作を開始させる。

【００１８】

図３は事象サーバ５０との通信の高レベルブロック図である。簡単なメニュー・インターフェースを介して、ユーザは新しい作業割り当てに対する支援あるいは新しいツール対話の自動化のための必要に応じたツール環境を変更することができる。本発明によれば、以上のことは、ユーザがツールを結合して所望のタスクを実行するメッセージ動作の集合であるルーチンを作成することによって達成される。かかるルーチンは、この事象が発生するとこの事象が起こるという「ｗｈｅｎ／「ｔｈｅｎ」関係に基づいている。ユーザはコードを書くこともなく、またツール自体を変更することもなくこれを行なうことができる。

【００１９】

ユーザはトップ・レベル・ユーザ・インターフェース、すなわち、ルーチン管理プログラム・ウィンド３１０からルーチンの作成を開始する。この目的のためのトップ・レベル・ユーザ・インターフェースの一例をルーチン管理プログラム・ウィンド３１０として図４に示す。このルーチン管理プログラム・ウィンドから、ユーザは『ＮＥＷ』（新ルーチン）ボタンあるいは『ＥＤＩＴ』（編集）ボタンのいずれかを選択し、メッセージサーバは、図８に示すようにルーチン編集プログラム・ウィンド３２０を呼び出す。ルーチン編集プログラム・ウィンド３２０はツール・ファイル３２２（図３）へのアクセスを有し、『Ｔｏｏｌ　Ｌｉｓｔ』（ツール・リスト）３２６によってユーザに使用可能なツールを表示する。ユーザはルーチンを作成した後、このルーチンをルーチン・ファイル３２４に保管し（図３）、ルーチン管理プログラム・ウィンド３１０に戻り、そのとき必要であれば『Ｅｎａｂｌｅ　Ｒｏｕｔｉｎｅ』（ルーチン・イネーブル）ボタンをクリックしてこの

ルーチンをイネーブルする。エンジン３３０がこのイネーブルされたルーチンを読み、事象サーバ５０によって「ｗｈｅｎ」事象に対する結合を確立してかかる「ｗｈｅｎ」事象メッセージがすべて事象サーバ５０によってエンジン３３０に送られるようにする。エンジン３３０はログ・ファイル３３２を維持する。エンジン３３０はログ・ファイル３３２にエンジンによって実行されるすべての動作を転記する。これによって事象履歴が維持される。

【００２０】

図６は図４から図６を参照して説明するユーザによるルーチンの作成の一例を示すフローチャートである。新しいルーチンの作成はステップ４０２で始まる。ユーザはルーチン管理プログラム・ウィンド３１０内の『ＮＥＷ』（新ルーチン）ボタンをステップ４０４で選択し、このルーチンに名前を付ける。これはこの実施態様ではルーチン管理プログラム・ウィンド３１０で行なわれるが、ルーチン編集プログラム・ウィンド３２０で行なうこともできる。ルーチンに名前を付けた後、ルーチン編集プログラムウィンド３２０が呼び出される。次に、ユーザは１つのツールを選択し、これを使用可能なツールのリスト３２６からの「ｗｈｅｎ」事象のためのツールとして使用する。ユーザが「ｗｈｅｎ」事象として使用するツールを選択した後、ルーチン編集プログラム・ウィンド３２０はツール動作リスト３２８に指定されたツールの動作をリストし、ユーザは「ｗｈｅｎ」事象のための特定のツール機能をステップ４０６で選択する。図５から、ツールの動作リスト（機能のリスト）はユーザには生のメッセージではなく人間にわかる抽象表現として表示される。次に、ユーザは『Ｃｏｐｙ　ｔｏ“ＷＨＥＮ：”』（「ｗｈｅｎ」にコピー）ボタン３３６をクリックすることによってこのツールと指定された機能を「ｗｈｅｎ」フィールド３３４に、ステップ４０８においてコピーする。ユーザは『Ｔｏｏｌ　Ｌｉｓｔ』（ツール・リスト）３２６から「ｔｈｅｎ」事象のツールとしてツールをステップ４１０で選択し、ステップ４１２において、そのツールの動作リスト３２８からこのツールの特定の動作が選択される。次に、ユーザは選択されたツールと指定された機能を『Ｃｏｐｙ　ｔｏ“ＴＨＥＮ：”』（「ｔｈｅｎ」にコピー）ボタン３４０をクリックすることによって「ｔｈｅｎ」フィールド３３８にステップ４１４でコピーする。また、ユーザは『２つの文字「ａｂ」で始まるファイルにのみ応答』あるいは『２つの文字「．ｔｘｔ」で終わるファイルにのみ応答』等の、所望のファイルに一致する任意のパターンを適用することができる。また、ユーザはメッセージのデータのうち任意のものに一致するパターンを適用することができる。ユーザはまたそのルーチン中に後に使用されるパターンに実際に一致するあらゆるフィールドを保管することができる。

【００２１】

このとき（ステップ４２８～４３０）、ユーザーは１つの「ｗｈｅｎ」事象によってトリガされる複数の「ｔｈｅｎ」事象を、複数の「ｔｈｅｎ」リスト３３９に追加し（複数ｔｈｅｎ領域内の『Ａｐｐｅｎｄ』（追加）ボタンをクリックする）、図のステップ４１０～４１６３を繰り返すことによって指定することができる。かかる複数の「ｔｈｅｎ」事象はユーザの必要に応じて『Ｓｅｒｉａｌ』（順次）ボタンあるいは『Ｃｏｎｃｕｒｒｅｎｔ』（並行）ボタンのいずれかをクリックすることによって順次にあるいは並行して実行することができる。ユーザーが自己の作成したルーチンに満足である場合、ユーザはフローチャートのステップ４１８に移行して『Ｓａｖｅ　Ｒｏｕｔｉｎｅ』（ルーチン保管）ボタン３４２をクリックしてそのルーチンを保管する。その結果、ユーザはルーチン管理プログラム・ウィンド３１０に戻り、このルーチン管理プログラム・ウィンドによってユーザは必要であれば『Ｅｎａｂｌｅ　Ｒｏｕｔｉｎｅ』（ルーチン・イネーブル）ボタン３１１をクリックしてこのルーチンをステップ４２０でイネーブルすることができる。ルーチンがイネーブルされるとルーチン管理プログラム・ウィンド３１０内でこのルーチンの隣に『ｅｎａｂｌｅｄ』（イネーブル）のワードが表示される。このとき、ユーザはルーチン管理プログラム・ウィンド３１０を出て、そのコンピュータ・セッションをステップ４２４で継続する。

【００２２】

図７および図８は本発明による新しいルーチンの作成時に発生する内部処理のフローチャートである。ユーザがルーチンのステップ４０２の作成と命名を開始するとき、ルーチン管理プログラム・ウィンド３１０はルーチン編集プログラム・ウィンド３２０を起動し、ユーザが使用することのできるツールを探索および表示する（ステップ５０１、５０２）。ユーザが「ｗｈｅｎ」事象のためのツールとしてあるツールを選択すると、ルーチン編集プログラム・ウィンド３２０はツール固有情報を読み（ステップ５０４）、その情報をルーチン編集プログラムのツールの動作リスト領域３２８においてユーザに表示する。次に、ステップ５０６において、ユーザが「ｗｈｅｎ」事象に対するある特定のツール事象を選択し、『Ｃｏｐｙ　ｔｏ“WHEN：”』（「ｗｈｅｎ」にコピー）ボタン３３６をクリックすると、ルーチン編集プログラム・ウィンドは選択された機能を「ｗｈｅｎ」領域にコピーおよび表示し、次にステップ５０８において選択されたツール機能に関連する生メッセージ・パターンをコピーする。ルーチン編集プログラム・ウィンド３２０においてユーザに表示されるのはツールおよびツール機能を人間の読むことのできる形態で表わす人間の言語による抽象表現だけであることに注意しなければならない。

【００２３】

次に、ユーザは「ｔｈｅｎ」事象ツールのためのツールを選択し、ルーチン編集プログラムはそのツールに固有の情報をステップ５１２で読み、この情報をそのツールの動作リスト領域３２８においてユーザに表示する。次に、ステップ５０６において、ユーザが「ｔｈｅｎ」事象に対するある特定のツール機能を選択し、『Ｃｏｐｙ　ｔｏ“ＴＨＥＮ：”』（「ｔｈｅｎ」にコピー）ボタン３４０をクリックすると、ルーチン編集プログラム・ウィンド３２０はこの機能を「ｔｈｅｎ」領域にステップ５１２でコピーおよび表示し、また選択されたツール機能に関連する生メッセージ・パターンを図８のステップ５１６でコピーする。次に、ユーザが『「Ａ　」で始まるファイルのみに応答』等のこのファイル名あるいはデータに一致する任意のパターンを加えることをステップ５１７で希望する場合、ルーチン編集プログラム・ウィンドはデフォルト情報をユーザの希望するものに置き換える。ユーザが任意の追加の「ｔｈｅｎ」事象を有する場合、ルーチン編集プログラムは必要に応じてこれらのステップを繰り返す。

【００２４】

ユーザが『Ｓａｖｅ　Ｒｏｕｔｉｎｅ』（ルーチン保管）ボタン３４２をクリックすると、ルーチン編集プログラムは表示からの「ｗｈｅｎ」ストリングと一時記憶場所に前にコピーされた生メッセージ・パターンをステップ５２０で読み、表示からの「ｔｈｅｎ」ストリングと一時記憶場所に前にコピーされた生メッセージ・パターンをステップ５２２で読み、一時記憶場所のデータをルーチン・ファイルに書き込む。この場合もユーザが見るのは生メッセージの人間の言語による抽象表現であることに注意しなければならない。したがって、ユーザはコンピュータ・コードやコンピュータのプログラム法を知っている必要はない。また、ユーザが生メッセージの実施態様を理解していることも必要ではない。ルーチン管理プログラム・ウィンドおよびルーチン編集プログラム・ウィンドはユーザがカスタム・ツール対話ルーチン処理を簡単に行なうことを可能にする。ユーザはソフトウエア・ツールのコードを変更せず、本発明のメッセージ結合子ツールを用いてツールの動作を変更することに注意しなければならない。

【００２５】

フローチャートの図９～図１１はユーザがルーチン管理プログラム・ウィンド３１０内の『Ｅｎａｂｌｅ　Ｒｏｕｔｉｎｅ』（ルーチン・イネーブル）ボタン３１１をクリックすることによってエンジン３３０を始動させる処理を示す。まず、ステップ５２８において、エンジンがまだ始動されていない場合、エンジンは事象サーバ５０との間に接続を確立し（ステップ５３０）、スタート・アップ・ルーチンの集合（ユーザが自動的にイネーブルしたいルーチンのリスト）を探す（ステップ５３２）。次に、エンジン３３０はイネーブル要求を待つ。イネーブル要求が検出されると（ステップ５３４）、エンジンはイネーブルすべき指定されたファイルを有するスレッドを開始し（ステップ５３６）、そのファイルが有効であるかどうかをステップ５４６で判定する。図１０のステップ５４６でファ

イルが有効であれば、エンジンはルーチンが必要とする可能性のある任意のパターン照合を行なうべき位置を表わす空変数集合をステップ５５２で作成する。次に、エンジンは一般的な有用な実施態様によって決まるストリングに割り当てられる初期値を表わす初期変数集合をステップ５５４で割り当てる。

【００２６】

次に、エンジンはルーチンファイルから「ｗｈｅｎ」事象をステップ５５６で読み出し、指定された「ｗｈｅｎ」事象の発生をステップ５５８で待機する。指定された「ｗｈｅｎ」事象がステップ５６０で検出されると、エンジンはそれが「待機」動作であったかどうかを図１１のステップ７０１で判定し、「待機」動作であった場合、エンジンは現在のルーチンの新しいスレッドをステップ７０２で開始する。「待機」動作でなかった場合、エンジンは対応する「ｔｈｅｎ」事象をステップ７０３で読み出し、指定されたメッセージをステップ７０４で事象サーバに送る。このメッセージがステップ７０５で要求である場合、エンジンはステップ７０６で応答を待つ。応答が障害である場合（ステップ７０６）、エンジンはメッセージ・サーバに誤り通知メッセージをステップ７０８で送る。最後に、エンジンは検出された「ｗｈｅｎ」事象に関連する「ｔｈｅｎ」事象がまだあるかどうかをステップ７０９でチェックする。ある場合、エンジンは次の「ｔｈｅｎ」事象を読み出す。

【００２７】

［エンジン］

本発明はユーザがコードを一切書くことなくソフトウエア・ツールの実行と統合を簡単に自動化および制御する方法を提供する。これを行なうためのルーチン管理プログラム・ウィンドとルーチン編集プログラム・ウスインドを上に説明した。上記の目的を達成するための本発明の他の重要な要素は図３に示すエンジン３３０である。エンジン３３０は事象サーバ５０および事象サーバ５０を介して他のソフトウエア・アプリケーション・ツール５２～５７に接続されている。エンジンは事象サーバ５０を用いてこれらのソフトウエア・ツールによって生成される事象をモニタし、対象となる事象がどれであるかを判定し、ソフトウエア・ツール自体を変更することなくソフトウエア・ツールの動作を変更するために事象および事象内容に基づいてルーチンを実行する。

【００２８】

このエンジン内においては、「０」、「１」あるいはそれ以上のルーチンを並行して実行することができることに注意しなければならない。あるルーチンのコピーが活動状態であり、実行中であるとき、これをスレッドと呼ぶ。エンジン３３０は（１）事象を待つ、あるいは（２）事象サーバ５０を介して１つあるいはそれ以上のソフトウエア・ツール内で動作を発生させることのできる他の事象を生成する。エンジン３３０は、また（３）他の動作シーケンスの同時実行を発生させる、（４）動作シーケンスを起動あるいは起動解除する、あるいは（５）（１）、（２）、（３）、（４）および（５）の任意の組み合わせの同時実行を発生させることができる。

【００２９】

ここで図１２を参照してこのエンジンについてより詳細に説明する。エンジン３３０はスレッド構文解析系６１０、スレッドツリー構造６０６、スレッド変数構造６０８、スレッド・ツリー解釈プログラム６０４、およびエンジン制御プログラム６０２を有する。

【００３０】

エンジン３３０は標準的なアプリケーション・ツールと同様にコンピュータ・オペレーティング・システムと通信する。本発明によれば、エンジン３３０はルーチン・ファイル３２４を用いて簡単に生成される。ルーチン・ファイル３２４はユーザが個々のアプリケーション・ツールとの通信およびアプリケーション・ツール間の通信を定義することを可能にする。ルーチン・ファイル３２４はスレッド構文解析系６１０がスレッド・ツリー構造６０６、スレッド変数構造６０８によって用いられる。本発明によるすべての可能なルーチン・ファイル・フォーマットの記述のためのＢａｃｋｕｓ　Ｎａｕｒ　Ｆｏｒｍ（バッカス・ナウル・フォーム）を付録Ａ　として添付する。

【００３１】
以下はスペル・チェック・ツールのためのルーチン・ファイルの一例である。
【００３２】
【表１】

```
StandardRoutine( 「B-MCROUTINE 」 ) / /Spell__Check.mcr
 {  // OpArray
      WaitForNotify( 「Editor Edit Succeeded: name:*NONE-REQUIRED」
"EDIT" "$HOST" "$DIR" "name":"*" "WINDOW" "PASS' )
        {  // DataCaseArray
            DataCase("*")
             {  // OpArray
            SendRequest("Spell Check $name NONE-REQUIRED" "SPELL"
"CHEK" "$name"  "")
                        {  // ReplyCaseArrsy
                          ReplyCase("FALL" "*")
                             {  // OpArray
                            SendNotify("" "EXECUTING-ROUTINE" "$RNAME"

'FAIL" "Request Compare to__Set __left__file NONE-REQUIRED NONE
REQUIRED ---> R SB-COMPARE SET-LEFT-FILE *")
                                    }
                        }
                }
        }
}
```

【００３３】
図１３は本発明のエンジンの可能な機能を示すフローチャートである。まず、エンジン制御プログラム６０２がステップ６２２においてエンジン要求事象が存在することを検出する。エンジン制御プログラム６０２はステップ６２４においてこの要求がイネーブル・ルーチンであるかどうかを判定する。この要求がイネーブル・ルーチンである場合、エンジン制御プログラム６０２はステップ６２６においてこのルーチンがすでにイネーブルされているかどうかを判定する。このルーチンがすでにイネーブルされている場合、エンジン

制御プログラム６０２はステップ６２８において指定されたルーチンのすべてのスレッドを破壊させ、次にステップ６３０において指定されたルーチンの新しいスレッドを開始する。あるいはステップ６３０において指定されたルーチンの新しいスレッドが単に開始される。次に、ステップ６３２においてエンジン制御プログラム６０２が事象を送って、ルーチン管理プログラム・ウィンドにこのルーチンがイネーブルされていることを通知する。次に、エンジン制御プログラムはステップ６２２に戻り、アイドル状態でルーチン管理プログラム・ウィンドからの他の要求を待つ。

【００３４】

ステップ６２４において、要求がイネーブル要求ではないと判定された場合、エンジン制御プログラム６０２はステップ６３４においてこの要求が不能化ルーチン要求であるかどうかを判定する。不能化ルーチン要求である場合、エンジン制御プログラム６０２はステップ６３６において指定されたルーチンのすべてのスレッドを破壊し、ステップ６３８においてルーチン管理プログラム・ウィンドにルーチン不能化通知を送り、ステップ６２２に戻り、ルーチン管理プログラム・ウィンドからの他の要求を待つ。

【００３５】

この要求がイネーブル要求でも不能化要求でもない場合、エンジン制御プログラム６０２はこの要求がルーチン照会要求であるかどうかをステップ６４０で判定し、ルーチン照会要求である場合、エンジン制御プログラム６０２はステップ６４２において１つあるいはそれ以上の活動状態のスレッドを有するそれぞれのルーチンについて、ルーチン・イネーブル済み通知事象をルーチン管理プログラム・ウィンドに送る。次に、エンジン制御プログラム６０２はステップ６２２に戻り、ルーチン管理プログラム・ウィンドからの他の要求を待つ。

【００３６】

この要求がイネーブル要求でも、不能化要求でも、ルーチン照会要求でもない場合、次に、エンジン制御プログラム６０２はステップ６４４においてこの要求がロギング設定要求であるかどうかを判定し、ロギング設定要求である場合、エンジン制御プログラム６０２はログファイルが指定されているかどうかをステップ６４６で判定する。指定されていない場合、エンジン制御プログラム６０２はステップ６４８においてロギングを不能化し、現在のログファイルを閉じる。ログファイルが指定されている場合、エンジン制御プログラム６０２はステップ６５０においてロギングをイネーブルし、空ログファイルを作成する。

【００３７】

上記の要求のいずれも検出されない場合、エンジン制御プログラム６０２は要求が停止要求であるかどうかをステップ６５２で判定し、停止要求である場合、エンジン制御プログラムは既存のスレッドをステップ６５６ですべて破壊してこれらをステップ６５８で停止する。要求が停止要求でない場合、エンジン制御プログラム６０２はステップ６５６においてルーチン管理プログラムに無効要求通知を送り、ステップ６２２に戻り、ルーチン管理プログラム空きの他の要求を待つ。

【００３８】

図１４はこのエンジンの典型的な用途を示すフローチャートである。図１３および図１４において、要求が（図１３のステップ６２４に示すように）ルーチン・イネーブル要求である場合、ステップ６６０でエンジン制御プログラムがステップ６３０で指定されたプログラムの新しいスレッドを作成した後、スレッド構文解析系６１０がステップ６６２において指定されたルーチン・ファイルをスレッド・ツリー構造に解析する。スレッド構文解析系６１０はまたステップ６６４においてスレッド変数構造６０８内にスレッド変数のためのスペースを割り当てる。次に、スレッド構文解析系６１０はステップ６６６において標準変数の値を設定する。次に、スレッド・ツリー解釈プログラム６０４はツリー中のすべての動作が網羅されているかどうかをチェックする。

【００３９】

ツリーにまだ実行すべき動作がある場合、スレッド・ツリー解釈プログラム６０４はステ

ップ６７６においてツリー内の次の動作を実行する。次に、解釈プログラム６０４はステップ６７８においてロギングがイネーブルされているかどうかを判定する。ロギングがイネーブルされている場合、スレッド・ツリー解釈プログラム６０４はログ・ファイル３３２に動作ログ情報を書き込む。次に、スレッド・ツリー解釈プログラム６０４は実行された動作が最初の動作であったかどうかを判定する。最初の動作であった場合、スレッド・ツリー解釈プログラム６０４はこの動作が通知待ちであるか要求待ちであるかどうかを判定し、そのいずれであっても、スレッド・ツリー解釈プログラム６０４はステップ６８６においてルーチン・ファイル３２４内の現在のルーチンの新しいスレッドを開始する。あるいは、解釈プログラム６０４はステップ６６８に戻り、ツリー内のすべての動作が網羅されるまでステップ６６８～６８６を繰り返す。

【００４０】

ツリー内のすべての動作が網羅されると、スレッド・ツリー解釈プログラム６０４は実行中のスレッドがルーチンの唯一のスレッドであるかどうかをステップ６７０で判定し、唯一のスレッドである場合、そのスレッドは完了し破壊される。次に、エンジンはルーチン管理プログラム・ウィンド３１０をルーチン不能化済み通知を送り、ステップ６７２で停止する。

【００４１】

当業者には、このエンジンは本発明のメッセージ結合子ツールをイネーブルする以外にも多くの機能を有することは明らかであろう。たとえば、このエンジンを用いて、前述の例で説明したようにメッセージ事象の検出と生成を行なう代わりに、ソフトウエア・ツールのユーザ・ウィンド内のユーザ・インターフェース事象の検出と生成を行なうことができる。この代替実施態様を用いると、環境内の他のツールに機能が実行されたことを通知する事象を生成しないツールを用いても同様のツール対話を達成することができる。この代替実施態様からわかるように、このエンジンはメッセージ・サーバを用いるシステムに限定されず、なんらかの形態の事象送出機構を用いる任意のコンピュータ・システム上で動作する。

【００４２】

本発明の以上の説明は例示および説明のためにのみ行なったものである。以上の説明は本発明を網羅するものでもまたそれをここに開示した形態に限定するものでもなく、以上の開示内容に照らして他の変更態様を実施することができる。たとえば、本発明はメッセージ・サーバを用いたコンピュータ・システムに限定されず、当業者には明らかなように要求および／または事象の通知を用いる任意のコンピュータ・システムにおいて実施することができる。また、メッセージ結合子ツールは２つ以上の「ｗｈｅｎ」事象によってトリガすることができる。また、１つあるいは２つ以上の「ｗｈｅｎ」事象、他のルーチンの１つあるいは２つ以上の「ｔｈｅｎ」事象、１つあるいは２つ以上の非ツール事象、あるいはこれらの事象の任意の組み合わせによってトリガすることができる。

【００４３】

本発明の実施例は原理とその実施を、当業者が本発明を具体的な用途に適したさまざまな変更態様で利用しうるように、最もよく説明できるように選択および説明されたものである。特許請求の範囲には、従来の技術によって限定されない限り本発明の例外の代替実施例を含むものである。

【００４４】

次にソフトウエア・ツールの実行のプログラム例を例示する。

【００４５】

【表２】

```
Routine:
        RoutineList
      | StandardRoutine
      | ServiceRoutine
    ;
RoutineList:
        "RoutineList"RoutineNameArray
    ;
StandardRoutine:
        "StandardRoutine("StringPattern=toolclass')'OpArray
    ;
opArray:'  {'opList'}'  |[];
opList:
       Op opList
     | []
    ;
```

【００４６】
【表３】

(表２つづき１)

```
Op:
        SendRequest
      | SendNotify
      | WaitForNotify
      | WaitForNotify
      | Enable
      | Disable
      | Concurrent
      | SendReply
    ;
SendRequest:
        "SendRequest("StringPattern=comment ValuePattern=toolclass
        ValuePattern=action ValuePattern=operand ValuePattern=data')'
        ReplyCaseArray
    ;
ReplyCaseArray:
       ' {'replyCaseList' }' | [];
replyCaseList:
        []
    |   ReplyCase replyCaseList
    ;
```

【００４７】
【表４】

（表２つづき２）

```
ReplyCase:
        "ReplyCase("Pattern=status DataPattern=data')' OpArray
    ;
SendNotify:
        "SendNotify("StringPattern=comment ValuePattern=action
        ValuePattern=operabd ValuePattern=status ValuePattern=data')'
    ;
WaitForNotify:
        "waiteForNotify("StringPattern=comment Pattern=toolclass
        Pattern=host Pattern=dir Pattern=operand Pattern=action')'
        DataCaseArray
    ;
WaitForNotify:
        "WaitForNotify("StringPattern=comment Pattern=toolclass
        Pattern=host Pattern=dir Pattern=operand Pattern=action
        Pattern=status')' DataCaseArray
    ;
```

【００４８】
【表５】

(表２つづき３)

```
DataCaseArray:
        '   { DataCaseArraytail '} '
    |   []
    ;
DataCaseArraytail:
        []
    |   Datacase DataCaseArraytail
    ;
DataCase:
        "Datacase("DataPattern=data ')' OpArray
    ;
Enable:
        "Enable("RoutineName ')'
    ;
Disable:
        "Disable("RoutineName ')'
    ;
Concurrent:
        "Concurrent" ConcurrentArray
    ;
```

【００４９】
【表６】

(表２つづき４)

```
ConcurrentArray:
        ' {' concurrentList ' } '  | [];
concurrentList:
        Op concurrentList
     |  []
    ;
RoutineNameArray:
        '  {' RoutineNameArraytail '} '
    ;
RoutineNameArraytail:
        []
     |  RoutineName RoutineNameArraytail
    ;
/ * Note: the exact form of a PATTERN is described elsewhere* /
RoutineName: PATTERN
    ;
DataPattern: PATTERN
    ;
Pattern: PATERN
    ;
```

【００５０】
【表７】

（表２つづき５）

```
ValuePattern: PATTERN
    ;
StringPattern: PATTERN
    ;
ServiceRoutine:
    "ServiceRoutine("StringPattern=toolclass StringPattern=action ')'
     ServiceCaseArray
    ;
ServiceCaseArray
    ;   '   {'serviceCaseList '}'
     |   []
    ;
serviceCaseList: ServiceCaseList
     ;
ServiceCase:
    "ServiceCase(" PATTERN=operand DataPattern=data ')'OpArray
    ;
// only legal within a ServiceRoutine. Should also ideally do path analy
// sis to ensure only one reply is possible :
SendReply: "SendReply("ValuePattern=satus ValuePattern=data ')'
    :
```

【００５１】

以上、本発明の各実施例について詳述したが、ここで各実施例の理解を容易にするために、各実施例を要約して、以下に列挙す。

【００５２】

１．　電子デジタル・コンピュータと、前記のコンピュータを動作させるための、事象ディスパッチング機構を介して互いに通信するための手段を有する複数のソフトウエア・ツールからなる、メモリを含むコンピュータ・ソフトウエアからなるコンピュータ・システム内においてソフトウエア・ツールの実行自動化および制御する方法であって、

人間の言語による抽象表現を用いて少なくとも１つのツール対話ルーチンを作成するステップであって、前記のツール対話ルーチンはユーザの定義する規則の集合であり、ユーザの指定する事象が発生すると、少なくとも１つのユーザによって指定される動作が開始されるステップ、

前記の少なくとも１つのツール対話ルーチンの前記の人間の言語による抽象表現に関連する生のメッセージを、一連の機械の実行可能な動作からなるルーチンファイルに変換するステップ、
少なくとも１つのツール対話ルーチンをイネーブルするステップ、
前記のユーザにより指定される事象の発生を検出するステップ、
前記のユーザにより指定される事象が発生すると、前記の少なくとも１つのユーザの指定する動作を開始するステップ、
からなるコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００５３】

２．　少なくとも１つのツール対話ルーチンを作成するステップが、
ｗｈｅｎ　事象として使用すべき少なくとも１つのユーザの指定するツール機能を選択するステップ、
ｔｈｅｎ　事象として使用すべき少なくとも１つのユーザの指定するツール機能を選択するステップからなる前記１に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００５４】

３．　少なくとも１つのツール対話ルーチンを作成するステップが、
ｗｈｅｎ　事象として使用すべき少なくとも１つのユーザの指定する非ツール事象を選択するステップ、
ｔｈｅｎ　事象として使用すべき少なくとも１つのユーザの指定するツール機能を選択するステップからなる前記１に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００５５】

４．　少なくとも１つのツール対話ルーチンを作成するステップが、さらに前記のツール対話ルーチンにユーザの定義するパターン照合を適用するステップを含む前記２に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００５６】

５．　少なくとも１つのツール対話ルーチンを作成するステップが、さらに前記のツール対話ルーチンにユーザの定義するパターン照合を適用するステップを含む前記３に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００５７】

６．　前記の少なくとも１つのユーザの指定する動作を開始するステップが、前記の開始すべき少なくとも１つの動作に対する少なくとも１つの要求事象を送るステップ、
前記の開始すべき少なくとも１つの動作に対する前記の少なくとも１つの要求事象を受け取るツールを含む前記１に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００５８】

７．　前記の少なくとも１つのユーザの指定する動作を開始するステップが、前記の開始すべき少なくとも１つの動作に対する少なくとも１つの通知事象を送るステップ、
前記の開始すべき少なくとも１つの動作に対する前記の少なくとも１つの通知事象を受け取るツールを含む前記１に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００５９】

８．　コンピュータシステム内において複数のソフトウエアツールの実行を自動化および制御する装置であって、
中央処理装置と前記のコンピュータを動作させるための複数のソフトウエアツールからなるコンピュータソフトウエアを格納した記憶装置を含む電子デジタルコンピュータ、

前記の複数のソフトウエアツールの間で事象デイスパッチングするための事象サーバ、
ユーザーの定義した規則の集合からなるユーザの指定するルーチンを作成および編集し、これによって少なくとも１つのユーザの指定する事象が発生すると、少なくとも１つのユーザの指定する動作が開始されるルーチン編集プログラム、前記のルーチン編集プログラムによって作成される任意のルーチンを管理するためのルーチン管理プログラム、
前記の人間の言語による抽象表現に関連する生のメッセージを、一連の機械の実行可能な動作からなるルーチン・ファイルに変換するエンジン、
ユーザの指定する事象の発生をモニタし、前記のユーザの指定する事象に対応する任意のユーザの指定する動作に対する要求メッセージを前記の事象サーバに通信するメッセージ結合子からなるコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法に適用されるソフトウエア・ツールの実行自動化および制御装置である。

【００６０】

９．　コンピュータシステム内において複数のソフトウエアツールの実行を自動化および制御する装置であって、さらに、
ある機能が要求されたとき実行されていない任意のソフトウエアツールを始動させる実行管理プログラムを有する前記８に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法に適用されるソフトウエア・ツールの実行自動化および制御装置である。

【００６１】

１０．　電子デジタルコンピュータと、前記のコンピュータを動作させるための、事象デイスパッチング機構を介して互いに通信するための手段を有する複数のソフトウエアツールからなる、メモリを含むコンピュータソフトウエアからなるコンピュータシステム内においてソフトウエアツールの実行を自動化および制御する方法であって、
ユーザが人間の言語による抽象表現を用いてルーチン編集プログラムによって提供される少なくとも１つのツール対話ルーチンを作成するステップであって、前記のツール対話ルーチンはユーザの定義する規則の集合であり、ユーザの指定する事象が発生すると、少なくとも１つのユーザによって指定される動作が開始されるステップ、
エンジンが前記の少なくとも１つのツール対話ルーチンの前記の人間の言語による抽象表現に関連する生のメッセージを、一連の機械の実行可能な動作からなるルーチン・ファイルに変換するステップ、
ユーザがルーチン管理プログラムを用いて少なくとも１つのツール対話ルーチンをイネーブルするステップ、
メッセージ結合子ツールが前記のユーザによりイネーブルされた前記の少なくとも１つのツール対話ルーチンのユーザによって指定される事象を検出するステップ、
前記のユーザにより指定される事象が発生すると、前記のエンジンが前記の少なくとも１つのユーザの指定する動作を開始するステップからなる前記１０に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００６２】

１１．　前記のユーザの指定する事象に関連する前記の少なくとも１つのユーザの指定する動作を開始するステップが、
メッセージ結合子ツールが事象サーバに適当なツールに前記の少なくとも１つのユーザの指定する動作の開始を要求するステップ、
前記の事象サーバが前記の適当なツールに前記の少なくとも１つのユーザの指定する動作の開始を命令するステップ、
前記の適当なツールが前記の少なくとも１つのユーザの指定する動作を開始するステップを含む前記１１に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００６３】

１２．　前記のユーザの指定する事象に関連する前記の少なくとも１つのユーザの指定する動作を開始するステップが、さらに、

前記の事象サーバが前記の適当なツールに前記の少なくとも１つのユーザの指定する動作の開始を命令するとき前記の適当なツールが稼働していない場合、実行管理プログラムが前記の適当なツールを始動させるステップを含む前記１１に記載のコンピュータ・システムにおけるソフトウエア・ツールの実行自動化および制御方法である。

【００６４】

【発明の効果】

以上のように、本発明によれば、ユーザの定義する規則の集合でユーザの指定する事象が発生すると少なくとも１つのユーザによって指定される動作が開始する少なくとも１つのツール対話ルーチンを人間の言語による抽象表現を用いて作成し、ユーザにより事象を指定すると、少なくとも１つの動作を開始するとともに、少なくとも１つのツール対話ルーチンの人間の言語による抽象表現に関連する生のメッセージを一連の機械の実行可能な動作からなるルーチン・ファイルに変換するようにしたので、ユーザが任意のツールを図形を用いて選択でき、そのツールから検出したい事象を選択し、その選択した事象を用いて他の任意のソフトウエア・ツールの機能を呼び出すことができる。

しかも、ユーザはこの機能の結合をソフトウエアのプログラミングの知識を必要とすることなく行うことができるとともに、「ｗｈｅｎ」事象を選択して、「ｔｈｅｎ」動作を非常に短時間で起動することができる。

さらに、１つのタスクの実行に際して、複数のユーザ／ツール対話を省くことができる。

【図面の簡単な説明】

【図１】本発明のメッセージ・サーバを介したツール機能間の変更可能な統合を示したツール・セットのブロック図である。

【図２】本発明のメッセージ・サーバとメッセージ結合子ツールおよび各種のソフトウエア・ツールの間の双方向の通信路を示すコンピュータ環境の概念ブロック図である。

【図３】本発明のメッセージ・サーバを用いたルーチン作成、格納および通信を示すブロック図である。

【図４】本発明によるツール・ルーチン管理プログラムインターフェースのためのユーザー・インターフェース・ウィンド表示の一例を示す説明図である。

【図５】本発明によるツール・ルーチン編集プログラム・インターフェースのためのユーザ・インターフェース・ウィンド表示の一例を示す説明図である。

【図６】本発明によるルーチン作成においてユーザの実行するステップの一例を示すフローチャートである。

【図７】本発明による新しいルーチンの作成に用いられる内部システムのステップを示すフローチャートである。

【図８】本発明による新しいルーチンの作成に用いられる内部システムのステップを示すフローチャートである。

【図９】本発明による前に作成されたルーチンをイネーブルするのに用いられる内部システムのステップを示すフローチャートである。

【図１０】本発明による前に作成されたルーチンをイネーブルするのに用いられる内部システムのステップを示すフローチャートである。

【図１１】本発明による前に作成されたルーチンをイネーブルするのに用いられる内部システムのステップを示すフローチャートである。

【図１２】本発明のエンジンの内部動作および他の構成要素を示すブロック図である。

【図１３】本発明によるエンジンの可能な動作を示すフローチャートである。

【図１４】本発明によるエンジンの典型的な用途を示すフローチャートである。

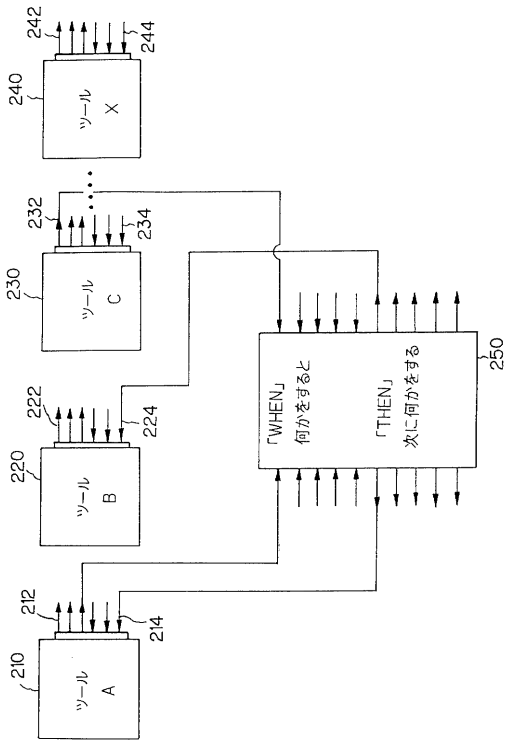
【図１５】コンピュータシステムのアプリケーション・ツール・セットのブロック図である。

【図１６】それぞれのツールの機能へのプログラムによるアクセスを示したツール・セットのブロック図である。

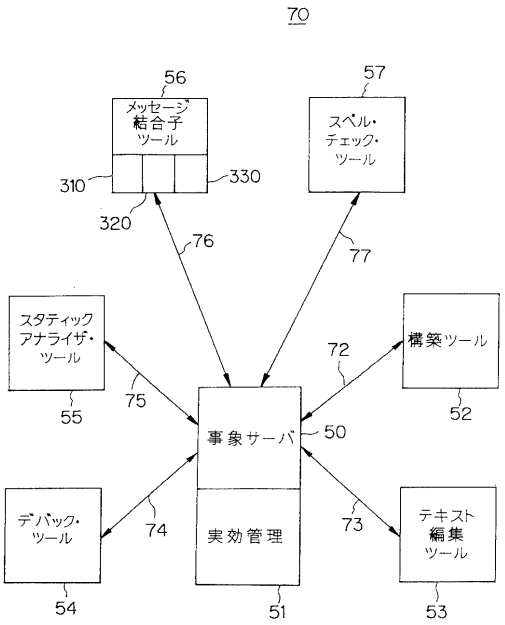
【図１７】異なるツールの機能の間の固定された結合を示したツールセットのブロック図である。

【符号の説明】
５０　事象サーバ
５１　実行管理プログラム
５２　構築ツール
５３　テキスト編集ツール
５４　デバッグ・ツール
５５　スタティック・アナライザ
５６、２５０　メッセージ結合ツール
５７　スペル・チェック
７０　計算環境
７３～７７　ソケット
２１０　ツールＡ
２２０　ツールＢ
２３０　ツールＣ
２４０　ツールＸ
３１０　ルーチン管理プログラム・ウィンド
３２０　ルーチン編集プログラム・ウィンド
３２２　ツール・ファイル
３２４　ルーチン・ファイル
３２６　ツール・リスト
３２８　ツール動作リスト
３３０　エンジン
３３２　ログ・ファイル
３３４　「ｗｈｅｎ」フィールド
３３６　『「ｗｈｅｎ」にコピー』ボタン
３３８　「ｔｈｅｎ」フィールド
３３９　「ｔｈｅｎ」リスト
３４０　『「ＴＨＥＮ」にコピー』ボタン
３４２　「ルーチン保管」ボタン
４０４　「ｗｈｅｎ」事象
４０６　「ｗｈｅｎ」事象
４２４　コンピュータ・セッション
６０２　エンジン制御プログラム
６０４　スレッド・ツリー解釈プログラム
６０６　スレッド・ツリー構造
６０８　スレッド変数構造
６１０　スレッド構文解析系

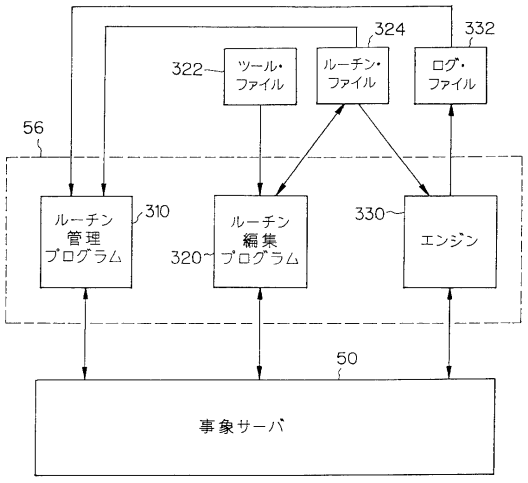
【図１】


242
244
240
ツール X
232
234
230
ツール C
222
224
220
ツール B
212
214
210
ツール A
250
「WHEN」 何かをすると
「THEN」 次に何かをする

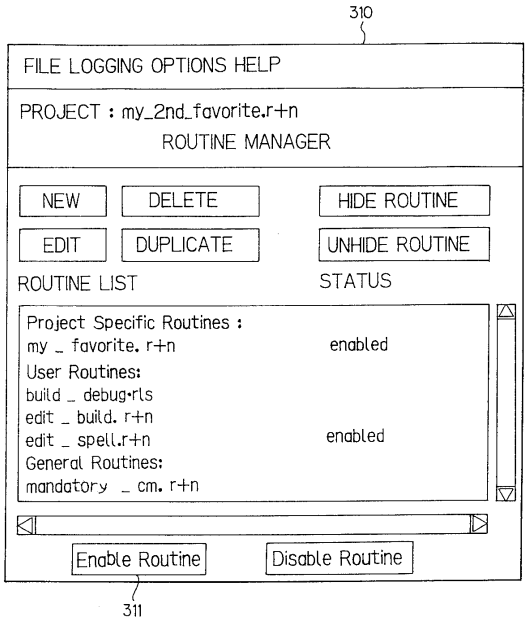
【図２】


70
56
メッセージ結合子ツール
57
スペル・チェック・ツール
310
330
320
76
77
スタティックアナライザ・ツール
構築ツール
72
75
55
52
事象サーバ
50
デバック・ツール
74
実効管理
73
テキスト編集ツール
54
51
53

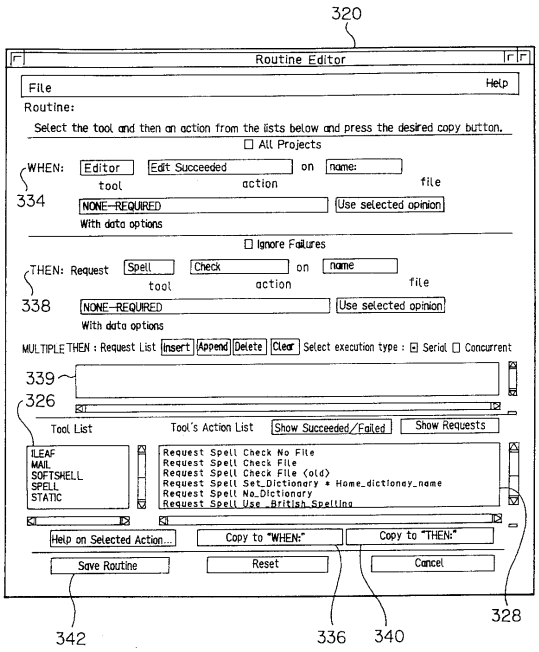
【図３】


324
332
322
ツール・ファイル
ルーチン・ファイル
ログ・ファイル
56
ルーチン管理プログラム
310
ルーチン編集プログラム
320
330
エンジン
50
事象サーバ

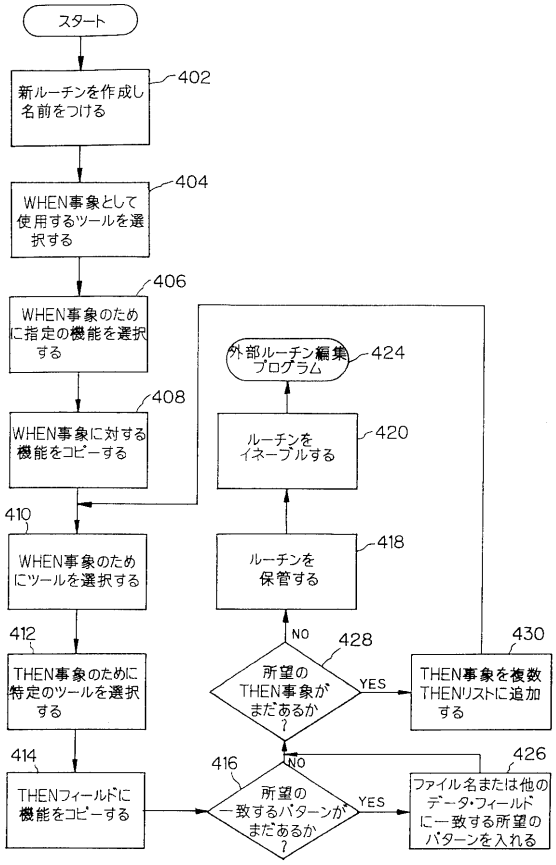
【図４】


310
FILE LOGGING OPTIONS HELP
PROJECT : my_2nd_favorite.r+n
ROUTINE MANAGER
NEW
DELETE
HIDE ROUTINE
EDIT
DUPLICATE
UNHIDE ROUTINE
ROUTINE LIST
STATUS
Project Specific Routines :
my _ favorite. r+n
enabled
User Routines:
build _ debug•rls
edit _ build. r+n
edit _ spell.r+n
enabled
General Routines:
mandatory _ cm. r+n
Enable Routine
Disable Routine
311

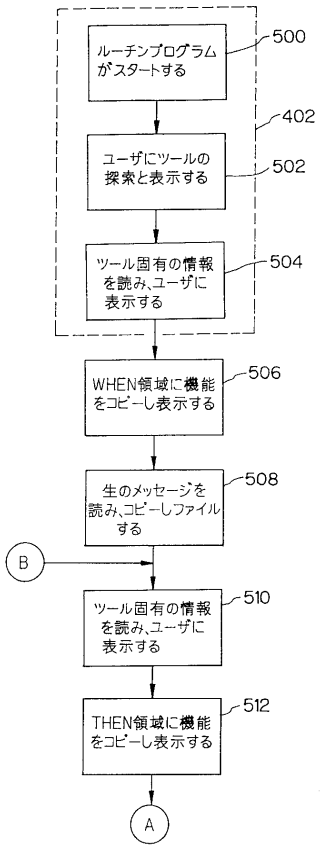
【図５】


320
Routine Editor
File
Help
Routine:
Select the tool and then an action from the lists below and press the desired copy button.
All Projects
WHEN:
Editor
Edit Succeeded
on
name:
tool
action
file
334
NONE-REQUIRED
Use selected opinion
With data options
Ignore Failures
THEN: Request
Spell
Check
on
name
tool
action
file
338
NONE-REQUIRED
Use selected opinion
With data options
MULTIPLE THEN : Request List
Insert
Append
Delete
Clear
Select execution type : Serial Concurrent
339
326
Tool List
Tool's Action List
Show Succeeded/Failed
Show Requests
ILEAF
MAIL
SOFTSHELL
SPELL
STATIC
Request Spell Check No File
Request Spell Check File
Request Spell Check File (old)
Request Spell Set_Dictionary + Home_dictionay_name
Request Spell No_Dictionary
Request Spell Use British Spelling
Help on Selected Action...
Copy to "WHEN:"
Copy to "THEN:"
Save Routine
Reset
Cancel
342
336
340
328

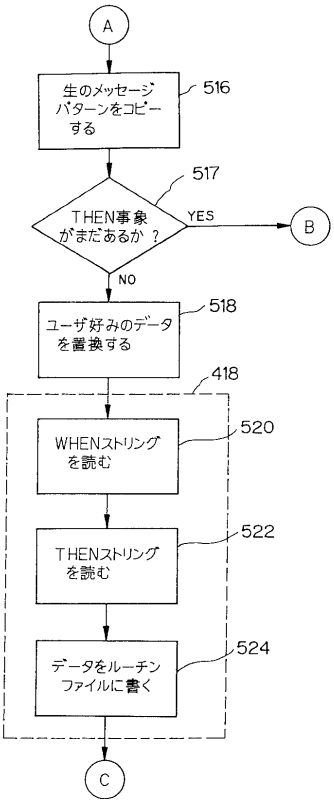
【図６】


スタート
新ルーチンを作成し名前をつける
402
WHEN事象として使用するツールを選択する
404
WHEN事象のために指定の機能を選択する
406
WHEN事象に対する機能をコピーする
408
WHEN事象のためにツールを選択する
410
THEN事象のために特定のツールを選択する
412
THENフィールドに機能をコピーする
414
所望のTHEN事象がまだあるか？
428
NO
YES
THEN事象を複数THENリストに追加する
430
所望の一致するパターンがまだあるか？
416
NO
YES
ファイル名または他のデータ・フィールドに一致する所望のパターンを入れる
426
ルーチンをイネーブルする
420
外部ルーチン編集プログラム
424
ルーチンを保管する
418

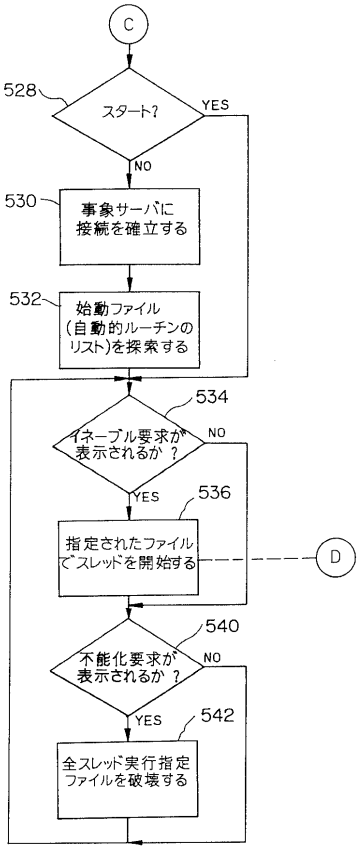
【図７】


ルーチンプログラムがスタートする
500
402
ユーザにツールの探索と表示する
502
ツール固有の情報を読み、ユーザに表示する
504
WHEN領域に機能をコピーし表示する
506
生のメッセージを読み、コピーしファイルする
508
B
ツール固有の情報を読み、ユーザに表示する
510
THEN領域に機能をコピーし表示する
512
A

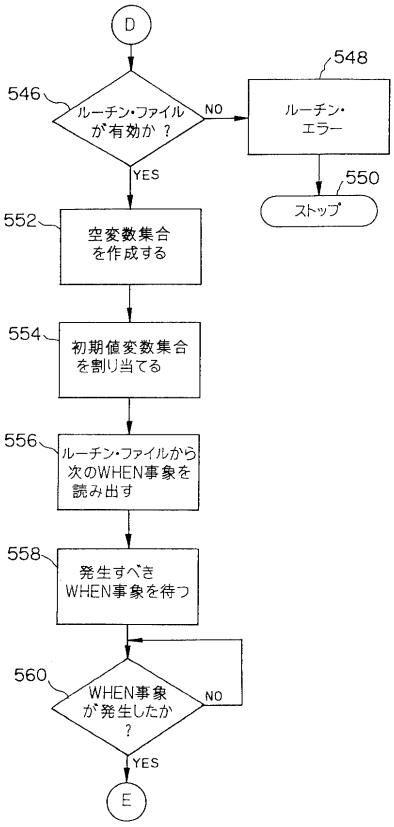
【図８】


A
生のメッセージパターンをコピーする
516
THEN事象がまだあるか？
517
YES
B
NO
ユーザ好みのデータを置換する
518
418
WHENストリングを読む
520
THENストリングを読む
522
データをルーチンファイルに書く
524
C

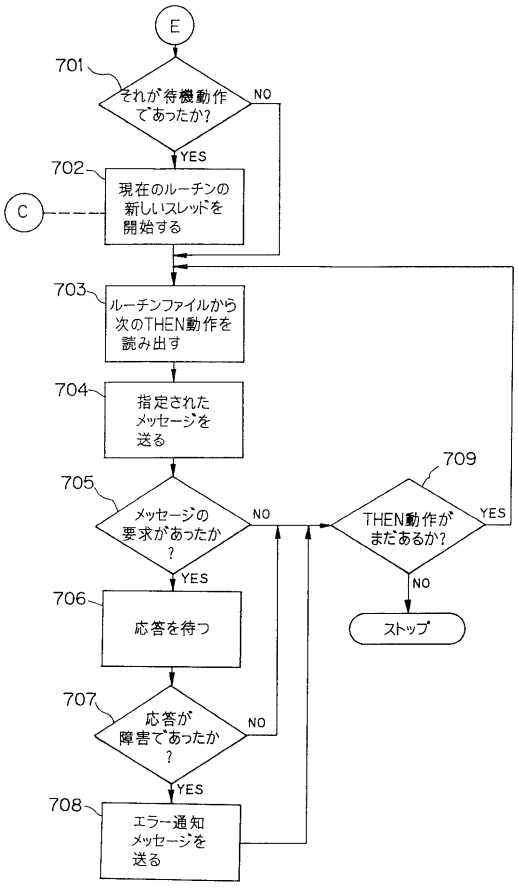
【図９】



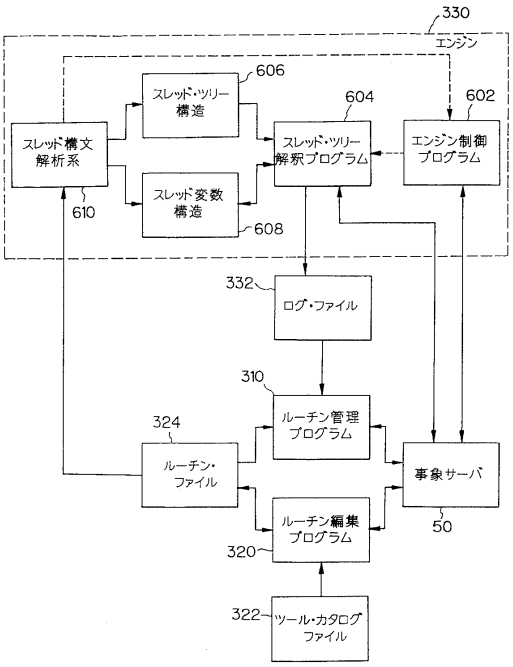
【図１０】



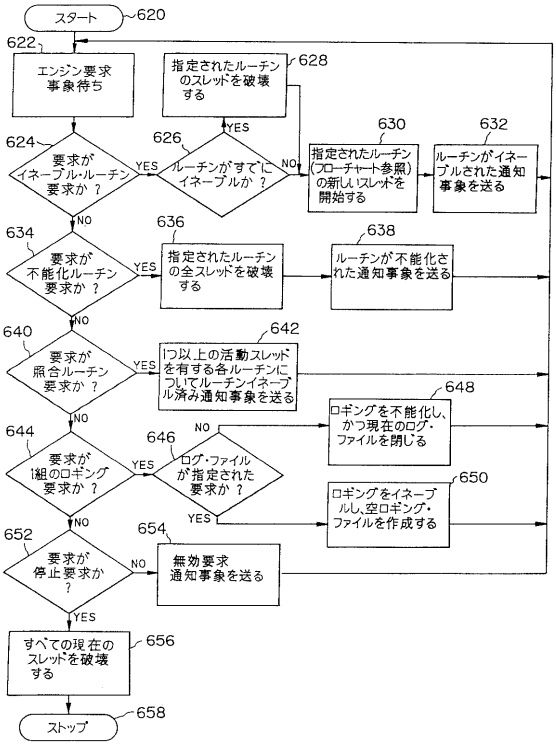
【図１１】



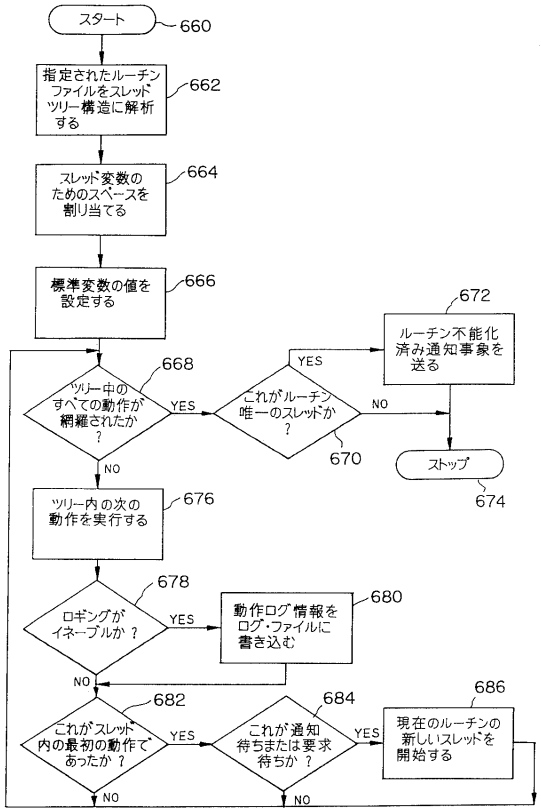
【図１２】



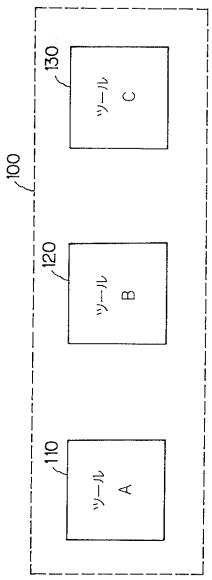
【図１３】



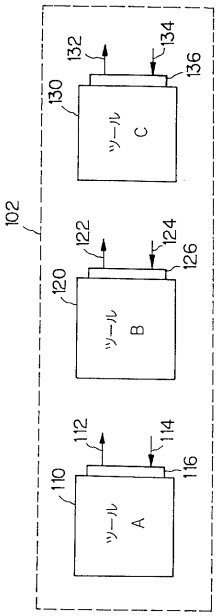
【図１４】



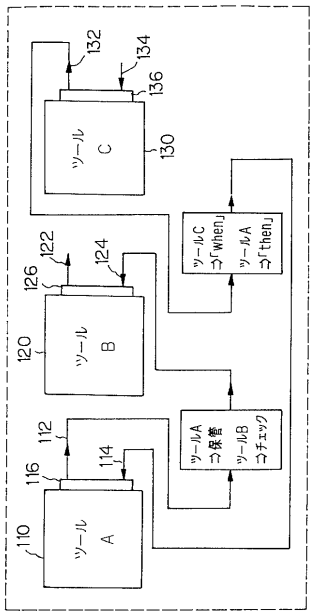
【図１５】



【図１６】



【図１７】


ツール A
ツール B
ツール C
110
112
114
116
120
122
124
126
130
132
134
136
ツールA ⇒保管 ツールB ⇒チェック
ツールC ⇒「when」 ツールA ⇒「then」

フロントページの続き

(72)発明者 バイロン・ティー・ジェニングス・ジュニア
アメリカ合衆国コロラド州フォートコリンズ、アパートメント ５１６、サウス・レメイ ４４７０

審査官 久保 光宏

(56)参考文献 特開昭６３－３１７８１５（ＪＰ，Ａ）
特開平３－３４０１８（ＪＰ，Ａ）
特開平２－２７２６３２（ＪＰ，Ａ）
特開平２－１４４７５４（ＪＰ，Ａ）
特開平２－１９３２４６（ＪＰ，Ａ）
特開平６－３４８６９８（ＪＰ，Ａ）
特開昭５９－６０６６０（ＪＰ，Ａ）
特開平６－５１９７２（ＪＰ，Ａ）
特開平２－１５１９５６（ＪＰ，Ａ）
特開平７－１０４９８１（ＪＰ，Ａ）
特開平５－３２４３３９（ＪＰ，Ａ）
特開昭６３－２２８３３５（ＪＰ，Ａ）
特開平１－２９１３３３（ＪＰ，Ａ）
特開平１－１０６２３０（ＪＰ，Ａ）
特開平１－２１１０３６（ＪＰ，Ａ）
特開平３－６６２７（ＪＰ，Ａ）
岩本康・他，「プロセス指向による交換ソフトウェア自動試験環境」，情報処理学会研究報告，日本，社団法人情報処理学会，１９９２年 7月２１日，Vol.92，No.59，pp.1-8
Andrew S. Tanenbaum，「原書２版 タンネンバウム コンピュータネットワーク」，日本，丸善株式会社，１９９２年 7月２５日，初版，pp.284-286，ISBN:4-621-03699-8

(58)調査した分野(Int.Cl.[7]，ＤＢ名)
G06F9/06，
G06F15/00，
G06F9/46-9/54，
G06N5/04，
G06F9/00，
G06F1/00，
G06F3/14，
G06F3/00，
CSDB(日本国特許庁)